Rhythmyx

Customizing FastForward for Web Content Management

5.7

Printed on 7 October, 2005

This document was created using ***AuthorIT™, Total Document Creation*** (see AuthorIT Home - http://www.author-it.com).

Schema documentation was created using XMLSpy™.

Percussion Software
600 Unicorn Park Drive
Woburn, MA 01801 U.S.A
781.438.9900
Internet E-Mail: technical_support@percussion.com
Website: http://www.percussion.com

Contents

Introduction 5

Overview 7

Initial Implementation Tasks 8
Customizing FastForward Applications 9
Secondary Implementation Tasks 10
Specialized Implementations 11
Sites and Navigation 12
Workflows in Rhythmyx 13

Modeling and Design 15

Modeling the CMS 16
Modelling Content Editors 21
Modelling Content Assemblers 22
    Global Templates 25
Modelling Workflow 27
Modelling Publishing 30
Corporate Investments Development Plan 33
    Scope 33
    Content Type Definitions 33
Summary 41

Customizing FastForward Applications 43

Adding New Fields to a Content Editor 44
Adding New Fields to an Assembly Resource 47
Modifying Variant Templates 49

Extending FastForward Applications 51

Implementing a New Content Type 52
    Creating a New Content Editor 53
    Creating a New Content Assembler 63

Specialized Implementations 103

Implementing Text Extraction and Non-Text Assembly 104
    Text Extraction 105
    Creating the Bio Content Editor 109
    Creating the Bio Assembly Application 111
    Creating a Non-Text Assembler 113
    Testing the Bio Content Type 115
    Continuous Conversion Example 117

Migration Example 125
WebImageFX....127
sys_WebImageFX and the WebImageFX Editor 127
Implementing WebDAV in Rhythmyx's Installation of Tomcat 136
Adding Additional WebDAV Servlets in Rhythmyx's Installation of Tomcat....141
Editing a WebDAV Configuration File....142
Simple Example Implementation of WebDAV 144

Sites and Managed Navigation 155

Using Managed Navigation 158
Navigation Communities....158
Navigation Reset 159
Setting up a Site....161
Creating Site Structures in Rhythmyx 161
Adding Navigation to Rhythmyx Site Folders....172
Creating a NavTree....172
Creating a Navon....172
Creating a NavImage....176
Maintaining a Site....177
Adding Content to a Site Folder 177
Assigning a Content Item to Site section....177
Locating an Item with Impact Analysis....178
Locating an Item within a Site Tree 179
Removing a Content Item from a Site Section 181
Editing an Item in a Site Section 181
Editing a Navigation Item....181
Editing a Public Navigational Element....182
Splitting Navigation Sections 182
Merging Navigation Items....185
Reordering a Submenu 186
Defining a Global Template for a Site Section....187
Customizing Navigation Look and Feel 188
Customizing Navigation CSS....188
Creating Custom Navigation Variants....194
Implementing Managed Navigation as Server-side Includes....203
Registering New Page Variants for the Server-side Include Files....203
Modifying XSL Stylesheets for Server-side Includes 206
Adding Modified Stylesheets to the Navon Assembly Application 209
Modifying the Rhythmyx Slots Stylesheet to Process Server-side Includes 214
Adding New Location Schemes to Publish Contexts 214
Validating the Managed Navigation Using Server-side Includes Implementation....218

About Workflows in Rhythmyx 221

Copying a Workflow 222
Maintaining Workflows....223
Edit Workflow Page 224
Editing a Workflow 225
Deleting a Workflow 225
Viewing a Workflow 226
About the Workflow Administrator....226
Mandatory Relationships and Workflows 227

Workflow Roles....229
Edit Role Page in Workflow....229
Creating a New Role in Rhythmyx Workflow....230
Editing a Role in Workflow....230
Deleting a Role in Workflow....230
Maintaining Server Roles....231
States....243
Edit State Page....243
Creating a New State....244
Editing a State....245
Deleting a State....245
Extending Publishable States....245
Transitions....247
Edit Transition Page....247
Creating a New Transition....249
Editing a Transition....249
Deleting a Transition....250
Maintaining Transition Roles....250
Aging....250
Notifications....257
Setting the Workflow Configuration Properties for Notification....257
Edit Notification Page....257
Creating a Notification....258
Editing a Notification....258
Deleting a Notification....258
Including User Comments in Notification E-mail Messages....258
Assignment....261
Role Assignment....261
Notification Assignment....263
Implementing Quick Edit....267

Index **271**

CHAPTER 1

Introduction

Welcome to Customizing FastForward for Rhythmyx Web Content Management. If you are reading this document, you are at a point in your Content Management implementation warranting a more in-depth look at FastForward including:

- In depth modeling of an existing Site;
- Customization or creation of Rhythmyx applications to meet unique Site requirements;
- Implementation of support functionalities including WebDAV, Text Extraction, and WebImageFx;
- Understanding of Sites and Navigation to facilitate the creation of new Sites;
- Extension of existing Workflows to include new groups of Business Users.

Customizing FastForward for WCM is designed to provide Rhythmyx Developers the information and support necessary to develop customized contribution mechanisms, extend existing Assembly applications, and publish unique Sites.

CHAPTER 2

Overview

This chapter introduces the concepts and approaches explained throughout this manual. At a high level, we will provide a road-map used to successfully implement Rhythmyx and several supporting technologies. Each sub-section increases in complexity, yet follows a logical progression followed by most development situations encountered when managing a Site with Rhythmyx. These steps are:

- Modeling and Design

  Modeling and Design is the process of dissecting your existing Site, breaking it down into implementable pieces, and tying each piece to existing applications and functionality provided with FastForward;

- Customizing FastForward Applications

  During the initial phase of implementation we deal with adding new fields to existing Content Editors and Assemblers including Variant templates, when dealing with Content Types, that are very similar to those provided by FastForward;

- Extending FastForward Applications

  This phase of implementation includes the creation of new Content Editors and Assemblers to meet the needs of an existing Site. We will also discuss the use of Shared Variants and the creation of new Local and Global Templates;

- Specialized Applications

  Text Extraction, WebImageFX, and Automated Indexes allow for specialized methods of inserting, editing, and presenting content. This section will focus on the creation of Content Editors and Assemblers showcasing these concepts.

- Sites and Navigation

  The FastForward content is organized into two Sites, Enterprise Investments and Corporate Investments. These Sites share content and similar site structures, but have unique navigation. Sites and Navigation explains how these two Sites were implemented and how to extend each Site and its Navigation menus.

- About Workflows in Rhythmyx

  The life cycle for Content Items is defined by Workflow. FastForward ships with two basic Workflow; One for Content Items and one for Navigation Items.

Initial Implementation Tasks

Initial implementation tasks are those activities common to all Rhythmyx implementations you need to understand before executing. These pieces of the Rhythmyx puzzle are critical to understand before embarking on their execution. Without the ability to model a Rhythmyx managed site, creating Content Editors to support the necessary Content Types often results in failed efforts. Gathering the appropriate data and meta-data for Items of each Content Type must occur before these Items can be published for consumption by site visitors. The proper foundation of knowledge followed by a well defined progression of activities is the only approach that will result in a maintainable, effective, and scalable Rhythmyx managed Site.

This process begins with the proper modeling of your existing site.

Figure 1: Pages in Corporate Investments Site

Up to now, we have assumed that the existing FastForward implementation meets the requirements of your site. At this point, in Customizing, we have realized a need to create new applications to solve any outstanding Content Management problems that have surfaced from the initial configuration. The first step is to get a better understanding of the problem with proper modeling.

Customizing FastForward Applications

During the configuration phase of implementing a Site with FastForward, very little change was imposed on any of the existing applications. The intent of configuring is to fit your existing site needs into a set of existing applications. It is easy to jump in and create new applications from scratch to meet all of our design needs. This is both time consuming and unnecessarily error prone. Why create a new Content Editor when existing Editors exist meeting 95% of our needs? Instead, our time is better spent tweaking these applications to meet the remaining 5% of a need while we get a better understanding of how the system and its supporting Editors and Assemblers co-exist.

This section of the manual focuses on the minor modifications that are often made to applications. Minor modifications include adding new fields to a Content Editor and displaying new content with an existing Assembler.

Secondary Implementation Tasks

At some point, modifying existing applications is not sufficient. The creation of a new Content Type becomes necessary when new Content Editors and Assemblers are required. In this chapter we will discuss:

- How to create new Content Editors and Assemblers;
- Apply customized Global and Local templates;
- Register new Variants against the new Assemblers.

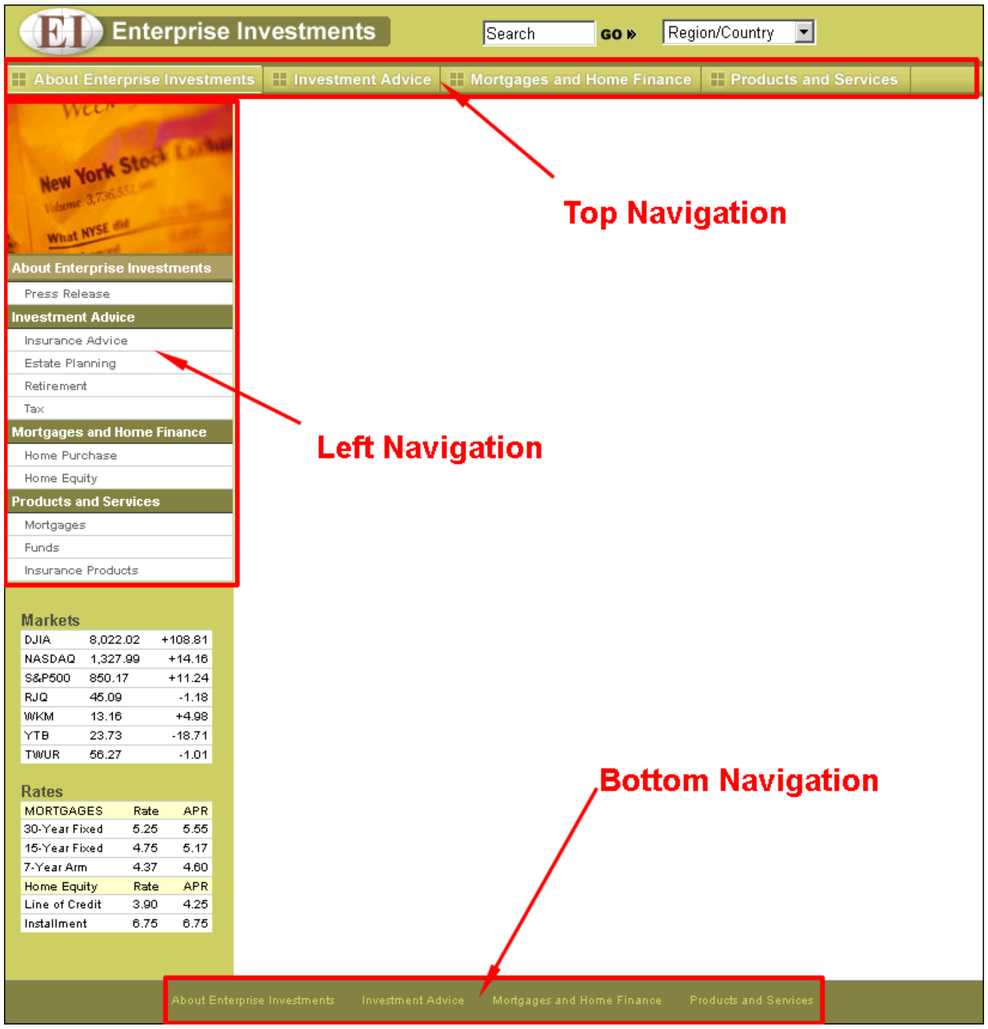
Specialized Implementations

Standard Content Editors and Assemblers can be modified to provide unique methods of gathering, parsing, and presenting data. Text Extraction, WebImageFX, and WebDAV allow Content Contributors alternate methods of working with their content. Through Text Extraction, content locked up in binary files can be easily imported as Content Items and managed with Rhythmyx. WebImageFX allows for a familiar Microsoft Paint-like environment for editing Images within a Content Editor. WebDAV removes the necessity of using the Content Explorer from those that work in third party applications, like Adobe Photoshop, that are WebDAV compliant and lets these users create, edit, and delete assets using the tools they are most comfortable with.

On the presentation side of managing content is the Automated Index. The Automated Index allows content contributors the ability to display lists of Content Items (and their Content) based on simple or complex queries. One such query, "The five most recent Press Releases" is a common Automated Index.

Sites and Navigation

Rhythmyx provides out of the box solutions for common navigation menus. In addition, intuitive XML Variants allow you to customize navigation when the standard solutions do not apply. For example, a Flash driven navigation menu is not provided by Percussion, but the necessary XML for such an application to consume is provided out of the box.



Markets

DJIA	8,022.02	+108.81
NASDAQ	1,327.99	+14.16
S&P500	850.17	+11.24
RJQ	45.09	-1.18
WKM	13.16	+4.98
YTB	23.73	-18.71
TWUR	56.27	-1.01

Rates

MORTGAGES	Rate	APR
30-Year Fixed	5.25	5.55
15-Year Fixed	4.75	5.17
7-Year Arm	4.37	4.60
Home Equity	Rate	APR
Line of Credit	3.90	4.25
Installment	6.75	6.75

Figure 2: Navigation Snippets on the Enterprise Investments Global Template

As shipped, Managed Navigation allows implementers to create common navigation without needing complex XSL while creating navigational structure based on the site's folder structure. A combination of specialized Communities, Content Types, and Effects allows for a multitude of navigation structures while masking the complexities from Content Contributors. Implementers and Web Masters can use the Managed Navigation elements to create new sections of the Site, move pages from one section to another, change the navigational label or image on a section, reorder sections, and if necessary remove sections from the navigation menu completely.

Workflows in Rhythmyx

A workflow is a business process that defines a sequence of events. Each Content Type is assigned a workflow. The workflow defines and controls the sequence of events that occur to Items of that Type throughout the approval process.

FastForward ships with several Workflows.

- Simple Workflow
- Standard Workflow

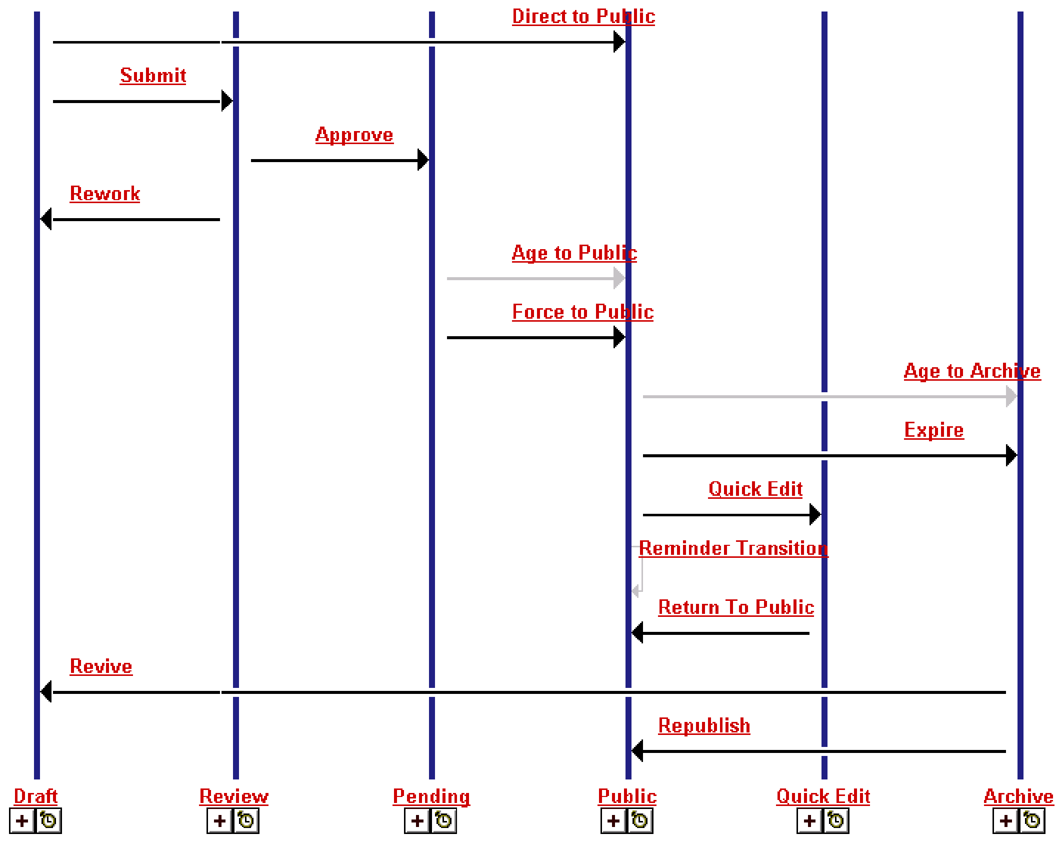


Figure 3: Standard Workflow Diagram

Often, we tend to start out with a more complicated Workflow then is actually necessary. The best plan for an efficient Workflow is to make it as simple as possible. Once a Workflow is being used it is easier to expand on it then reduce it. Adding new States, Transitions, and Roles to a Workflow can facilitate the awareness of content among stakeholders, but can quickly become unruly. Keep it simple.

When a single Workflow no longer meets the needs of your business, you can implement entire new Workflows. The most common reason to create a new Workflow is the creation of a new Site. At minimum, most Site Content can be managed by two Workflows; a standard workflow for site content and a simple workflow for navigation Items and utility content (such as Automated Indexes) created by Web Masters and other site administrators.

CHAPTER 3

Modeling and Design

A model for your Managed Site is developed in a set of discovery sessions at the outset of the implementation, followed by the documentation of the intended implementation in the form of a development plan. This event focuses on working with your various groups, units, partners, and internal organizations to uncover Corporate scope and needs. The purpose of the event is to gain high-level project goals, flush out possible difficulties and hurdles, understand the architectural environment, and map these findings against the Rhythmyx model.

Modeling should touch upon the following topics:

- Understanding Design Processes
- Object Naming Conventions
- Rhythmyx Methodology, Development Environments, and Deployment
- Rhythmyx Security, Publishing, and Administration
- Validation, Review, and Assignments
- Understanding common elements and establishing foundations

Modeling the CMS

Your Web Site may already exist. When you design a web site, most of the formatting and directory structures have already been defined. We recommend that you spend a great deal of time breaking down your web site and generating a thorough project plan before you consider doing any development. The temptation is to begin developing areas of your site blindly. This will only wind up costing your vast amounts of wasted time later in the process. A core piece of the 'web site breakdown' is to define your directory structure and isolate directory inconsistencies. Resolve these inconsistencies before moving forward with additional development. Once you have done this, print out paper copies of web pages, grab a marker or pencil, and begin the design and breakdown of your web site.

If you already have a partial implementation of the CMS, 'what's already done' includes the existing Content Editors, directory structure (context) definition, Content Assemblers, Communities, Workflows, etc. These will serve as a starting point for any new augmentations that you make to your Rhythmyx Content Manager.

Identify Page Types

The first step is to identify the distinct page types that the site will have. A good place to start is by identifying the site's hierarchy. Most sites are designed using a section/subsection structure. Products and Services, Press Releases, About Us, Investment Advice, and Support are examples of typical sections.

Each section and subsection should be named and grouped into a tree-like structure. Other structures may be used as necessary to represent more complex interrelationships.

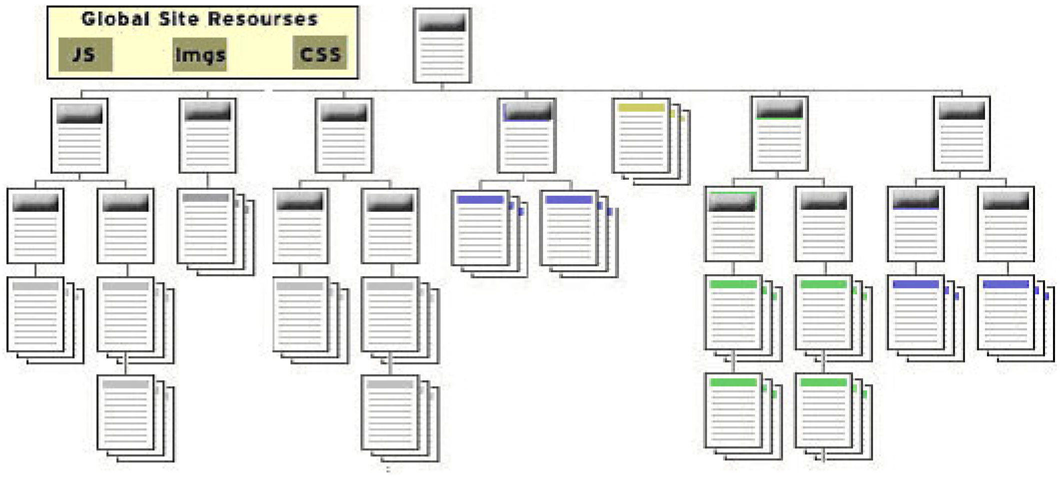


Figure 4: Typical Site Structure

Page Layout

The next step is to provide a rough design of each page type. Paper prototypes, design images or block diagrams are sufficient for this step.

During this process, you should be able to reduce the page types by normalizing the design. If two page types have the exact same layout and differ only in the format or type of information presented, those two page types can possibly be collapsed into one Content Type with multiple Variants for presentation.

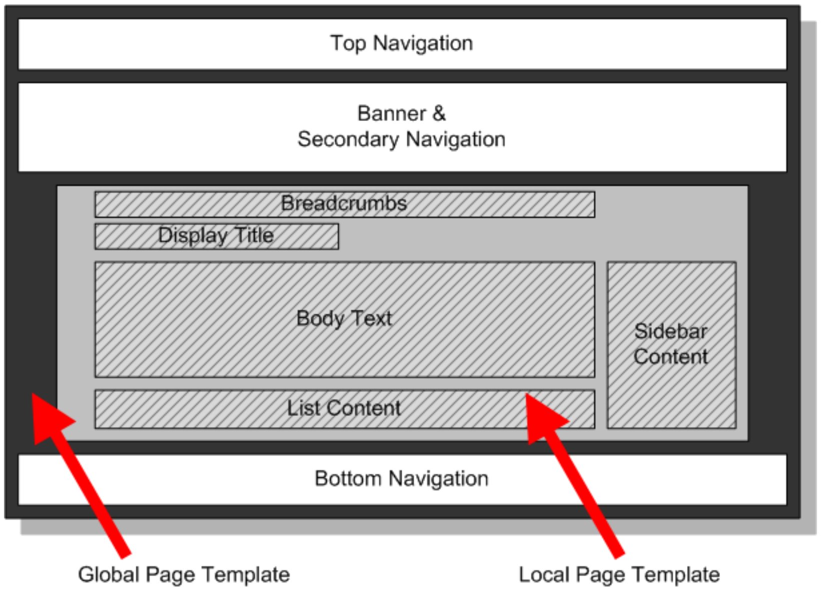


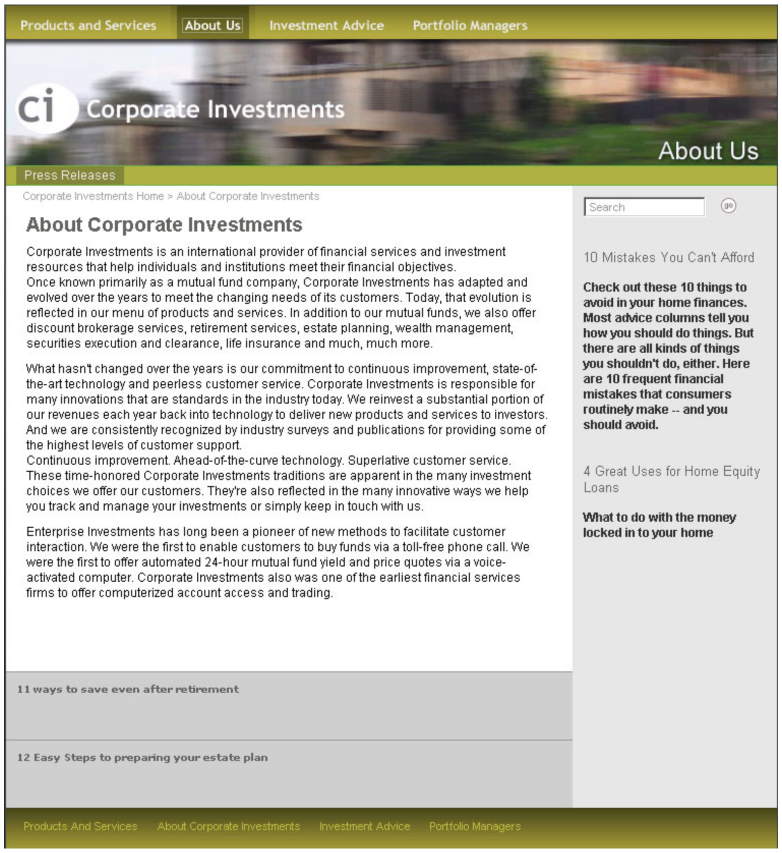
Figure 5: Generic Page Structure

Identify Components

Once all the page designs have been created, the next step is to break each page into components. A component is used whenever you encounter one of the following:

- A re-usable component (navigation bars, banners, etc.) that will be the same across several different page layouts.
- A representation of another page. The classic example is a link to an article page that contains the title and abstract.
- An image or other file (e.g. PDF, DOC, etc.) that will be managed by the CMS.
- A list of other content items (e.g. related articles, “see also”, etc.)
- A piece of dynamic data that comes from some other system (e.g. news feeds, stock tickers, etc.)

Let’s start at the top and take an example Generic page from the site and break it into components. We will the Generic page from the Corporate Investment Site as an example of content type decomposition.



About Corporate Investments

Corporate Investments is an international provider of financial services and investment resources that help individuals and institutions meet their financial objectives.
Once known primarily as a mutual fund company, Corporate Investments has adapted and evolved over the years to meet the changing needs of its customers. Today, that evolution is reflected in our menu of products and services. In addition to our mutual funds, we also offer discount brokerage services, retirement services, estate planning, wealth management, securities execution and clearance, life insurance and much, much more.

What hasn't changed over the years is our commitment to continuous improvement, state-of-the-art technology and peerless customer service. Corporate Investments is responsible for many innovations that are standards in the industry today. We reinvest a substantial portion of our revenues each year back into technology to deliver new products and services to investors. And we are consistently recognized by industry surveys and publications for providing some of the highest levels of customer support.
Continuous improvement. Ahead-of-the-curve technology. Superlative customer service. These time-honored Corporate Investments traditions are apparent in the many investment choices we offer our customers. They're also reflected in the many innovative ways we help you track and manage your investments or simply keep in touch with us.

Enterprise Investments has long been a pioneer of new methods to facilitate customer interaction. We were the first to enable customers to buy funds via a toll-free phone call. We were the first to offer automated 24-hour mutual fund yield and price quotes via a voice-activated computer. Corporate Investments also was one of the earliest financial services firms to offer computerized account access and trading.

11 ways to save even after retirement

12 Easy Steps to preparing your estate plan

10 Mistakes You Can't Afford

Check out these 10 things to avoid in your home finances. Most advice columns tell you how you should do things. But there are all kinds of things you shouldn't do, either. Here are 10 frequent financial mistakes that consumers routinely make -- and you should avoid.

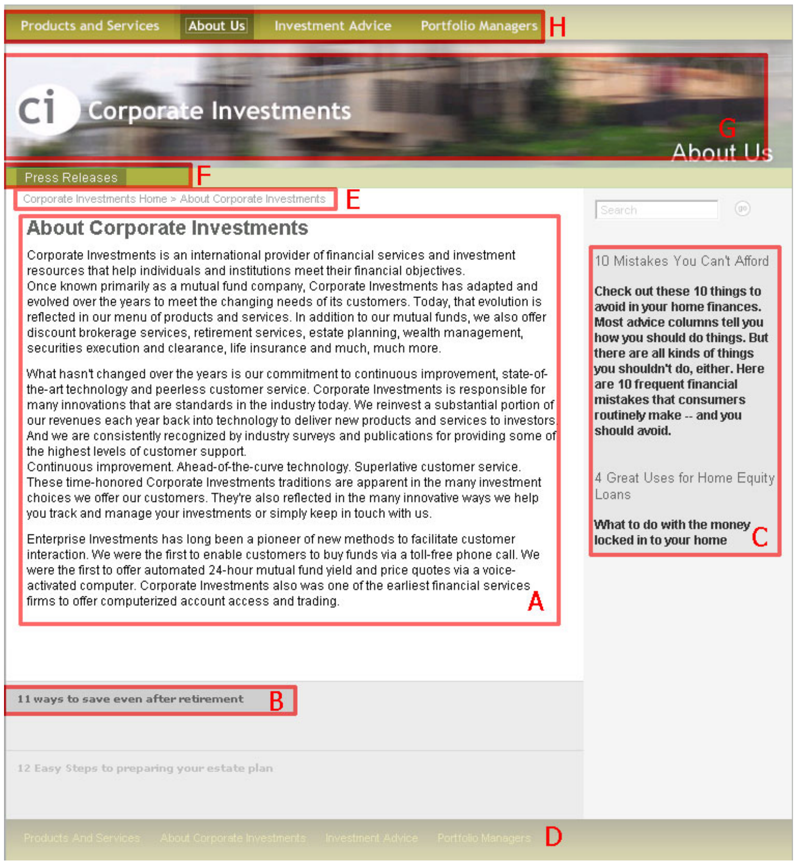
4 Great Uses for Home Equity Loans

What to do with the money locked in to your home

Figure 6: Generic Page

The page includes content from several different Items of the Generic Type and some from the Navon and NavImage Types. The bulk of the content is from the Item itself.

The Generic page has at least eight page components (identified below). This number is uncertain because content in the Sidebar and List Slot is sub-categorized into different components.



About Corporate Investments

Corporate Investments is an international provider of financial services and investment resources that help individuals and institutions meet their financial objectives.
Once known primarily as a mutual fund company, Corporate Investments has adapted and evolved over the years to meet the changing needs of its customers. Today, that evolution is reflected in our menu of products and services. In addition to our mutual funds, we also offer discount brokerage services, retirement services, estate planning, wealth management, securities execution and clearance, life insurance and much, much more.

What hasn't changed over the years is our commitment to continuous improvement, state-of-the-art technology and peerless customer service. Corporate Investments is responsible for many innovations that are standards in the industry today. We reinvest a substantial portion of our revenues each year back into technology to deliver new products and services to investors. And we are consistently recognized by industry surveys and publications for providing some of the highest levels of customer support.
Continuous improvement. Ahead-of-the-curve technology. Superlative customer service. These time-honored Corporate Investments traditions are apparent in the many investment choices we offer our customers. They're also reflected in the many innovative ways we help you track and manage your investments or simply keep in touch with us.

Enterprise Investments has long been a pioneer of new methods to facilitate customer interaction. We were the first to enable customers to buy funds via a toll-free phone call. We were the first to offer automated 24-hour mutual fund yield and price quotes via a voice-activated computer. Corporate Investments also was one of the earliest financial services firms to offer computerized account access and trading.

10 Mistakes You Can't Afford

Check out these 10 things to avoid in your home finances. Most advice columns tell you how you should do things. But there are all kinds of things you shouldn't do, either. Here are 10 frequent financial mistakes that consumers routinely make -- and you should avoid.

4 Great Uses for Home Equity Loans

What to do with the money locked in to your home

11 ways to save even after retirement

12 Easy Steps to preparing your estate plan

Figure 7: Generic Page Broken Down

A. Local Page Content. For the Generic Page, this includes the Display Title and Body Text.

B. Related Content in the List Slot. This Content represents a different Item of the Generic Type.

C. Related Content in the Sidebar Slot. Once again, content from a different Generic Item is represented on this page. The presentation of this Item is different than the List Slot.

D. Bottom Navigation. An auto populated Slot (nav_bottom) filled with Content from the Navon Content Type.

E. Breadcrumb Navigation. Another auto populated Slot (nav_breadcrumb) filled with Navon Content.

F. Secondary Top Navigation Auto populated content in a third Navigation Slot (nav_TopDescendent) displaying a particular Site section's sub-sections.

G. Banner. This image is Site Section dependant and is part of the Secondary Top Navigation Item.

H. Top Navigation. The final Navigation Slot (nav_top) populated with Images linking to each of the Site's major sections.

For the purposes of the content type analogy, we will take the Generic content type and decompose it a bit further. We will assume for ease of explanation that the only areas of the site where the Generic content type appears in the above described page is within sections A, B, and C.

HTML Prototypes

When creating HTML prototypes of pages, the W3C's XHTML 1.0 recommendations (http://www.w3.org/TR/xhtml1) should be followed as closely as possible. Some of the differences between XHTML and HTML 4 include:

• Documents must be well-formed;

• Element and attribute names must be in lower case;

• Attribute values must always be quoted;

• Empty elements must be closed.

The W3C provides a tool, HTML Tidy (http://www.w3.org/People/Raggett/tidy/), which can be used to validate the prototype pages. Tidy is also available when using the XSpLit tool provided with Rhythmyx.

In addition, the use of cascading style sheets (CSS) is strongly encouraged, as is storing JavaScript functions in separate files, rather than in the HTML document itself. This further separates code from content, and allows simple and efficient changes to be made across the entire site, as well as allowing for smaller page sizes.

You will use the HTML prototypes to the generate XSLT files used in your Content Assemblers. As stated, Rhythmyx's XSpLit allows for this generation without you or your team needing to interact with the XSLT (that the splitter generates) directly. However, it does require that you have a strong mastery of how to effectively break you source HTML into functional components. We also encourage you to always generate well-formed XHTML so that there are no formatting "surprises" when HTML Tidy is run on your source as Rhythmyx splits your pages.

Extract Component HTML

Determine which pieces of content on a given page must be handled separately. The management of a banner, for example might be handled separately from the pages that contain that banner.

All content that is included as part of another page is considered a 'component.' In the example Generic Page above, only the Content in "A" is unique to this page and is considered to be Local Content. All other content belongs to other Content Items. While those content items may be of the same type as this page, they will be handled as discrete items by the system, and will be included in the page by reference at creation (preview and publishing) time. Each of these components will be generated by the Content Assemblers for that particular content type.

In the displayed example, the Generic Page assembler will generate the page and will request each 'component' from their appropriate assemblers. Each assembler will then create the HTML for that small area (components "B", "C", "D", "E", "F","G", and "H" in the example above) and return that HTML to the Generic assembler, which, in turn, will insert the HTML into the correct section of the page. In this way, each Content type controls the look and feel of its own output, no matter where in the system that output is used.

Modelling Content Editors

Identify Methods of Data Collection

Consider how the display content and metadata is entered into the CMS. Using the fictitious Generic Content Type as an example:

- Will the body field be contained in a Microsoft Word file that will be uploaded to the CMS, or will an author type the body directly into a CMS form?
- Will the title field be contained inside the Word file (and need to be parsed), or will it be entered into a form?
- Will the Body field on the form be rich or plain text?
- Can Snippets of other Content Types be included in the Body Field?
- Are the Page's Keywords gathered via a simple input=”text” element or as a drop-down list?
- If data is entered via a drop-down list, where are possible values stored? Is the data hard-coded or is it coming via a lookup to a table or application?
- If these values are coming from an outside source, is that source native to a Rhythmyx table or application or is it from an outside source?
- Is this data shared across more than one content type?

Hopefully what is clear here is that there is a great deal to consider about collecting and managing content item display data and metadata. When you begin to define your own Content Type, you will need to carefully reflect on all of these very important details before even setting out to write a Statement of Work document.

Consider whether the metadata is required or optional. If required, must the metadata be present whenever the Item is edited, or only before the Item is transitioned to a new state?

Plan your Schema

After you have fully analyzed the site that you want to content manage and determined the content types and metadata, you will need to develop a database schema.

Each discrete Content Type may have its own table. The table must have contentid and revisionid columns, as well as one column for each piece of metadata. When the Content Editor creates a piece of content, it will write one value per revision of each distinct piece of content. Therefore, repeating metadata, such as information from checkboxes, should be stored in a separate table. This child table must include the required CONTENTID and REVISIONID columns, as well as one column for each metadata field.

In the case of simple repeating metadata, such as checkboxes, the child table will only need to contain one additional field to contain the values of selected boxes. In the case of more complex repeating data, such as a table of repeating information, you will need the two required fields as well as one field per piece of metadata.

Modelling Content Assemblers

At this point you must identify what data is unique to each page. The Generic Page, for example, only uses the Item's Display Title and Body Text. However, the main page section may actually be composed of many fields. Consider how you will use the data across output formats (Variants) and how you want the information entered into the system. You may have pieces of data that are displayed together on your Site but are entered as separate Content Items. Conversely, the content displayed in sections A, B, and C are all derivative of Items of the same Type. They differ in the Variant being displayed. For the Generic Content Type, several Variants are available:

- P - CI Generic

  A full page Variant used as a details page on the Corporate Investments Site.
- P - EI Generic

  A full page Variant used as a details page on the Enterprise Investments Site.
- S - Callout

  A short teaser or summary paragraph used to describe the contents of the Full Page.
- S - Image Link

  Images associated with the Generic Item, in the Image Link Slot, are used as links back to the Full Page.
- S - Title Callout Link

  The title and callout of the Item are used as links back to the Full Page.
- S - Title Link

  The title is used alone as a link to the Full Page.
- S - Title, Callout, and More Link

  The title and callout are used as a link back to the Full Page. This Snippet also contains the text "more" hyperlinked to the Full Page.

Variants prefaced with a "P" indicate Page Variants. A Page Variant represents a free standing HTML(or similar) file. Generally, Page Variants include any number of "S" or Snippet Variants. The concept of Pages and Snippets allows us to combine multiple Items of multiple Types to create unique pages that are interrelated. Since it is possible to have an unlimited number of Pages or Snippet Variants for any given Content Type, the final output of any given Item could take on endless variants in look and feel. Adding to this availability of multiple Snippets, we have the option to include or not to include these Snippets as necessary to a Page (or to other Snippets for that matter). The combinations can be as simple or as complex as necessary. For example, an Item of the Generic Type is created to describe a new Corporate Investments portfolio offering. The Page Variant is published as a landing page describing the portfolio. The title callout link Snippet is placed on the Home Page to help direct site visitors to the new offering.

Contributing to the amount of possible customization and facilitating the normalization of page designs is the concept of Global Templating. A Global Template can be pictured as the c-clamp that surrounds the page content with the Site's look and feel. For Corporate Investments, our sample site, the Global Template contains most of the navigation elements and the basic page color scheme.

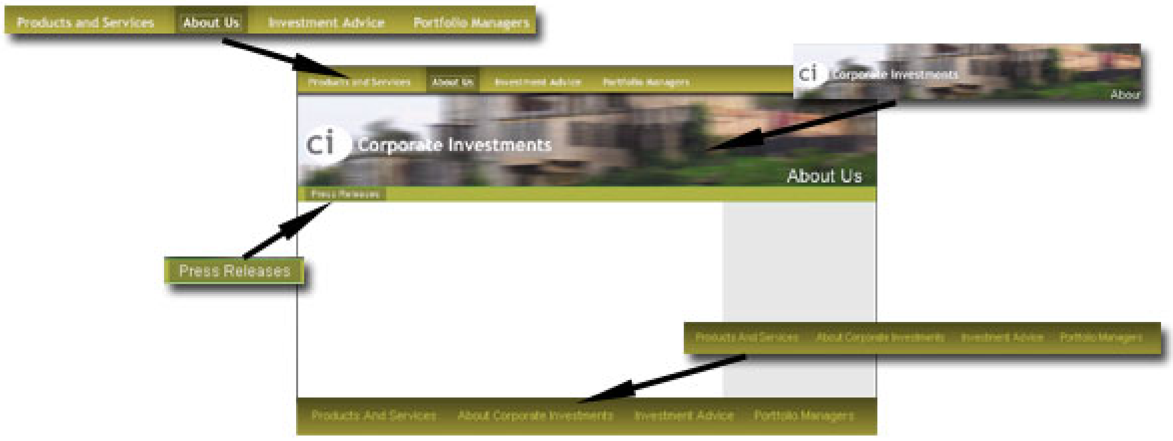
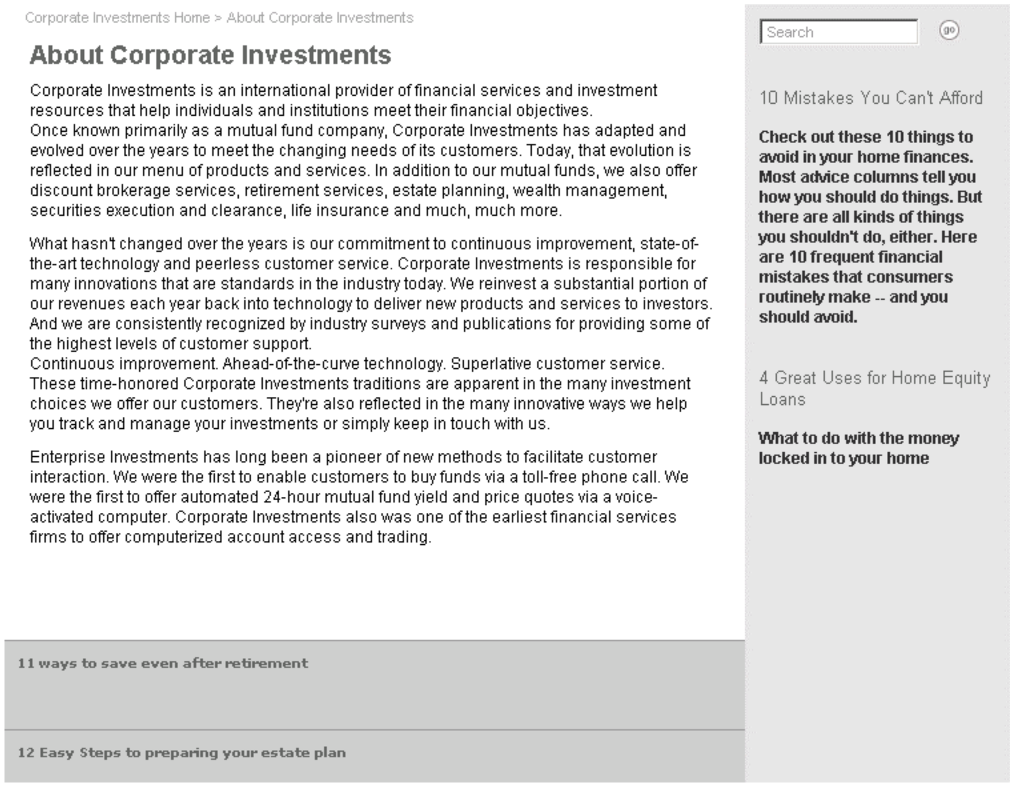


Figure 8: Components of a Global Template

This Global Template can be applied to any of the available Page Templates, such as the Generic Page.



Corporate Investments Home > About Corporate Investments

About Corporate Investments

Corporate Investments is an international provider of financial services and investment resources that help individuals and institutions meet their financial objectives.
Once known primarily as a mutual fund company, Corporate Investments has adapted and evolved over the years to meet the changing needs of its customers. Today, that evolution is reflected in our menu of products and services. In addition to our mutual funds, we also offer discount brokerage services, retirement services, estate planning, wealth management, securities execution and clearance, life insurance and much, much more.

What hasn't changed over the years is our commitment to continuous improvement, state-of-the-art technology and peerless customer service. Corporate Investments is responsible for many innovations that are standards in the industry today. We reinvest a substantial portion of our revenues each year back into technology to deliver new products and services to investors. And we are consistently recognized by industry surveys and publications for providing some of the highest levels of customer support.
Continuous improvement. Ahead-of-the-curve technology. Superlative customer service. These time-honored Corporate Investments traditions are apparent in the many investment choices we offer our customers. They're also reflected in the many innovative ways we help you track and manage your investments or simply keep in touch with us.

Enterprise Investments has long been a pioneer of new methods to facilitate customer interaction. We were the first to enable customers to buy funds via a toll-free phone call. We were the first to offer automated 24-hour mutual fund yield and price quotes via a voice-activated computer. Corporate Investments also was one of the earliest financial services firms to offer computerized account access and trading.

11 ways to save even after retirement

12 Easy Steps to preparing your estate plan

Search

go

10 Mistakes You Can't Afford

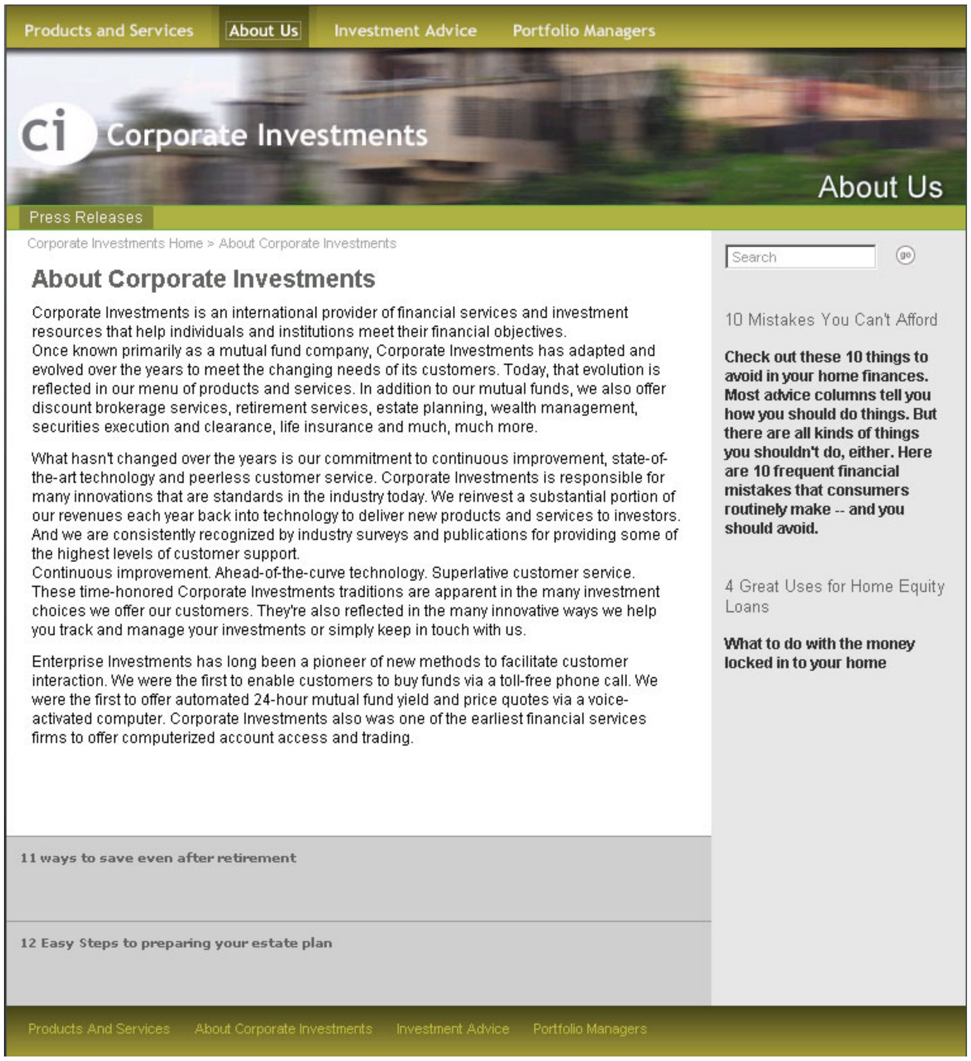
Check out these 10 things to avoid in your home finances. Most advice columns tell you how you should do things. But there are all kinds of things you shouldn't do, either. Here are 10 frequent financial mistakes that consumers routinely make -- and you should avoid.

4 Great Uses for Home Equity Loans

What to do with the money locked in to your home

Figure 9: Generic Page without Global Template

The combination of Global and Local Template yields a unique Page displaying the Content Item with a particular look and feel.



Corporate Investments Home > About Corporate Investments

About Corporate Investments

Corporate Investments is an international provider of financial services and investment resources that help individuals and institutions meet their financial objectives.
Once known primarily as a mutual fund company, Corporate Investments has adapted and evolved over the years to meet the changing needs of its customers. Today, that evolution is reflected in our menu of products and services. In addition to our mutual funds, we also offer discount brokerage services, retirement services, estate planning, wealth management, securities execution and clearance, life insurance and much, much more.

What hasn't changed over the years is our commitment to continuous improvement, state-of-the-art technology and peerless customer service. Corporate Investments is responsible for many innovations that are standards in the industry today. We reinvest a substantial portion of our revenues each year back into technology to deliver new products and services to investors. And we are consistently recognized by industry surveys and publications for providing some of the highest levels of customer support.
Continuous improvement. Ahead-of-the-curve technology. Superlative customer service. These time-honored Corporate Investments traditions are apparent in the many investment choices we offer our customers. They're also reflected in the many innovative ways we help you track and manage your investments or simply keep in touch with us.

Enterprise Investments has long been a pioneer of new methods to facilitate customer interaction. We were the first to enable customers to buy funds via a toll-free phone call. We were the first to offer automated 24-hour mutual fund yield and price quotes via a voice-activated computer. Corporate Investments also was one of the earliest financial services firms to offer computerized account access and trading.

10 Mistakes You Can't Afford

Check out these 10 things to avoid in your home finances. Most advice columns tell you how you should do things. But there are all kinds of things you shouldn't do, either. Here are 10 frequent financial mistakes that consumers routinely make -- and you should avoid.

4 Great Uses for Home Equity Loans

What to do with the money locked in to your home

11 ways to save even after retirement

12 Easy Steps to preparing your estate plan

Products And Services About Corporate Investments Investment Advice Portfolio Managers

Figure 10: Generic Page

At this point we have defined what data is to be collected and how it should be displayed. We will also want to determine what data must be collected to describe how the item is managed by the CMS. In other words, what metadata do we need to collect? Primarily we need to consider Workflow and Item delivery to the runtime server environment; Publishing. We need to ask early in the design phase:

- How is Workflow assigned to the Content Item?
- Who will be able to view, edit, and transition Items through a Workflow?
- Will multiple Workflows be necessary to manage Items of different Content Types?
- How does the Rhythmyx server know when and where to publish or un-publish the Item?

Typical examples of content type metadata are: Content Start Date, Content Expiration Date, Web Server location, the published Item's file type extension, Workflow assignment, Email Notification Reminder Date, etc. This information must be gathered when the Item is created to provide Rhythmyx an opportunity to process the Item correctly.

Global Templates

Typically a web site is structured in such a way as to have a number of static elements on each page, as well as navigation, establishing a wrapper around the main body of content. The details of the diagram below do not fully represent the structure of every possible site, but it is a fairly common one. Most web sites in some way or another conform to this "C-clamp" concept.

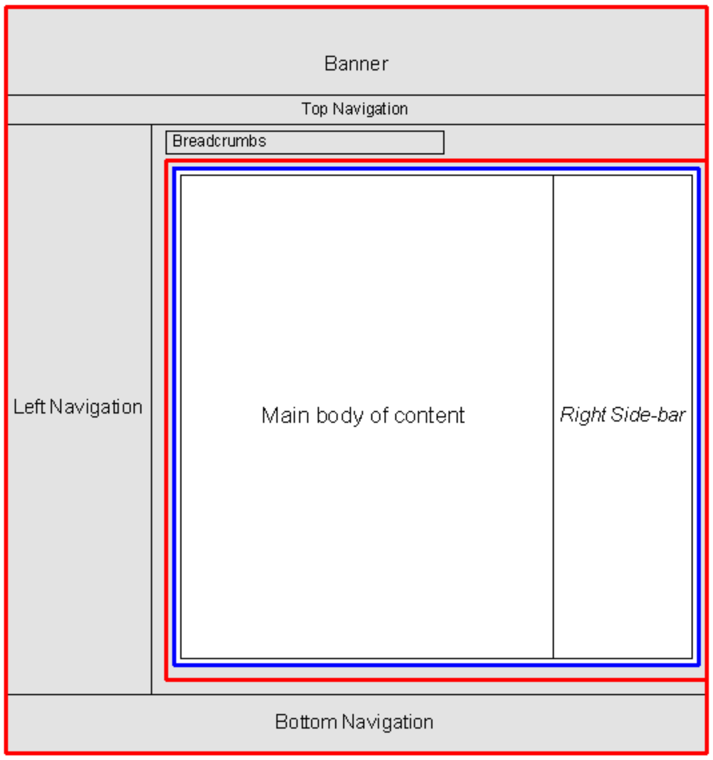


Figure 11: Global Templates

Prior to building assembly applications with Rhythmyx, it is a good idea to break down a site's structure and determine which elements of the page are common. By doing so, it is possible to implement the applications with an "outer template" (represented by the shaded, boxes above) in order to make design changes easier to accommodate, as well as provide a extensible inner templates that can efficiently be applied to multiple sites.

The Corporate Investments Global Template uses the same pieces of navigation, yet places them in slightly different locations.

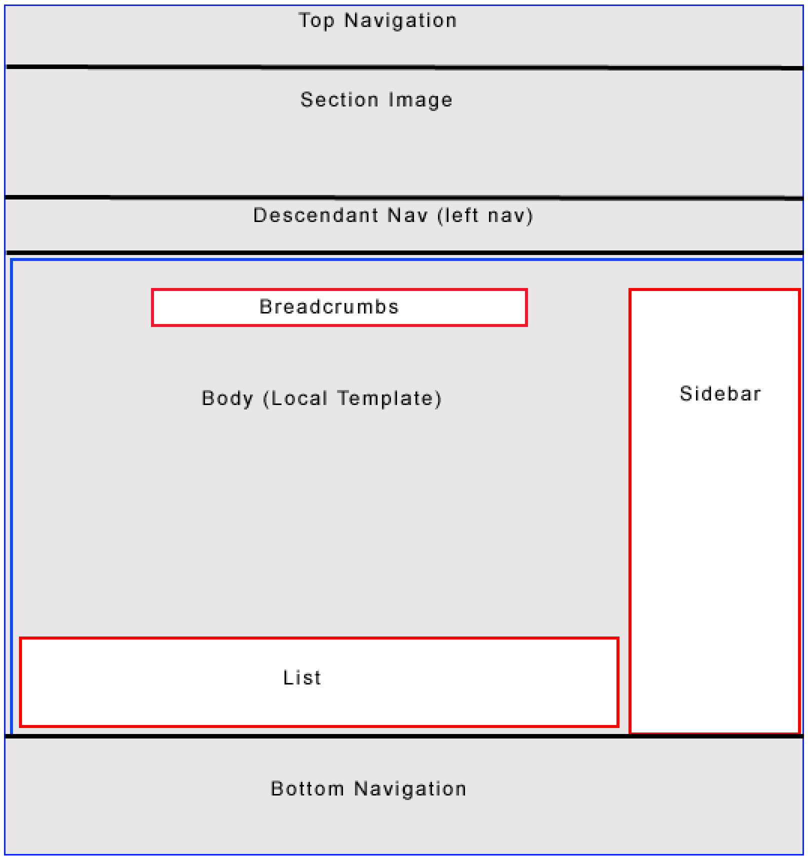


Figure 12: CI Wire Frame

Modelling Workflow

A Workflow is a business process that follows a sequence of events. In Rhythmyx, every Content Item must exist in a Workflow. The Workflow determines:

- the steps in the processing of the Content Item;
- which users can access the item at each phase in the process; and
- what the users can do with the item at that point.

In general, you need one Workflow per Content Type, although occasionally two or more Content Types can use the same Workflow. Different classes of Content Types, however, definitely need different Workflows. For example, text-rich Content Types require different Workflows than graphic or file management Content Types. Different users, generally with different skills, are responsible for the processes of creation and review of each class of content.

Identify Workflows

For each type of content, you will need to identify what Workflow is required. A good place to start is to look at any existing manual workflows that are in place.

Let's look at the example of a Press Release. A member of the marketing department may start a Press Release as a request. A copywriter will then take the request and generate the body. This is then sent to a member of the public relations department for review, who then forwards it to legal. Once legal has approved the press release, it's sent to the Webmaster, who will post it on the website on the start date. This manual workflow is easily translated to a Rhythmyx Workflow. Five roles are required:

- Marketing;
- Copywriter;
- Public Relations;
- Legal;
- Webmaster.

The Workflow consists of the following States:

- Request;
- Copy;
- Writing;
- PR Approval;
- Legal Approval;
- Pending Publish
- Public;
- Archived.

Role members move the Item from State to State with a Transition. A few of these Transitions can be identified:

- Submit for Copy;
- Submit to PR;
- Submit to Legal;
- Post for Publish;
- Publish;
- Unpublish.

This set of Transitions will also include Reject and or Rush Transitions to manage mistakes and the need for updates. Additionally, some transitions might occur based on meta-data associated with the Item. An Item might Un-publish based on a pre-defined expiration date. FastForward ships with a standard workflow that can be modified or replaced as necessary.

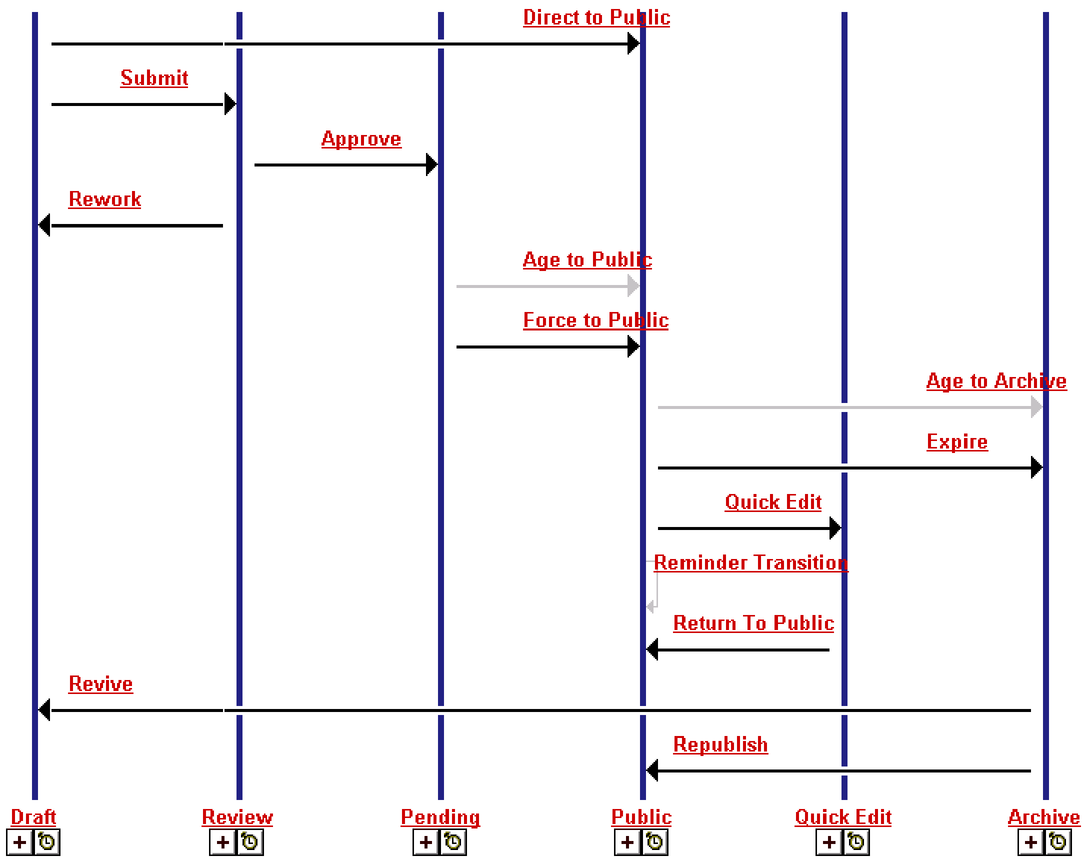


Figure 13: Standard Workflow Diagram

Identify Notification Rules

For each workflow state, identify any notification rules. Some possible triggers for a notification are:

- Entering a new state;
- Rejected;
- Ready for Publish;

- Having been in a state without action for too long.
- How long is “too long” for an item to remain in a state? Does this value change based on the content type, or the state?

In addition, a method for determining the notification recipient is defined. Some questions to consider include:

- Is the notification sent to an individual or a group?
- Does the email address vary for each content item, or is the same address used for all items in a state?
- What mail server will be used to relay the notifications? What are the authentication requirements of that mail server?

Modelling Publishing

Once the Content Item has been created a few things take place. The Item gets associated with a few other Items within the Site and all the appropriate data and meta-data for the Item is stored. Finally, a desired look and feel for the Item has been determined and the Item has been approved by all necessary parties. At this point the Item must be delivered to some external delivery mechanism for those traveling to your Site to see the Item. As a result of decoupled delivery, the final step in the planning phase is to decide and define how and where to deliver the Item to the Web Server, Application Server, Portal, or Database responsible for displaying the page (or pieces of the page).

The most common way to organize Items for delivery is through Site Folder Publishing. SFP allows Content Contributors to organize content in the Content Explorer in a Site Hierarchy that will mimic the resulting directory structure on the delivery server.

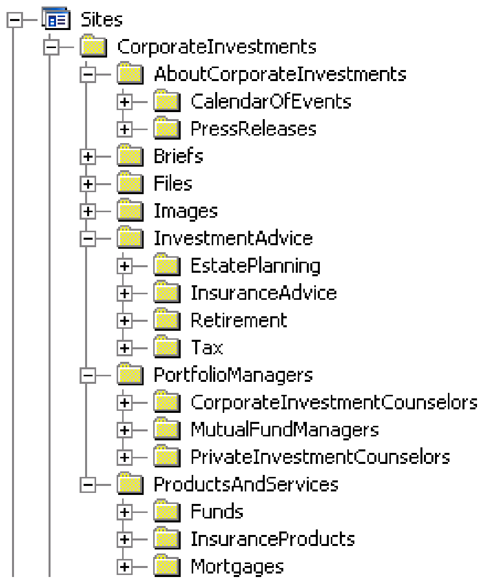


Figure 14: Site as defined in the Content Explorer

During a Publish, the Site to be published is selected, and the valid Items (those in the Public State) are sent to the physical Site. This is often accomplished over FTP but could also occur through a simple file system publish, too.

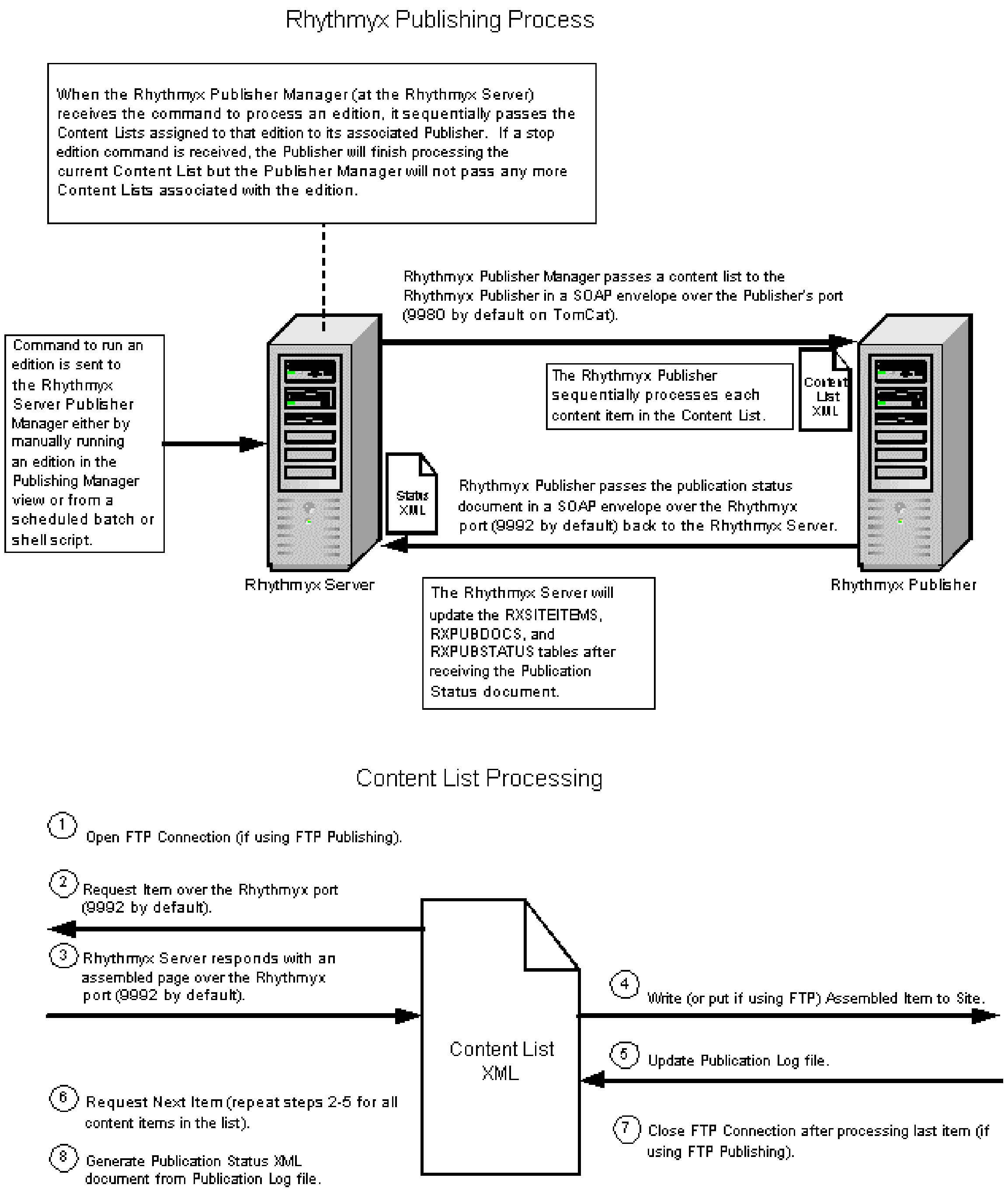

Rhythmyx Publishing Process
When the Rhythmyx Publisher Manager (at the Rhythmyx Server) receives the command to process an edition, it sequentially passes the Content Lists assigned to that edition to its associated Publisher. If a stop edition command is received, the Publisher will finish processing the current Content List but the Publisher Manager will not pass any more Content Lists associated with the edition.
Rhythmyx Publisher Manager passes a content list to the Rhythmyx Publisher in a SOAP envelope over the Publisher's port (9980 by default on TomCat).
Command to run an edition is sent to the Rhythmyx Server Publisher Manager either by manually running an edition in the Publishing Manager view or from a scheduled batch or shell script.
The Rhythmyx Publisher sequentially processes each content item in the Content List.
Content List XML
Status XML
Rhythmyx Publisher passes the publication status document in a SOAP envelope over the Rhythmyx port (9992 by default) back to the Rhythmyx Server.
Rhythmyx Server
Rhythmyx Publisher
The Rhythmyx Server will update the RXSITEITEMS, RXPUBDOCS, and RXPUBSTATUS tables after receiving the Publication Status document.
Content List Processing
1 Open FTP Connection (if using FTP Publishing).
2 Request Item over the Rhythmyx port (9992 by default).
3 Rhythmyx Server responds with an assembled page over the Rhythmyx port (9992 by default).
Content List XML
4 Write (or put if using FTP) Assembled Item to Site.
5 Update Publication Log file.
6 Request Next Item (repeat steps 2-5 for all content items in the list).
7 Close FTP Connection after processing last item (if using FTP Publishing).
8 Generate Publication Status XML document from Publication Log file.

Figure 15: Rhythmyx Publishing Process

If the target is a Database, Rhythmyx can be configured to deliver the data to a given schema. Before this can occur, though, a few pieces need to be defined within Rhythmyx:

- What Rhythmyx Publisher is used for Publishing?
- What content will be published?
- What Site will the content be delivered to?
- How often will these deliveries take place?

Corporate Investments Development Plan

The purpose of this section is to outline a sample development plan illustrating some of the requirements discussed during the Modeling section. The implementation details described will be used as a guideline for development of example applications. The goal of this is to provide the Corporate Investments development team with hands-on experience with Rhythmyx, accomplishing tasks varying in level of complexity.

Scope

The Modeling and Design sessions cover a wide variety of topics and formalized a number of Content Type definitions. It has been determined that the Rhythmyx FastForward for WCM package is a good fit for the Corporate Investments system, which will be installed with Rhythmyx. Some of the out-of-the-box Content Types will be modified to meet CI's requirements, as defined in this document. All others will be installed but may only be minimally customized as needed during the course of the engagement. Further customization will be tasked to the CI development team.

Content Type Definitions

The specifications for the Content Types below were defined during the Modeling and Design sessions by the CI project team.

- Bio
- Contacts (extension of existing FastForward application)
- Events Automated
- Job Listing

Global Templates

The Global template for Corporate Investments will be applied to all Page Variants except the Home Page. Items of the Home Page Type will only use a Local Template. Though similar, the Home Page Variant is unique compared to all other pages on the Site.

The CI Global Template will look as follows:

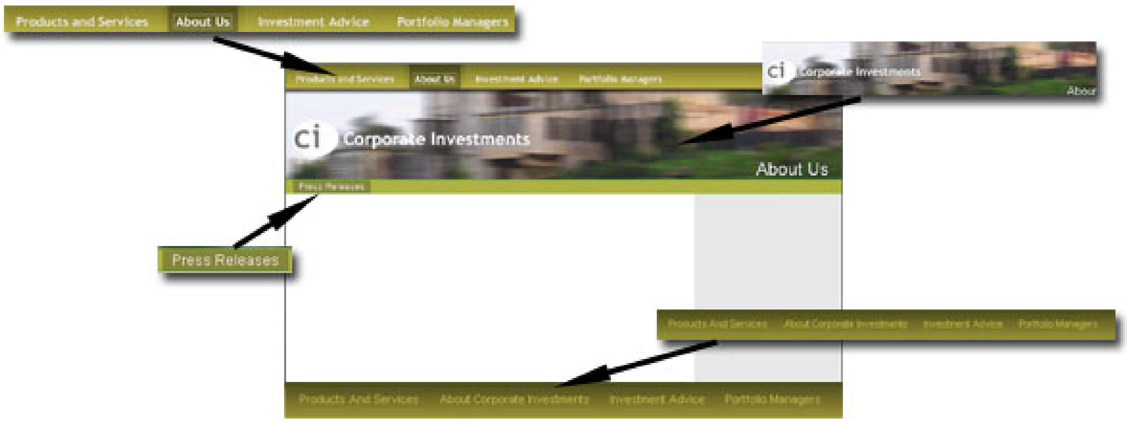
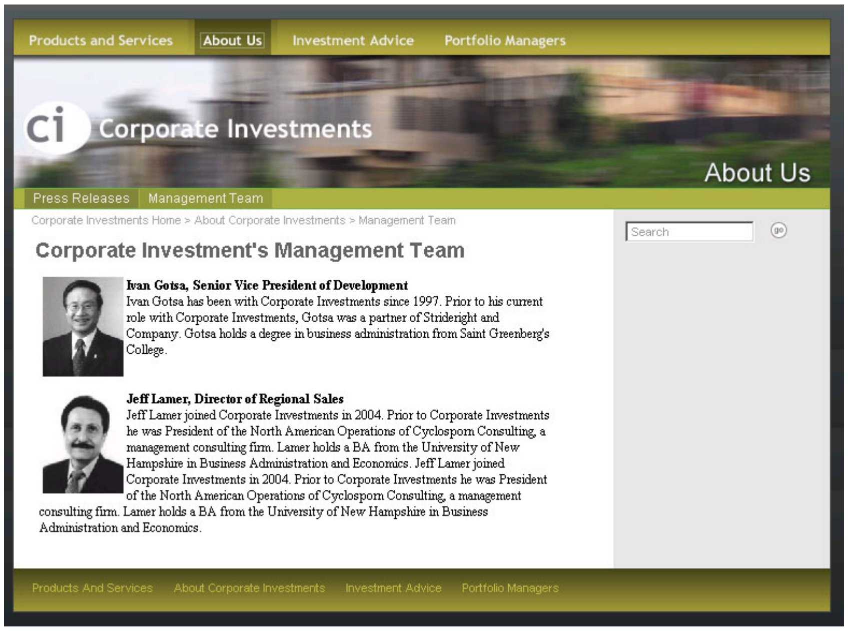


Figure 16: Components of a Global Template

This template will include the top, top_descendant, and bottom navigation Variants.

Bio Content Type

The Bio Content Type will be used to create Items about Corporate Investments' employees. This Type will gather data through a standard Content Editor form and will use Text Extraction to acquire the body text of the Item. An image of the employee will be directly associated with the Item as well.



Corporate Investments Home > About Corporate Investments > Management Team

Corporate Investment's Management Team

Ivan Gotsa, Senior Vice President of Development
Ivan Gotsa has been with Corporate Investments since 1997. Prior to his current role with Corporate Investments, Gotsa was a partner of Strideright and Company. Gotsa holds a degree in business administration from Saint Greenberg's College.

Jeff Lamer, Director of Regional Sales
Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclospom Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics. Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclospom Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics.

Products And Services About Corporate Investments Investment Advice Portfolio Managers

Figure 17: Generic Page with S - Bio Listing Snippet

Content Editor

The Content Editor for the Bio Content Type will contain the following fields:

____	Field/Label	Field Name	Control	Occur	Type(Format)
sys	System Title:	sys_title	sys_EditBox	Required	50
sys	Start Date:	sys_contentstartdate	sys_CalendarSimple	Required	none
sys	Expiration Date:	sys_contentexpirydate	sys_CalendarSimple	Optional	none
sh	Employee Name:	shared/displaytitle	sys_EditBox	Required	512
loc	Employee's Title:	employeetitle	sys_EditBox	Optional	50
loc	Bio Source Field:	bio	sys_File	Optional	max
loc	Bio Body:	biobody	sys_TextArea	Optional	max
sh	Image::	sharedimage/img1	sys_File	Required	max
sh	Image Filename:	sharedimage/img1_filenam e	sys_EditBox	Required	512
loc	Error Text:	errortext	sys_TextArea	Optional	max
sys	Locale:	sys_lang	sys_DropDownSingle	Optional	50
sys	Community:	sys_communityid	sys_DropDownSingle	Optional	none
sys	Workflow:	sys_workflowid	sys_DropDownSingle	Optional	none
sys		sys_currentview	sys_HiddenInput	Optional	none
sys		sharedimage/img1_ext	sys_HiddenInput	Optional	50
sh		sharedimage/img1_height	sys_HiddenInput	Optional	none
sh		sharedimage/img1_size	sys_HiddenInput	Optional	none
sh		sharedimage/img1_type	sys_HiddenInput	Optional	50
sh		sharedimage/img1_width	sys_HiddenInput	Optional	none

The following configurations will be made in the Advanced Tab:

- General - Allow fields to be searched and Enable Related Content
- Workflow - Standard Workflow and Allows Items to Enter Any Workflow
- Input Item Translation - sys_textExtraction as follows:

Source - PSXParam/bio

OutputParam - biobody

ErrorMessageParam - errortext

OutputFormat - TEXT

PDFConversion - SINGLE

In addition, add a conditional to this Exit as follows:

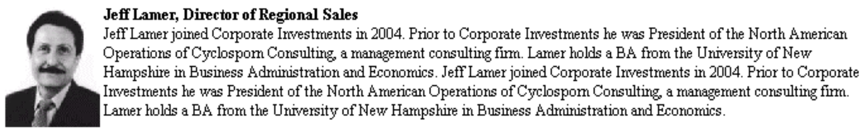
Conditional - PSXParam/bio IS NOT NULL

- Item Output Translation - N/A
- Item Validation - N/A

Variants

Bio will have only one Snippet, S - Bio Listing. This Snippet will be added as Allowed Content in the List Slot and used on Generic and similar pages as necessary.

S - Bio Listing



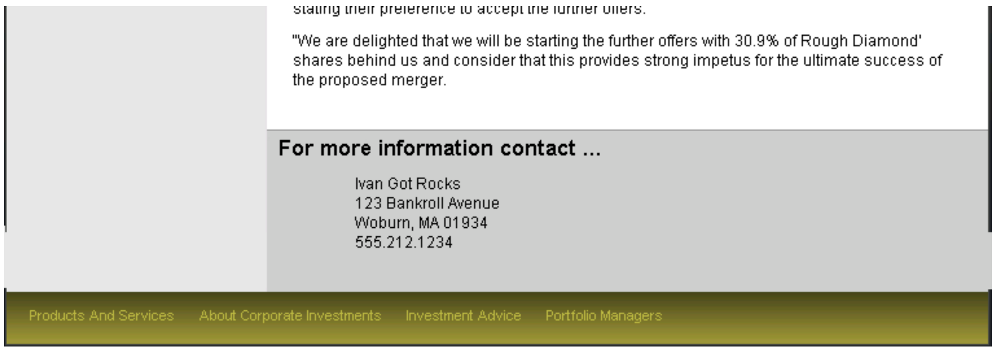
Jeff Lamer, Director of Regional Sales
Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclosporn Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics. Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclosporn Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics.

Figure 18: S - Bio Listing Snippet

The Snippet will display the Item's displaytitle, employeetitle, bodytext, and image.

Contacts Content Type

The Contacts Content Type requires only the augmentation of the existing Contacts Content Editor, Assembler, and Template to gather and display an office phone number.



stating their preference to accept the further offers.

"We are delighted that we will be starting the further offers with 30.9% of Rough Diamond' shares behind us and consider that this provides strong impetus for the ultimate success of the proposed merger.

For more information contact ...

Ivan Got Rocks
123 Bankroll Avenue
Woburn, MA 01934
555.212.1234

Figure 19: Contact Snippet on a Press Release

Contacts Content Editor

The field, officephone will be added to the Contacts Content Editor definition.

____	Field/Label	Field Name	Control	Occur	Type(Format)
sys	System Title:	sys_title	sys_EditBox	Required	50
sh	Title	shared/displaytitle	sys_EditBox	Required	512
sys	First Name:	firstname	sys_EditBox	Optional	50
sh	Middle Name:	middlename	sys_EditBox	Optional	50
loc	Last Name:	lastname	sys_EditBox	Optional	50

____	Field/Label	Field Name	Control	Occur	Type(Format)
loc	Dept:	dept	sys_EditBox	Optional	50
loc	Address1:	address1	sys_EditBox	Optional	50
sh	Address2:	address2	sys_EditBox	Optional	50
sh	City:	city	sys_EditBox	Optional	50
loc	State:	state	sys_EditBox	Optional	50
sys	Zipcode:	zipcode	sys_EditBox	Optional	50
sys	Country:	country	sys_EditBox	Optional	50
sys	Phone:	phone	sys_EditBox	Optional	50
sys	Cell Phone:	cellphone	sys_EditBox	Optional	50
	Office Phone:	officePhone	sys_EditBox	Required	50
	Fax:	fax	sys_EditBox	Optional	50
	Email:	email	sys_EditBox	Optional	50
	Start Date:	sys_contentstartdate	sys_CalendarSimple	Required	none
	Community:	sys_communityid	sys_DropDownSingle	Required	none
	Workflow:	sys_workflowid	sys_DropDownSingle	Required	none
	Locale:	sys_lang	sys_DropDownSingle	Required	50
		sys_currentview	sys_HiddenInput	Optional	none
sh	WebDAV Owner:	shared/webdavowner	sys_TextArea	Optional	255

The following configurations will be made in the Advanced Tab:

- General - Allow fields to be searched and Enable Related Content
- Workflow - Standard Workflow and Allows Items to Enter Any Workflow
- Input Item Translation -N/A
- Item Output Translation - N/A
- Item Validation - N/A

Variants

The Contacts Content Type will continue to maintain two Snippet Variants, S - Name and Email, and S - Name and Address. The S - Name and Address Snippet will have the field officephone added to the bottom of the address information.

S - Name and Email

Ivan Got Rocks | gotrocks@makemoney.com

Figure 20: Name and Email Snippet

The firstname, middlename, lastname, and email address fields are displayed.

S - Name and Address

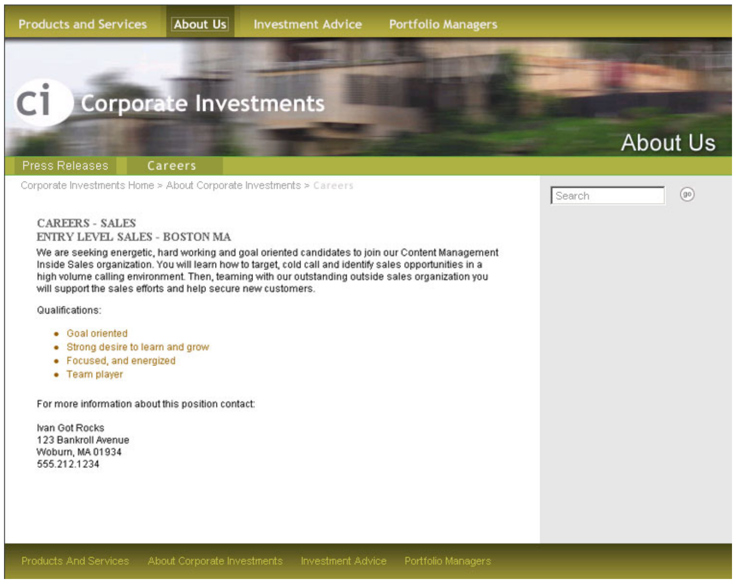
Ivan Got Rocks
123 Bankroll Avenue
Woburn, MA 01934
555.212.1234

Figure 21: Name and Address Snippet with Office Phone

The firstname, middlename, and lastname fields are displayed on the first line. Address1 is on the second line and address2 is on the third line if present. City, state, and Zipcode are on the next line. Officephone is on the last line.

Job Listing

The Job Listing Content Type will be used to create pages listing available positions at Corporate Investments. The Job Listing Type is very similar to the template used by Event Pages.



CAREERS - SALES
ENTRY LEVEL SALES - BOSTON MA
We are seeking energetic, hard working and goal oriented candidates to join our Content Management Inside Sales organization. You will learn how to target, cold call and identify sales opportunities in a high volume calling environment. Then, teaming with our outstanding outside sales organization you will support the sales efforts and help secure new customers.

Qualifications:

- Goal oriented
- Strong desire to learn and grow
- Focused, and energized
- Team player

For more information about this position contact:

Ivan Got Rocks
123 Bankroll Avenue
Woburn, MA 01934
555.212.1234

Figure 22: Job Listing Page

Content Editor

The Job Listing Content Editor will be defined as follows:

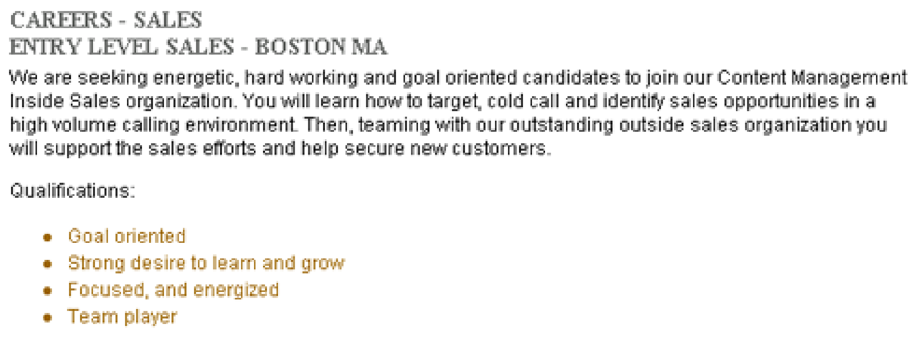
Source	Type	Name	Label	Control	Occur	Data Type	Format
System	field	sys_title	System Title:	sys_EditBox	Required	Text	50
System	field	sys_contentstartdate	Start Date:	sys_CalendarSimple	Optional	DateTime	None
System	field	sys_contentexpirydate	Expiration Date:	sys_CalendarSimple	Optional	DateTime	None
System	field	sys_reminderdate	Reminder Date:	sys_CalendarSimple	Optional	DateTime	None
System	field	sys_suffix		sys_EditBox	Optional	Text	50
System	field	sys_communityid		sys_DropDownSingle	Required	Integer	50
System	field	sys_workflowid		sys_DropDownSingle	Required	Integer	50
System	field	sys_lang		sys_DropDownSingle	Required	Text	50
System	field	sys_currentview		sys_HiddenInput	Optional	Text	50
Shared	field	shared/body	Body:	sys_EditLive	Optional	Text	max
Shared	field	shared/default_variantid	Default Variant:	sys_DropDownSingle	Optional	Text	50
Shared	field	shared/filename		sys_EditBox	Optional	Text	512
Shared	field	shared/displaytitle	Title:	sys_EditBox	Required	Text	512
Local	field	position	Position:	sys_EditBox	Optional	Text	50
Local	field	department	Department:	sys_DropDownSingle	Optional	Text	50
Local	field	location	Location:	sys_EditBox	Optional	Text	50
Local	field	summary	Summary:	sys_EditLive	Optional	Text	max

Items of this Type will enter the Standard Workflow. There will be two Variants of this type of Item: P - Job Listing and S - Position Location Link. The Page Variant will display the Position, Department, Location, a Summary, and the Body text. At the footer of the Item, a Contact Slot will allow for the inclusion of a Contact Item.

Variants

The Job listing Content Type will produce two Snippets, P - Job Listing and S - Position Location Link. The Snippet will display the position and where the position is being offered. This Snippet will link users to the details page, P - Job Listing.

P - Job Listing



CAREERS - SALES
ENTRY LEVEL SALES - BOSTON MA

We are seeking energetic, hard working and goal oriented candidates to join our Content Management Inside Sales organization. You will learn how to target, cold call and identify sales opportunities in a high volume calling environment. Then, teaming with our outstanding outside sales organization you will support the sales efforts and help secure new customers.

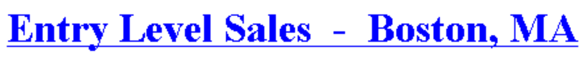
Qualifications:

- Goal oriented
- Strong desire to learn and grow
- Focused, and energized
- Team player

Figure 23: Job Listing Local Template Only

The Page will display the department field on the first line. The following line will contain the position and the location. The final field displayed is the summary then the body.

S - Position Location Link



Entry Level Sales - Boston, MA

Figure 24: Job Listing Position Location Link Snippet

The Snippet will display the position and location fields as a link to the Full Page.

Summary

From start to finish, the planning process must be thorough. Significant effort placed in the Modeling process allows those developing the necessary applications and components to gain the necessary interactions between all the pieces in the CMS puzzle. This process is likely to cause those planning the project to re-evaluate some existing processes that were once handled in one way and will now require a new approach due to the implementation of a Content Manager. These considerations often allow for a more streamlined, scalable, and manageable Site.

CHAPTER 4

Customizing FastForward Applications

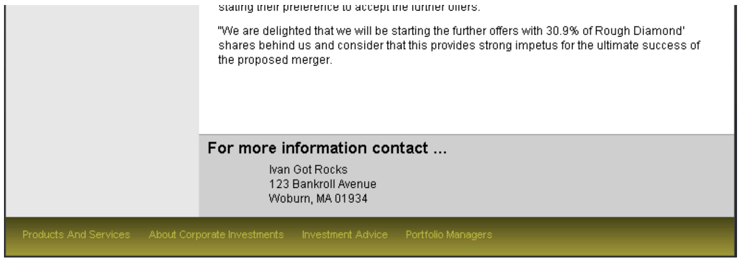
Often, it is possible to take existing applications and modify them slightly to meet a specific business need. This approach reduces the time required to deliver an application and likelihood of mistakes.

A few of the more common introductory customizing tasks are:

- Adding new fields to a Content Editor;
- Mapping a new field in a Content Assembler;
- Modifying Variant templates to display a new field.

Adding New Fields to a Content Editor

The Contacts Content Type contains many fields that might be necessary when displaying information about a contact when posting a Press Release or similar page.



stating their preference to accept the further offers.

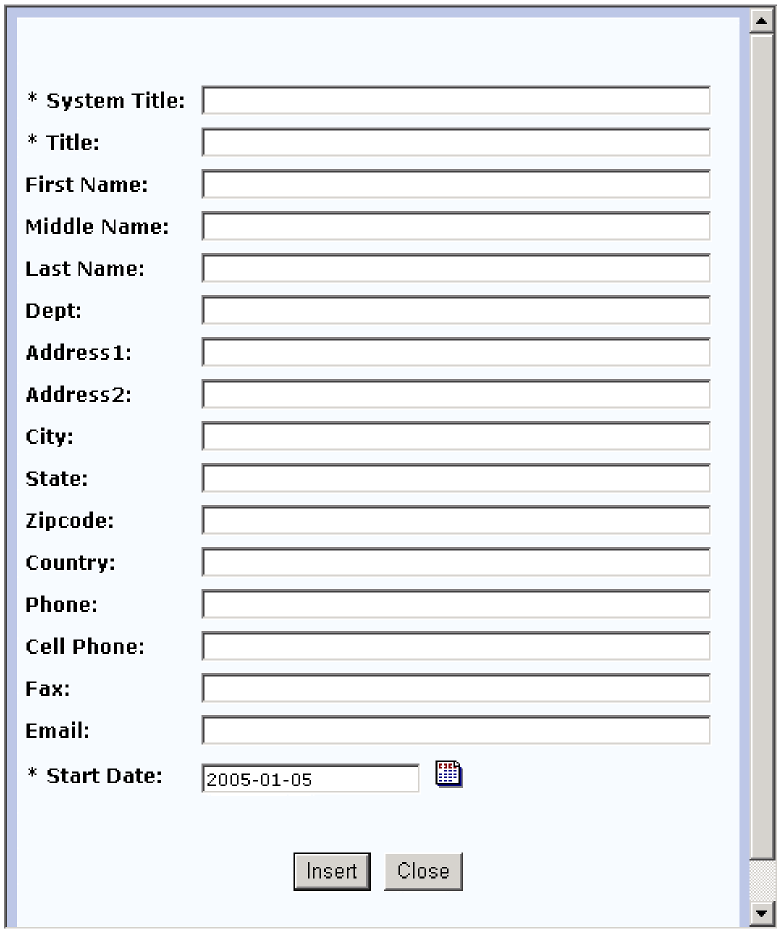
"We are delighted that we will be starting the further offers with 30.9% of Rough Diamond' shares behind us and consider that this provides strong impetus for the ultimate success of the proposed merger.

For more information contact ...

Ivan Got Rocks
123 Bankroll Avenue
Woburn, MA 01934

Figure 25: Contact Information at the footer of a Press Release

Our new business requires state that this contact must display the person's office phone number along with the mailing address. Unfortunately, our existing form for gathering contact information does not include this field.

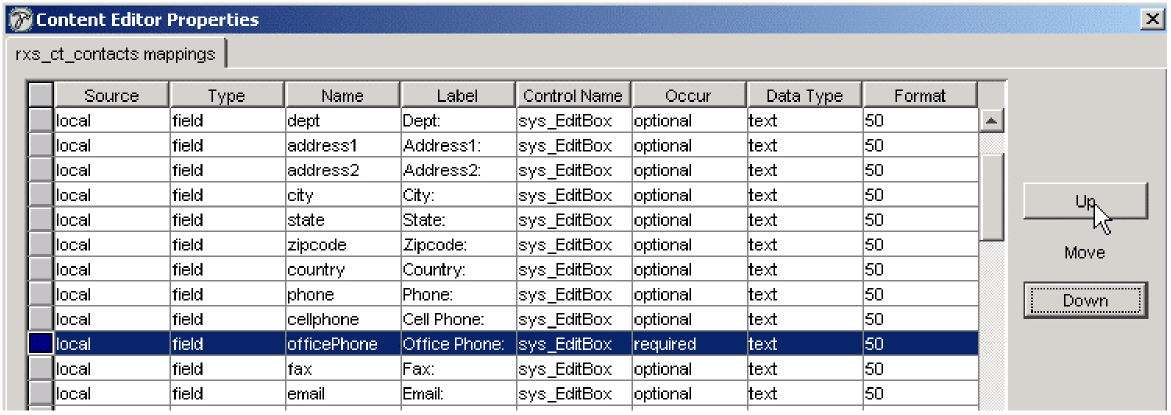


* System Title:	
* Title:	
First Name:	
Middle Name:	
Last Name:	
Dept:	
Address1:	
Address2:	
City:	
State:	
Zipcode:	
Country:	
Phone:	
Cell Phone:	
Fax:	
Email:	
* Start Date:	2005-01-05

Figure 26: Contact Content Editor

Our goal is to add this field to the Content Editor and store this data along with the Contact Item. Our requirements also state that this new field must be displayed on every Press Release. As a result, we will make the desk phone a required field.

1 Log into the Workbench.
2 Open the rxs_Contacts_ce application.
3 Open the Contacts Content Editor Resource.
4 Scroll to the bottom of the mappings list and add a new field based on the following values:
    - Source - local
    - Type - field
    - Name - officePhone
    - Label - Office Phone:
    - Control Name - sys_EditBox
    - Occur - required
    - Data Type - Text
    - Format - 50
5 Select the newly created mapping and move it up under the Cell Phone mapping.



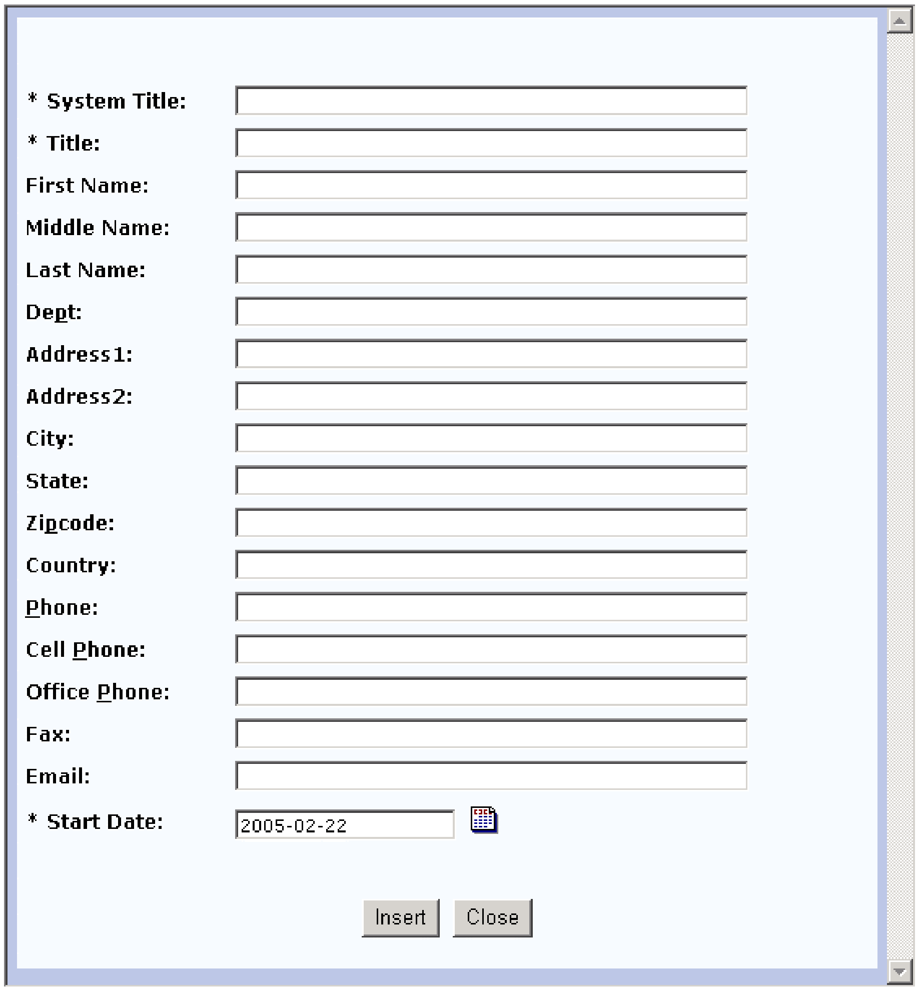
Source	Type	Name	Label	Control Name	Occur	Data Type	Format
local	field	dept	Dept:	sys_EditBox	optional	text	50
local	field	address1	Address1:	sys_EditBox	optional	text	50
local	field	address2	Address2:	sys_EditBox	optional	text	50
local	field	city	City:	sys_EditBox	optional	text	50
local	field	state	State:	sys_EditBox	optional	text	50
local	field	zipcode	Zipcode:	sys_EditBox	optional	text	50
local	field	country	Country:	sys_EditBox	optional	text	50
local	field	phone	Phone:	sys_EditBox	optional	text	50
local	field	cellphone	Cell Phone:	sys_EditBox	optional	text	50
local	field	officePhone	Office Phone:	sys_EditBox	required	text	50
local	field	fax	Fax:	sys_EditBox	optional	text	50
local	field	email	Email:	sys_EditBox	optional	text	50

Figure 27: Ordering Fields in a Content Editor

6 Select [OK] to close the Content Editor Properties dialog.
7 Save the rxs_Contacts_ce application.

When creating a new Item of the Contact Content Type, we are presented with a new field, "Office Phone:". Failure to enter a value in this field will restrict the entire Item's ability to be transitioned out of the Draft State.

Note: Though this new field is available, the value associated with the field is not yet being used on any final pages.



* System Title:	
* Title:	
First Name:	
Middle Name:	
Last Name:	
Dept:	
Address1:	
Address2:	
City:	
State:	
Zipcode:	
Country:	
Phone:	
Cell Phone:	
Office Phone:	
Fax:	
Email:	
* Start Date:	2005-02-22

Figure 28: Contacts Content Editor with Office Phone Field

Adding New Fields to an Assembly Resource

Once our new field, "Office Phone:" is available for Contact Items, we need to get the field and values associated with the field into the rendered XML for the Items. Our XML only returns the data currently mapped in the Assembly Resource.

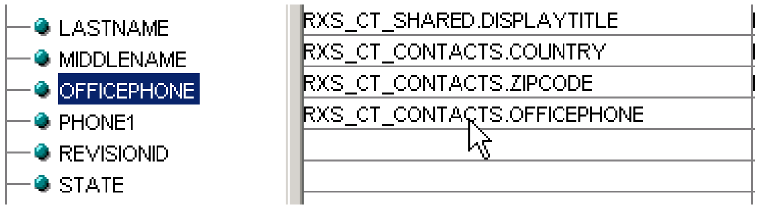
```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl"
href="/Rhythmyx/rxs_PressRelease_cas/p_CIPressRelease.xsl"?>
<whitebox ...>
...
   <shared>
      <displaytitle>Corporate Investments Announces Results of the
Initial Offers</displaytitle>
      <body>
            <p>Monday November 29, 11:00 AM EST ...</p>
      </body>
      <default_variantid>400</default_variantid>
      <filename/>
      <keywords/>
      <description/>
      <callout>
            <p>On October 18, 2004, Corporate Investments ...</p>
      </callout>
   </shared>
   <pr_summary>
         <p>On October 18, 2004, Corporate Investments ...</p>
   </pr_summary>
   <pr_type>CORPORATE</pr_type>
   <includehome/>
   <link2def>http://localhost:9992/Rhythmyx/rxs_PressRelease_cas/...</li
nk2def>
</whitebox>
```

Even though our Item now contains data for <officePhone />, it is not assembled. To add this data to our Item XML, we need to create a new mapping between the column in the table where the Content Editor stores the data and the DTD defining the schema. We will open the Assembly Resource for the Contacts Content Type and map our new field. The result will be a new node in our Item's generated XML for officePhone.

```
<officePhone>(555)212-1234</officePhone>
```

1. Log into the Workbench.
2. Open the rxs_Contacts_cas application.
3. Open the Contacts Assembly Resource Properties dialog
4. Open the Mapper.
5. Select the RXS_CT_CONTACTS table in the left hand Data Table column. Notice the addition of the column, OFFICEPHONE.

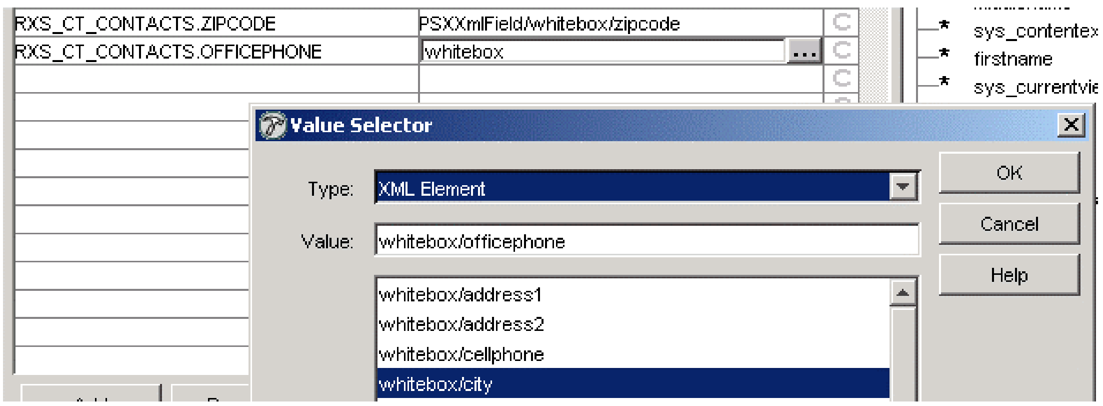
6 Select and drag the column OFFICEPHONE to the backend column of the mappings.



RXS_CT_SHARED.DISPLAYTITLE
RXS_CT_CONTACTS.COUNTRY
RXS_CT_CONTACTS.ZIPCODE
RXS_CT_CONTACTS.OFFICEPHONE

Figure 29: OFFICEPHONE column

7 Looking at the right hand column of DTD elements, we notice the officePhone element is not a part of our existing document structure. We can manually add the new element by selecting the empty XML field in the column next to our new OFFICEPHONE mapping and defining the new XML Element, whitebox/officephone.



RXS_CT_CONTACTS.ZIPCODE	PSXXmlField/whitebox/zipcode
RXS_CT_CONTACTS.OFFICEPHONE	whitebox

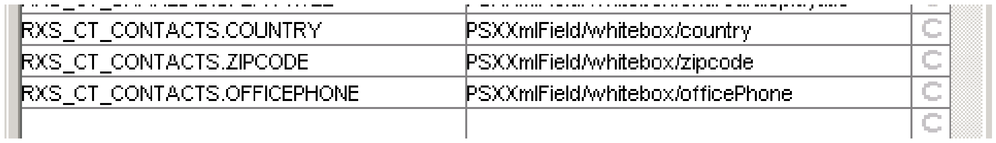
Type: XML Element

Value: whitebox/officephone

whitebox/address1
whitebox/address2
whitebox/cellphone
whitebox/city

Figure 30: Adding a New XML Element

8 Press [OK] to enter the new element.



RXS_CT_CONTACTS.COUNTRY	PSXXmlField/whitebox/country
RXS_CT_CONTACTS.ZIPCODE	PSXXmlField/whitebox/zipcode
RXS_CT_CONTACTS.OFFICEPHONE	PSXXmlField/whitebox/officePhone

Figure 31: New Field Mapping in the Contacts Assembler

9 Press [OK] to close the Mapper Properties dialog.

10 Close the Datapipe.

11 Save the rxs_Contacts_cas application.

At this point, the XML generated for Items of the Contact Type will now include the name value pair for the data collected for Office Phone. The data is not, though, displayed yet on our final page.

Modifying Variant Templates

Once data is collected by a Content Editor and mapped to our Item's rendered XML, we can use this new Content in Pages and Snippets. We will modify our existing Contacts Snippet Variant titled, S - Name And Address. Currently this Variant displays the contact's first and last name, address1, address2 if provided, city and state and zip.

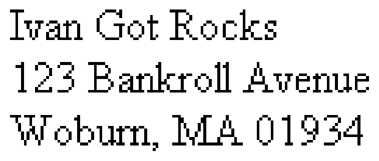

Ivan Got Rocks
123 Bankroll Avenue
Woburn, MA 01934

Figure 32: Name and Address Snippet

We will add to this output, the officephone.

1 Locate the original HTML for the S - Name and Address Variant in the Rhythmyx\rxs_Contacts_cas\src directory.

2 Open the template in an editor and find the Zipcode Field.

```
<span psxedit="zipcode" psxshow="psx-zipcode">Zipcode goes
here</span>
```

We will create a new field that will mimic this Zipcode entry.

3 Below the Zipcode field, add a new <br /> and <span> tag. Inside this <span> tag add some placeholder data to indicate the type of data that will be dynamically replaced when the template is used to render an Item.

```
<span>Office Phone goes here</span>
```

4 Add the necessary attributes for dynamic replacement based on the field mappings defined in the Contacts Assembly Resource. The element is "officephone."

```
<br />
<span psxedit="officephone" psxshow="psx-officephone">Office Phone
goes here</span>
```

5 Save and close the updated template.

6 Open the Workbench.

7 Open the rxs_Contacts_cas application.

8 Switch to the Files tab and navigate to the new Name and Address template.

9 From the Files tab in the Workbench, click and drag the template and drop it onto the S - Name and Address.xsl page in the application.

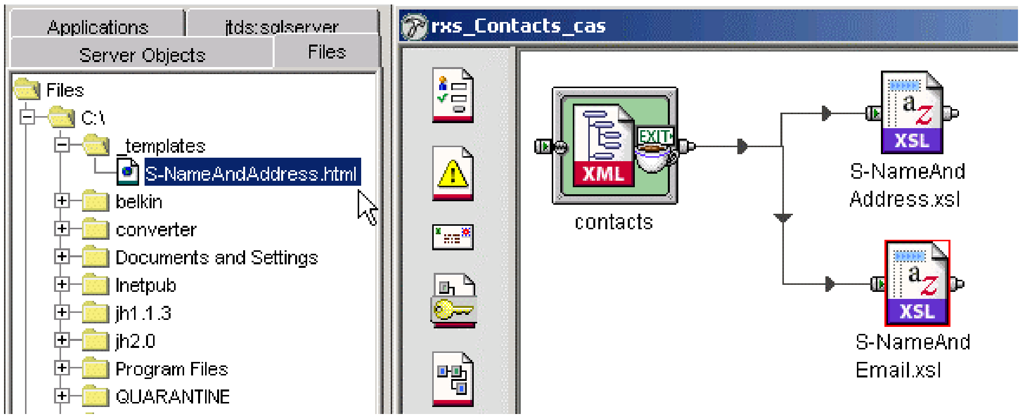
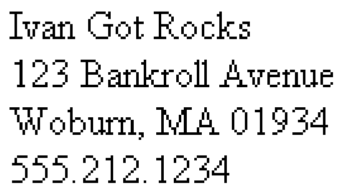


Figure 33: Updating a Page Template

10 Save the application.

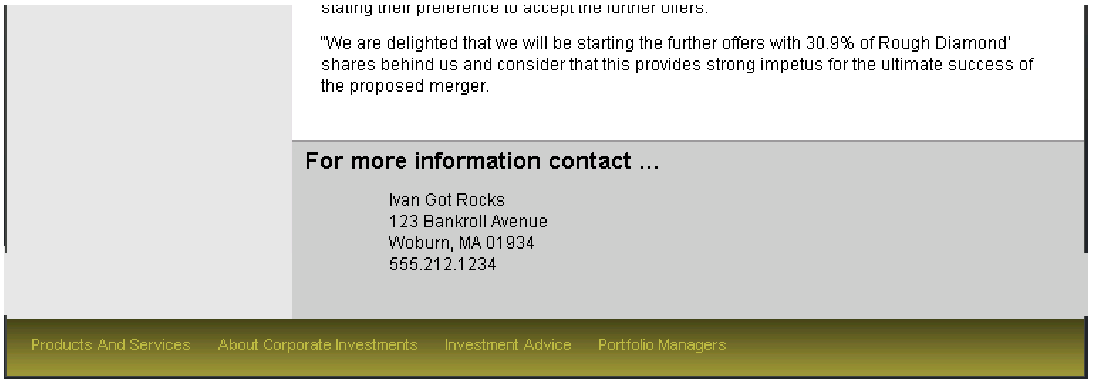
11 Upon preview of the S - Name and Address Snippet, we now see the phone Number created with the Item.



Ivan Got Rocks
123 Bankroll Avenue
Woburn, MA 01934
555.212.1234

Figure 34: Name and Address Snippet with Office Phone

12 When this Snippet is used on a Page, the text inherits the Page's formatting.



stating their preference to accept the further offers.

"We are delighted that we will be starting the further offers with 30.9% of Rough Diamond' shares behind us and consider that this provides strong impetus for the ultimate success of the proposed merger.

For more information contact ...

Ivan Got Rocks
123 Bankroll Avenue
Woburn, MA 01934
555.212.1234

Figure 35: Contact Snippet on a Press Release

CHAPTER 5

Extending FastForward Applications

In addition to making modifications to existing applications to meet the business needs for your CMS, it is also often necessary to create new applications. These new applications are generally designed to fulfill the need for a new Content Type. Implementing a new Content Type involves:

- Creating a New Content Editor
- Registering a New Content Type
- Creating a New Assembly Resource
- Creating a New Local Template
- Registering a New Variant

Implementing a New Content Type

We have identified the need to create a new Content Type for Job Listings. This Content Type will gather the following System, Shared, and Local data:

Source	Type	Name	Label	Control	Occur	Data Type	Format
System	field	sys_title	System Title:	sys_EditBox	Required	Text	50
System	field	sys_contentstartdate	Start Date:	sys_CalendarSimple	Optional	DateTime	None
System	field	sys_contentexpirydate	Expiration Date:	sys_CalendarSimple	Optional	DateTime	None
System	field	sys_reminderdate	Reminder Date:	sys_CalendarSimple	Optional	DateTime	None
System	field	sys_suffix		sys_EditBox	Optional	Text	50
System	field	sys_communityid		sys_DropDownSingle	Required	Integer	50
System	field	sys_workflowid		sys_DropDownSingle	Required	Integer	50
System	field	sys_lang		sys_DropDownSingle	Required	Text	50
System	field	sys_currentview		sys_HiddenInput	Optional	Text	50
Shared	field	shared/body	Body:	sys_EditLive	Optional	Text	max
Shared	field	shared/default_variantid	Default Variant:	sys_DropDownSingle	Optional	Text	50
Shared	field	shared/filename		sys_EditBox	Optional	Text	512
Shared	field	shared/displaytitle	Title:	sys_EditBox	Required	Text	512
Local	field	position	Position:	sys_EditBox	Optional	Text	50
Local	field	department	Department:	sys_DropDownSingle	Optional	Text	50
Local	field	location	Location:	sys_EditBox	Optional	Text	50
Local	field	summary	Summary:	sys_EditLive	Optional	Text	max

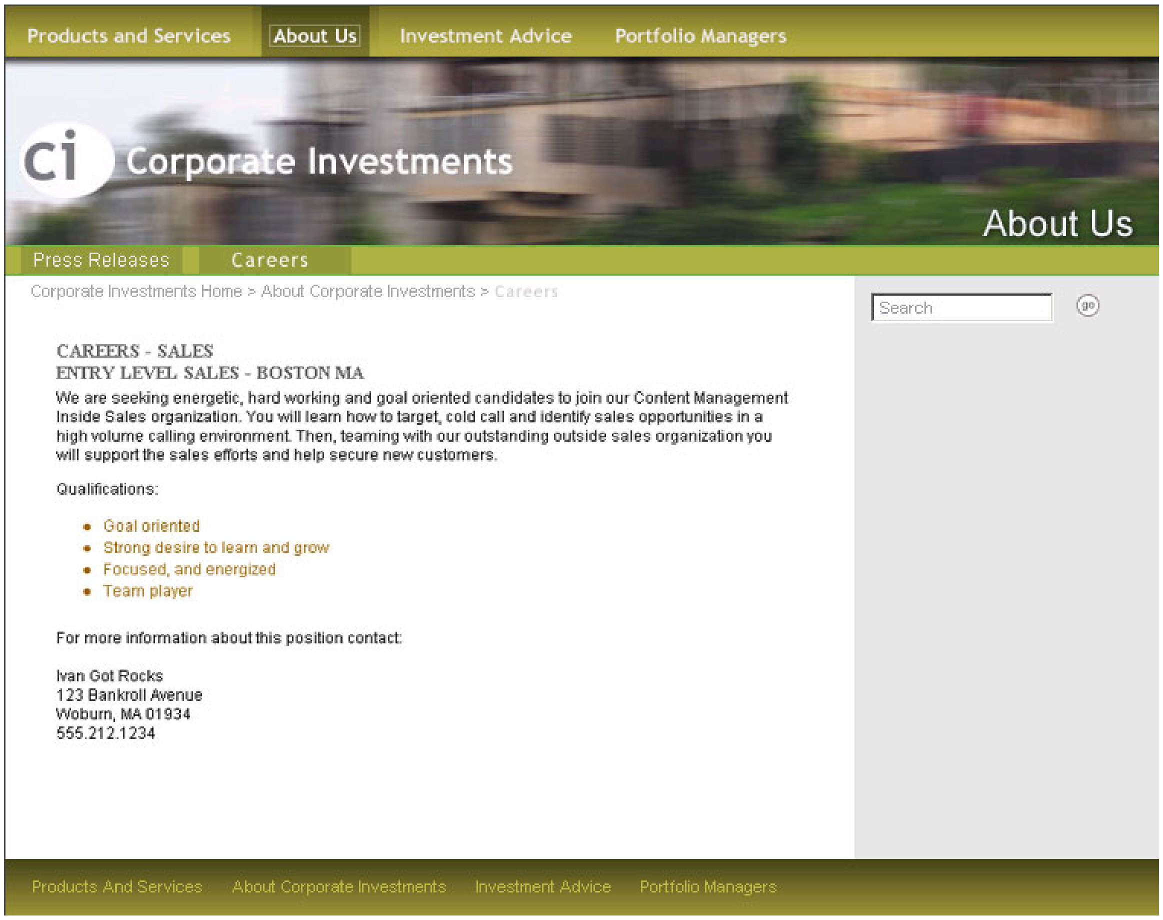
Items of this Type will enter the Standard Workflow. There will be two Variants of this type of Item:

- P - Job Listing

  The Page Variant will display the Position, Department, Location, a Summary, and the Body text. At the footer of the Item, a Contact Slot will allow for the inclusion of a Contact Item.

- S - Position Location Link.

  The Snippet Variant will be used as related content on an Index Page. This Snippet will display the Position, Department, and Location. This text will be linked to the Job Listing Page. Additionally, the Shared Variant, S - Title Link will also be implemented.



CAREERS - SALES
ENTRY LEVEL SALES - BOSTON MA

We are seeking energetic, hard working and goal oriented candidates to join our Content Management Inside Sales organization. You will learn how to target, cold call and identify sales opportunities in a high volume calling environment. Then, teaming with our outstanding outside sales organization you will support the sales efforts and help secure new customers.

Qualifications:

- Goal oriented
- Strong desire to learn and grow
- Focused, and energized
- Team player

For more information about this position contact:

Ivan Got Rocks
123 Bankroll Avenue
Woburn, MA 01934
555.212.1234

Figure 36: Job Listing Page

Creating a New Content Editor

Using the information provided in the specification for the Job Listing Content Type, we will begin by creating the new Content Editor.

Insert a new Content Editor Resource

1. Open the Workbench.
2. Create a new application (Application > New).
3. Insert a new Content Editor Resource (Insert > Content Editor Resource).
4. Select the sys_Default.xml template and press [OK].
5. Right click the Resource and select Resource Request Properties.

6 Change the Request Name from Default to JobListings.

7 Click [OK] to close the Request Properties dialog.

Add System, Shared, and Local fields to the Editor Resource

1 Right click the JobListings Content Editor Resource and select Properties.

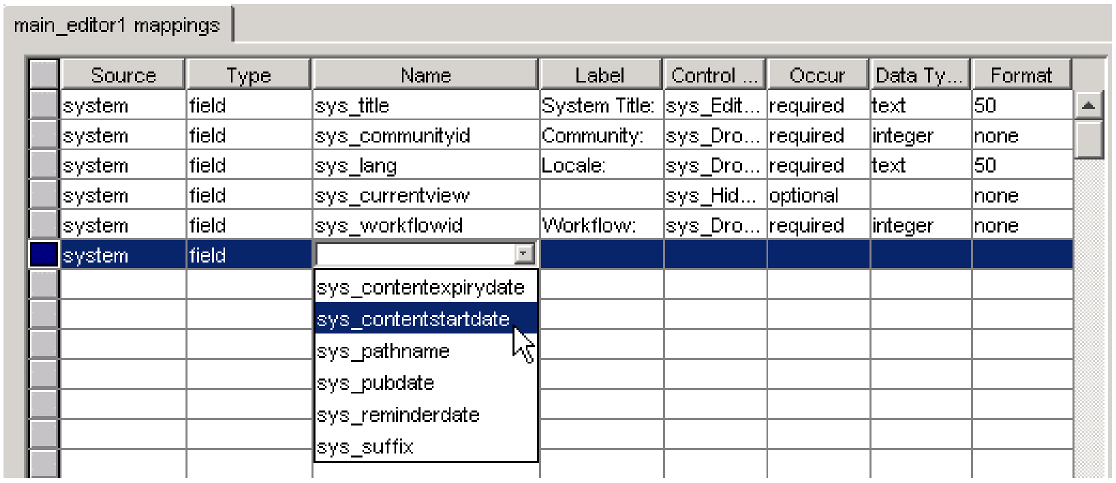
2 The Content Editor Properties dialog is presented with five System fields pre-defined:

- sys_title;
- sis_communityid;
- sys_lang;
- sys_currentview;
- sys_workflowid.

3 These fields were defined as part of the sys_Default.xml template. Begin by adding four of the remaining six System Fields:

- sys_contentstartdate;
- sys_contentexpirydate;
- sys_reminderdate;
- sys_suffix.

4 Double clicking the right hand side of each column for Source, Type, and Name will expose drop-down lists.

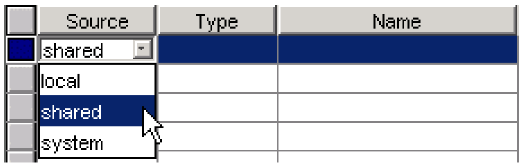


Source	Type	Name	Label	Control ...	Occur	Data Ty...	Format
system	field	sys_title	System Title:	sys_Edit...	required	text	50
system	field	sys_communityid	Community:	sys_Dro...	required	integer	none
system	field	sys_lang	Locale:	sys_Dro...	required	text	50
system	field	sys_currentview		sys_Hid...	optional		none
system	field	sys_workflowid	Workflow:	sys_Dro...	required	integer	none
system	field						

Figure 37: Adding System Fields to the Job Listings Content Editor

5 Once the Name is selected, the dialog will pre-fill each remaining field with guessed values. Modify these values as necessary to match the values provided in the specification.

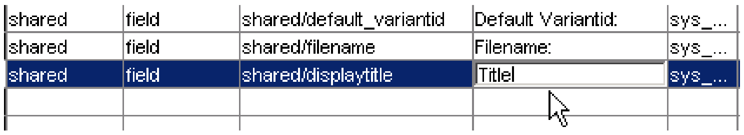
6 Continue by adding the four Shared Fields. The process is similar to adding the System fields but for the source field, choose Shared.



Source	Type	Name
shared		
local		
shared		
system		

Figure 38: Adding a Shared Field

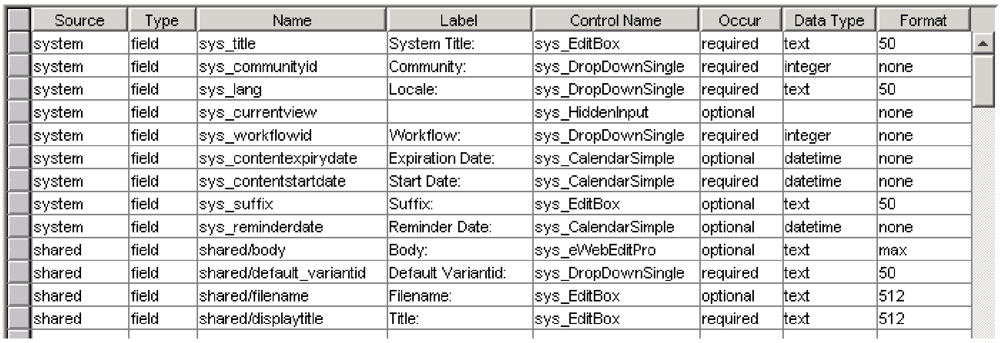
Note that you can edit inline or by selecting the entire row and pressing the [Edit] button.



shared	field	shared/default_variantid	Default Variantid:	sys_...
shared	field	shared/filename	Filename:	sys_...
shared	field	shared/displaytitle	Titlel	sys_...

Figure 39: Editing a Content Editor Label

Both approaches produce similar results.



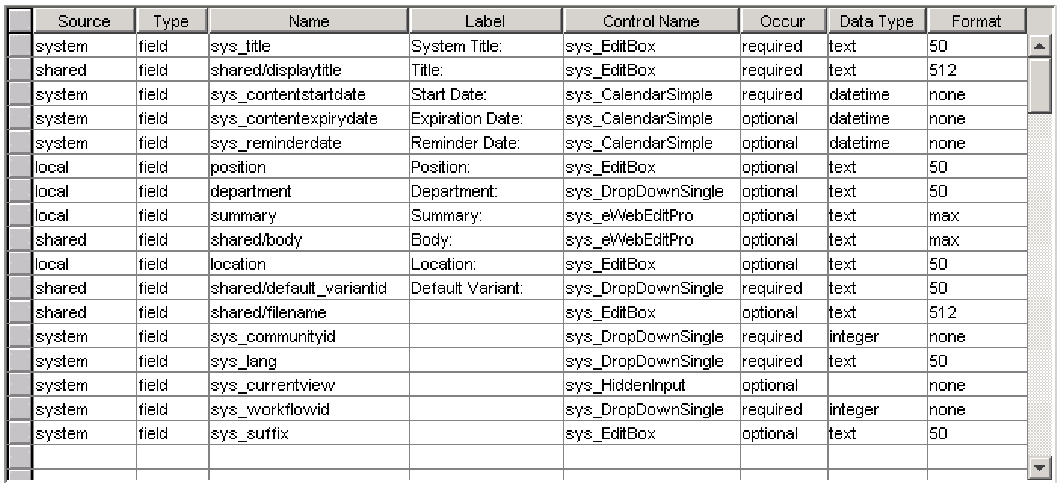
Source	Type	Name	Label	Control Name	Occur	Data Type	Format
system	field	sys_title	System Title:	sys_EditBox	required	text	50
system	field	sys_communityid	Community:	sys_DropDownSingle	required	integer	none
system	field	sys_lang	Locale:	sys_DropDownSingle	required	text	50
system	field	sys_currentview		sys_HiddenInput	optional		none
system	field	sys_workflowid	Workflow:	sys_DropDownSingle	required	integer	none
system	field	sys_contentexpirydate	Expiration Date:	sys_CalendarSimple	optional	datetime	none
system	field	sys_contentstartdate	Start Date:	sys_CalendarSimple	required	datetime	none
system	field	sys_suffix	Suffix:	sys_EditBox	optional	text	50
system	field	sys_reminderdate	Reminder Date:	sys_CalendarSimple	optional	datetime	none
shared	field	shared/body	Body:	sys_eWebEditPro	optional	text	max
shared	field	shared/default_variantid	Default Variantid:	sys_DropDownSingle	required	text	50
shared	field	shared/filename	Filename:	sys_EditBox	optional	text	512
shared	field	shared/displaytitle	Title:	sys_EditBox	required	text	512

Figure 40: Shared and System Fields

7 Finally, enter the local fields. Rhythmyx does not pre-populate the name field for local fields. Select Local as the Value for Source and manually enter the necessary Name, Label, Control Name, Occur, Data Type, and Format values as specified for the following four local fields:

- position;
- department;
- location;
- summary.

Once complete, arrange the fields to allow content contributors a logical order to enter data. Move the blank fields to the bottom of the editor to be sure blank spaces do not disrupt the look and feel of the final Content Editor.

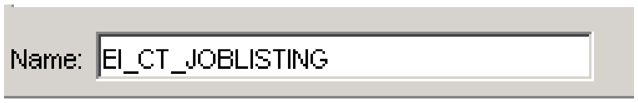


Source	Type	Name	Label	Control Name	Occur	Data Type	Format
system	field	sys_title	System Title:	sys_EditBox	required	text	50
shared	field	shared/displaytitle	Title:	sys_EditBox	required	text	512
system	field	sys_contentstartdate	Start Date:	sys_CalendarSimple	required	datetime	none
system	field	sys_contentexpirydate	Expiration Date:	sys_CalendarSimple	optional	datetime	none
system	field	sys_reminderdate	Reminder Date:	sys_CalendarSimple	optional	datetime	none
local	field	position	Position:	sys_EditBox	optional	text	50
local	field	department	Department:	sys_DropDownSingle	optional	text	50
local	field	summary	Summary:	sys_eWebEditPro	optional	text	max
shared	field	shared/body	Body:	sys_eWebEditPro	optional	text	max
local	field	location	Location:	sys_EditBox	optional	text	50
shared	field	shared/default_variantid	Default Variant:	sys_DropDownSingle	required	text	50
shared	field	shared/filename		sys_EditBox	optional	text	512
system	field	sys_communityid		sys_DropDownSingle	required	integer	none
system	field	sys_lang		sys_DropDownSingle	required	text	50
system	field	sys_currentview		sys_HiddenInput	optional		none
system	field	sys_workflowid		sys_DropDownSingle	required	integer	none
system	field	sys_suffix		sys_EditBox	optional	text	50

Figure 41: Job Listing Content Editor Fields Completed

Name the Job Listing Local Table

1 The Local Field data gathered by this Editor must be stored in a new table. Replace the value main_editor1 with the name EI_CT_JOBLISTING where EI is the name of our Community, CT describes that this table will hold data for a Content Type, and JOBLISTING names the Type.

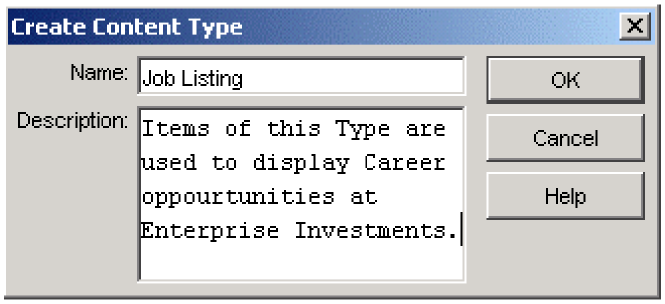


Name: EI_CT_JOBLISTING

Figure 42: Job Listing Table Defined for Content Editor

Register the New Content Type

1 Click the [New] button to the right of the Content Type drop-down. Enter the name, Job Listing and a description in the dialog.



Create Content Type

Name: Job Listing

Description: Items of this Type are used to display Career oppourtunities at Enterprise Investments.

Figure 43: Job Listing Content Type Registration

2 Press [OK] to enter the new name.

Defining Additional Properties to the Content Editor

1 Click the [Advanced] button on the Content Editor Properties Dialog.

2 In the General tab, check the Support Related Content check box. Checking this box allows content contributors to access Active Assembly dialogs to edit the Job Posting Content Item in a WYSIWYG interface.

3 In the Workflow tab, select the Standard Workflow.

4 Click [OK] to close the Content Editor Settings dialog.

5 Click [OK] to close the Content Editor Properties dialog.

Save and Start the Application

1 Save (Application > Save) the new Application.

2 Rhythmyx displays the Save Application dialog. Enter <u>ei_JobListing_ce</u> as the application name.

3 Press [OK] to save the new application.

When you save the application, Rhythmyx registers the new Content Type and creates the local table EI_CT_JOBLISTING.

4 Start (Application > Start) the new application.

Rhythmyx lists the new application in the application tab.

Configuring the Default Variant Control

The Default Variant control allows the content contributor the option of explicitly defining the Page Variant used when the Item is Published. Though this selected Variant can be overridden during the publish process, selecting the Default Variant gives the contributor to select be multiple outputs when available. In the event only a single output Variant can be selected, the drop-down will automatically be populated with this choice. This Control requires minor configuration to be operational.

1 Log into the Content Explorer and navigate to the System tab.

2 Click the Content Types [By Name] link.

3 Note the Content Id issued by the system when the Job Listing Content Type was registered.

4 Open the Workbench.

5 Open the ei_JobListing_ce application.

6 Open the Content Editor Properties dialog.

7 Select the Default Variant field and press [Edit].

Rhythmyx displays the Field Properties dialog.

8 Press the [...] next to the Control field.

9 Select the Choices tab.

Note: The Entry set source of Get choices from a Rhythmyx Application is selected.

10 Press the [...] next to the URL field.

Rhythmyx displays the Create Choice Lookup Request dialog.

11 The Application Name, rxs_Support_ce is pre-populated along with the Resource Name, rxs_PageVariantLookup. Enter the following support parameters to be fed to this application:

- ParamName - sys_contenttypeid
- Value - [The Content ID of the registered Job Listing Content Type identified in step 3]

12 Press [OK] as necessary to close all open dialogs for this Content Editor.

13 Save the application.

Note: This control will continue to be empty until a Page Variant is registered to this Content Type and defined as a possible Default.

This control requires an additional parameter that must be added manually.

1 Start an XML editor such as XML Spy, and open the file <Rhythmyxroot>\ObjectStore\ei_ct_joblisting.xml.

2 Find the following node:

```
<PSXDisplayMapping>
      <FieldRef>default_variantid</FieldRef>
         <PSXUISet>
            <PSXChoices sortOrder="user" type="internalLookup">
               <PSXUrlRequest>
                  <Href>../rxs_Support_ce/
                     rxs_PageVariantLookup</Href>
                     <Anchor/>
               </PSXUrlRequest>
            </PSXChoices>
         </PSXUISet>
</PSXDisplayMapping>
```

3 Add the following additional XML code after the <Href> tag:

```
<PSXParam name="sys_contenttypeid">
      <DataLocator>
         <PSXTextLiteral id="1">
            <text>999</text>
         </PSXTextLiteral>
      </DataLocator>
</PSXParam>
```

Where 999 is the Content Type ID of the Content Type of the Content Editor.

The updated node should resemble the following:

```
<PSXDisplayMapping>
      <FieldRef>default_variantid</FieldRef>
         <PSXUISet>
            <PSXChoices sortOrder="user" type="internalLookup">
               <PSXUrlRequest>
                  <Href>../rxs_Support_ce/
                     rxs_PageVariantLookup</Href>
```

```
                        <PSXParam name="sys_contenttypeid">
                           <DataLocator>
                              <PSXTextLiteral id="1">
                                    <text>999</text>
                              </PSXTextLiteral>
                           </DataLocator>
                        </PSXParam>
                     <Anchor/>
                  </PSXUrlRequest>
               </PSXChoices>
            </PSXUISet>
</PSXDisplayMapping>
```

Assigning the New Content Editor to a Community

Once a new Content Type is registered and the Content Editor created, the Type should be registered to a Community for testing. We will temporarily assign the Job Listing Type to the Corporate Investments Admin Community for this testing. Once the Assembler is completed and tested, we will assign this to additional Communities for their use as necessary.

1 Log into the Content Explorer.

2 Select the System tab and navigate to Communities [By Name].

3 Select the [Corporate Investments Admin] Community.

4 Select [Content Types].

5 Select [Add Content Type].

6 Select the Job Listing Content Type and press [Save].

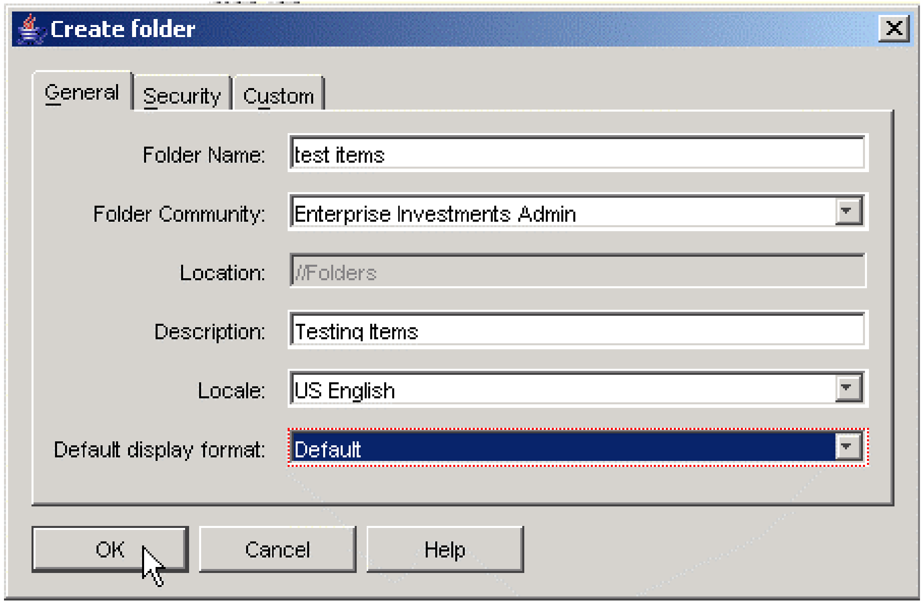
Testing the Job Listing Content Editor

Confirm the success of your efforts up to now by creating an Item of the Job Listing Type.

1 Log into the Content Explorer under the Corporate Investments Admin Community.

2 Right click the Folder icon, select [New Folder] and create a new Folder titled, "Test Folder."

3 Assign the folder to the Corporate Investments Admin Community only.



Folder Name: test items

Folder Community: Enterprise Investments Admin

Location: //Folders

Description: Testing Items

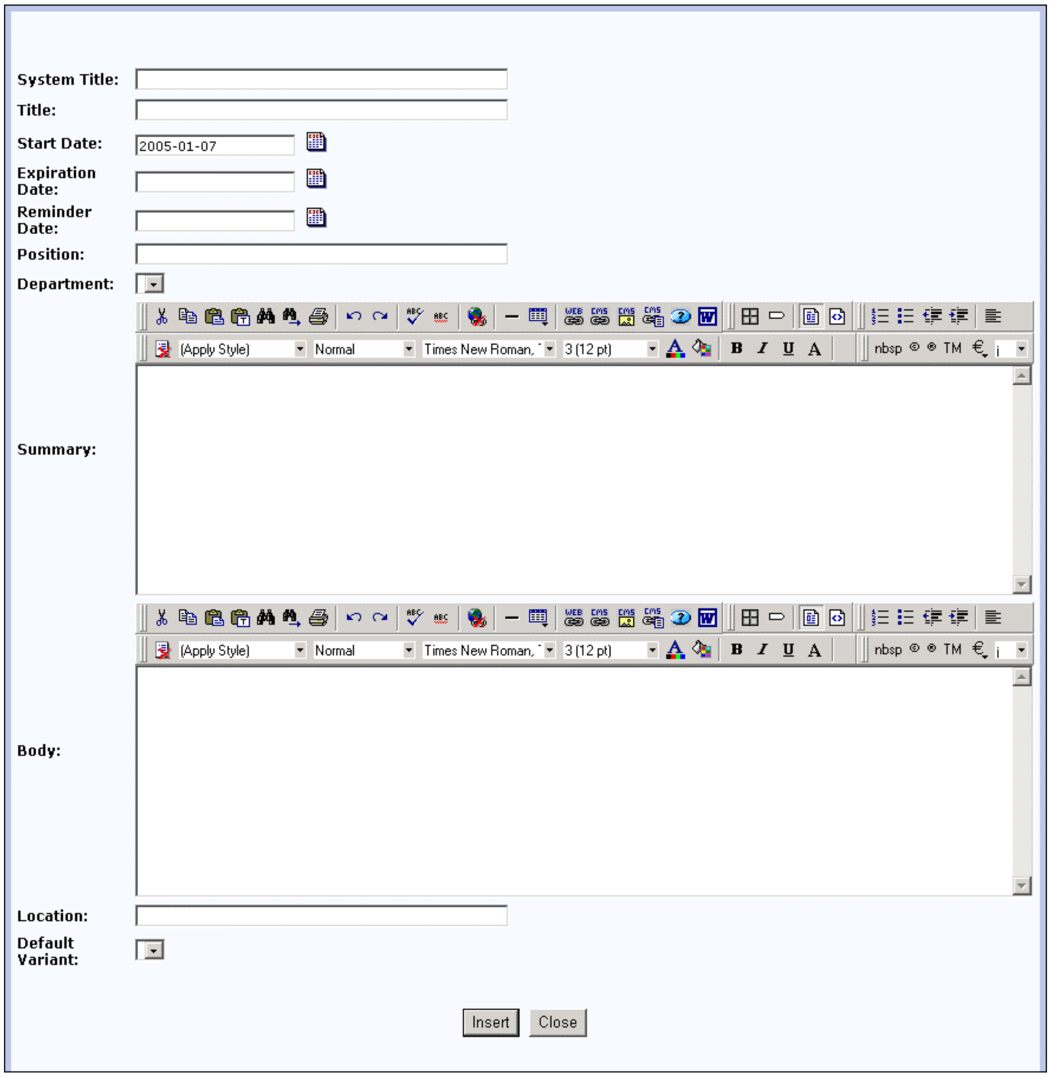
Locale: US English

Default display format: Default

Figure 44: Creating a New Folder

4 Press [OK] to create the new folder.

5 Right click the Test Folder and create a new Item of the Job Listing Type (New Item > Job Listing).



System Title:

Title:

Start Date: 2005-01-07

Expiration Date:

Reminder Date:

Position:

Department:

Summary:

Body:

Location:

Default Variant:

Figure 45: Job Listing Content Editor

One thing to notice about this new Content Editor is that the Department field has an unpopulated drop-down control. We will ignore this control for now and test the rest of the form's functionality by entering some dummy data into each field and pressing the [Insert] button. [Close] the form when all testing for functionality and visual layout is complete.

Creating a Department Keyword

Our Job Listing Content Editor has a field for Department. The possible departments a Content Contributor can associate Items of this Type with are:

- Human Resources
- Personal Finance
- Investing

- Operations
- Sales

We will populate this drop down with these values for use within this Editor by adding new Keywords. When creating a new Keyword, we create a display name, used value, and the order the arguments are displayed.

1. Log into the Content Explorer.
2. Navigate to the System tab.
3. Select Keywords [By Name].
4. Select [New Keyword].
5. Enter the following new Keyword:
   - Name - EI Departments
   - Description - Available departments for Job Listings at Corporate Investments.
6. Press [OK] to create the new Keyword.
7. Select the new Keyword [EI Departments] from the list of available Keywords.
8. Select [Add Choice].
9. Enter the following data for the new Keyword Value:
   - Choice Label - Human Resources
   - Description - Human Resources Department
   - Choice Value - Human Resources
   - Order -
10. Press [OK] to enter the new Keyword Value.
11. Repeat for the remaining departments: Personal Finance, Investing, Operations, Sales.

Adding Keywords to Content Editor Controls

Once a Keyword is registered, it is available for assignment to a Control. Assign the new Keyword to the Department field.

1. Log into the Workbench and open the ei_JobListing_ce application.
2. Open the JobListing Content Editor Properties dialog.
3. Locate the Department field, select it, and press [Edit].
4. Press the [...] next to the Control field.
5. Select the Choices tab presented in the Display Control Properties for Associations dialog.
6. In the Entry Set Source section, select Pre-defined set from RXLOOKUP.
7. Open the RXLOOKUP drop-down and select the newly created EI Departments Keyword.
8. Press [OK] to close the Associations dialog, [OK] to close the Properties dialog, then [OK] to close the Editor Properties dialog.
9. Save (Application > Save) the Content Editor application.

Final Testing of the Content Editor

To test the final Job Listing Content Editor:

1. Create a second test Job Listing Item in the Test Folder.
2. Confirm all desired values are available and accepted in each field.
3. Confirm each field label is displayed as expected and in their expected order.
4. Purge the first Job Listing Item to confirm the functionality of the Purge Resource.

Creating a New Content Assembler

Content assemblers define the assembly process for each type of output. The assembly process transforms the content items managed by the system into the published page elements and pages. The assembly process is recursive, allowing any output combinations.. The assembly process may produce both pages for standard publishing and partially assembled page elements for publishing to application servers and databases.

Content assemblers use the following components to define the assembly process:

Variants

A Variant is essentially a template, or set of templates, defining the layout, formatting, and other transformation rules (such as truncating or filtering text) applied to a content item. Typically, multiple Variants are available for any given item of content in the system, which allows the system to assemble content into different output pages automatically. Variants are defined using XSLT (Extensible Stylesheet Language Transformations). This XSL can be authored natively, or designers can use the XSpLit utility to generate the XSL automatically from existing HTML page designs. When a Variant is applied to an item of content, it produces either a fully formatted page or a partially formatted page element called a "Snippet" (see below).

Slots

When defining a Variant, Slots control the layout of aggregated content items. Each Slot defines a location on the page and the types of content that are allowed within that part of the page layout. When multiple content items are aggregated together, the Slots control where they appear in the final page.

Content Assemblers produce the following types of output:

Snippets

When a Variant is applied to an item of content, it may only generate part of the final output page. This partially assembled page element is called a "Snippet". Each Snippet may contain other aggregated Snippets, as well as formatted content. Typically, the assembly process continues this aggregation recursively, with Snippets including other Snippets until the final page is completely assembled and ready for publishing. However, any Snippet may also be published directly "as is" to applications that require only parts of the final output page, including personalization and dynamic delivery applications. The Content Assembler makes this Snippet available at any time through a URL. The publisher may use this URL to extract the Snippet from the system. Other assemblers may use this URL to include the Snippet in other phases of the assembly process.

Pages

A page is the final output of the content assembly process. A page is produced when a Variant is applied to an item of content. The Variant applies format and layout to that item and aggregates together all other required Snippets. As each Snippet is requested, the assembler processes the Variant for that Snippet, and so on recursively until all required Snippets are generated, and assembled into the right parts of the page. The Content Assembler makes this final page available at any time through a URL. The publisher uses this URL to extract the page from the system to save it out as a file.

The Assembly Process

Pages are assembled by applying a series of templates successively to the content data that makes up the page. For example, for a product page, first the marketing sidebar template may be applied, then the product page template, then the corporate template and so on, until all the content items are formatted and aggregated together to produce the final page layout.

Because content in Rhythmyx is XML-based, templates are defined with XSL style sheets. XSL is the open standard XML-based transformation and templating language. Rhythmyx, however, also includes the XSpLit utility, which automatically generates XSL templates from existing standard HTML pages. The Rhythmyx template developer may thus work in any standard HTML design tool, such as Dreamweaver, or even plain text editors, to develop the HTML template. Then, using XSpLit, the developer splits the HTML to produce the XSL templates. In addition, developers can also use native XSL. Native XSL offers the template developer greater power to transform and present content, such as to produce other file formats like WML or plain text output.

Since the assembly process involves recursively applying templates to many items of content; it is important to understand what templates are required at what levels to control and manage the assembly process. In general, each XSL style sheet contains the templates necessary to generate one level of template application. For example, one style sheet may be used to define the "side bar" template, another defines the "product page" template, and so on. Because each template only defines part of a page or a snippet, content items can be reused and changed across many pages at once.

At each level, the XSL style sheet defines the formatting of the content field's local to that level, as well as the placement of any included content elements produced by other levels of templating. Thus, each HTML page (or manually authored XSL style sheet) used to define a template for Rhythmyx includes:

- **Fixed Formatting** - Any fixed formatting markup, or static content needed to define the look and feel of the output produced by the template. Fixed formatting is generally HTML for most websites, but may include virtually any text depending on the type of output desired from this template.
- **Local Fields** - Field labels indicating where to place each field of the content item within the template.
- **Slots** - Slot markers define where snippets (sections of other formatted content items) will be placed within the current template. These snippets are generated by other layers of templates being applied "further down" in the assembly recursion.
- **Embedded XSL** - XSL is the "formatting API" for all Rhythmyx templates. It can be used natively or placed anywhere within the original HTML page defining the template. This XSL can produce more complicated forms of transformation, including both fixed formatting and slot behavior, such as filtering or truncating the text within content fields.

Recursive Content Roll-up

Rhythmyx assembles content into an output page by applying the fixed formatting to the local content of the content item, then recursively rolling up any snippets into the slots. The snippets are in turn assembled by applying the fixed formatting to the local content of the snippet, then recursively rolling up any child snippets into their slots, and so on. Each level of recursion includes only its own local content and templates and the URLs to each included snippet one level beneath. It needs no further information about deeper levels.

When Rhythmyx assembles a page:

1. A request for a page or snippet is passed to the assembler application via a URL.
2. The assembler requests the content item from the system. The system returns:
   - All local fields for that content item, as XML;
   - Assembly metadata, which includes a set of pointers (URLs) to all included snippets for this content item; Rhythmyx generates this list, which is also XML, automatically based on the related content defined for the item.
3. The assembler then applies the XSL stylesheet to the XML content item. The stylesheet:
   - Formats and transforms all local XML field content into some output text (generally HTML);
   - For each slot within the stylesheet, requests all the included snippets that go into that slot.
   - These requests trigger other assemblers to generate these snippets. Each of these requests thus begins again with step 1.
4. When the recursion is complete, Rhythmyx returns the output page or snippet.

Assembling Binary Outputs

A special type of assembly support also exists in Rhythmyx to support any type of binary outputs, such as image files (gif, jpg), downloadable files (zip, exe), and streaming media (avi, flash). In general, binary files do not require (or support) any kind of transformation. Thus, they can be extracted from the system directly for publishing as files without the need to create any formatting templates. The binary assembler simply defines the MIME type of the output binary file. However, most of the time, these binary files are also linked to the pages produced by the system. For example rendering a page with an image, requires both the page HTML file with an image tag, as well as the actual image file. Typically then, a standard template described above is also used to produce the pages or snippets that house the included binary content. Both the page and the binary file are then assembled and published to the web site, so that the end result is seamless to the browser user, regardless of what type of binary content is included on the web page.

Elements of the Assembly Process

The basic elements of the assembly process are Variants and Slots.

A Variant defines how to produce a type of output page or snippet from a content item. The variant defines transformation and formatting rules that are applied to a content item as well as the rules for aggregating related content items together. Each variant is associated with a specific content type, though each content type may have any number of associated variants. A variant is defined by the following properties:

- **Stylesheet**: The Stylesheet is an XSL document containing the templates Rhythmyx will use to transform and format the content item, as well as aggregate related content items, in order to produce this variant of output.
- **Assembly URL**: The Assembly URL defines the Rhythmyx application request that performs the actual assembly of the snippet.
- **Output Form**: The Output Form defines whether the output of the variant is a snippet or a page. This property determines how the output can be used. Snippets can be included in other snippets or pages, while pages cannot be included in other pages. Snippets must be well-formed because they will be parsed in later assembly processes. Pages should be well-formed, but because they will not be parsed further, they need not be.
- **Slots**: Slots specifies the slots that can be included in the variant. Only slots defined as eligible for the variant can be included when the variant is assembled.

A Slot defines an area on a page or snippet that will be filled by other snippets during assembly. Slots are defined primarily by name. Each slot also has a unique slot ID, which can be used along with the slot name to identify the Slot in any embedded XSL used to modify the Slot behavior.

Creating a Content Assembler Resource

Previously, we created a Content Editor Application to gather content for the Job Listing Content Type. The next step in the process is to build the application responsible for formatting this content for presentation on our web site. The first step of this process is to create an Assembler Resource. This resource will gather the data associated with this Item and any related Items and build from this an XML document. This resource basically queries the content repository, selects the appropriate fields from the selected records, and builds an XML document based on a DTD we define.

Building the Whitebox

1. Open the Workbench and open a new application (Application > New).
2. Open the ei_JobListing_ce application.
3. Right click the Content Editor Resource and choose [Copy Source XML].
4. Switch focus to the new empty application.
5. Right click in the application work area and choose [Paste as Assembler > Page]. A new Assembler Resource will be created with the title, page_assembler. This Assembler includes:
    - A DTD titled whitebox with the root element of the same name. This DTD contains elements for all System, Shared, and Local fields.
    - A Data Tank describing all tables used in the query and their appropriate joins.
    - All field mappings between table columns and DTD elements will be established.

- The sys_casAddAssemblerInfo, sys_tdTextToTree, and rxs_NavAutoSlot attached and mapped as necessary.
- A Result Pager sorting Items on CONTENTID.

In addition, the Resource will have a default Page attached to it.

6 Save the application (Application > Save) and give it the name ei_JobListing_cas.

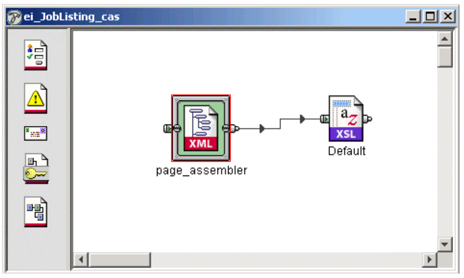


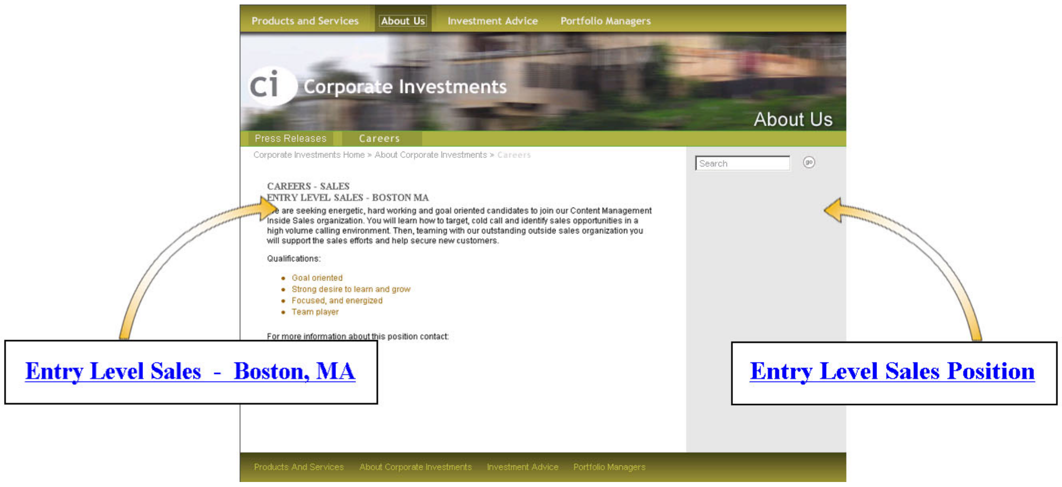
Figure 46: Job Listing Assembly Resource

7 Start the application (Application > Start).

- When you create the whitebox assembler, it does not automatically create child data mappings.
- All fields in a Page Assembler with a Text/max format must have the sys_tdTextToTree extension mapped manually.
- Both Snippet and Page Assemblers select Items based on CONTENTID and REVISIONID by default.

Creating Variant Templates

The final step in creating an Assembly Application is preparing templates to define how to display the managed content. As described earlier, we need to think about more than just how the Item will look as a page, but instead, how it will look as a Page Variant and how it will be represented on other pages. For example, you may want a link to this page from the home page, or an index page, or even other details pages. The Variant rendered on pages is defined by the Page Variant.



CAREERS - SALES
ENTRY LEVEL SALES - BOSTON MA
e are seeking energetic, hard working and goal oriented candidates to join our Content Management Inside Sales organization. You will learn how to target, cold call and identify sales opportunities in a high volume calling environment. Then, teaming with our outstanding outside sales organization you will support the sales efforts and help secure new customers.

Qualifications:

- Goal oriented
- Strong desire to learn and grow
- Focused, and energized
- Team player

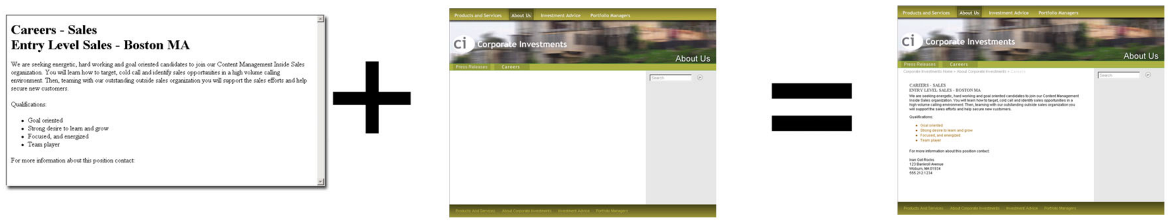
For more information about this position contact:

Entry Level Sales - Boston, MA

Entry Level Sales Position

Figure 47: Snippets Linking to Page

Our Assembly Resource produces the XML defining all of the data associated with the Job Listing Item. We will begin by building a template for the details page. The overall look and feel of the Job Listing page will conform to our current Corporate Investments Site. This will mainly be controlled by the Global Template wrapping our full page.



Careers - Sales
Entry Level Sales - Boston MA

We are seeking energetic, hard working and goal oriented candidates to join our Content Management Inside Sales organization. You will learn how to target, cold call and identify sales opportunities in a high volume calling environment. Then, teaming with our outstanding outside sales organization you will support the sales efforts and help secure new customers.

Qualifications:

- Goal oriented
- Strong desire to learn and grow
- Focused, and energized
- Team player

For more information about this position contact:

Figure 48: Local Plus Global Template Equals Page

As such, we will not be concerned about the Global Template itself (since it already exists). We will focus on the Local Template for our details page.

Our Job Listing page will need to display the following information:

- Department
- Position
- Location
- Summary
- Body

It will also include an Item of the Contact Type at the bottom of the page. These details pages will not include any content in the Sidebar or List Slot. As such, these Slots will not need to be associated with the page.

Looking through our existing Site, we notice that the Event Page has a similar structure to our new Job Listing page. We will make a copy of the p_EIEvent.html page and use that as our starting point. The following summarizes the creation of a new Variant:

- Prepare HTML (Your goal here is XHTML if possible);
- Add Context Variables, dynamic content markers, and Slots;
- Add Global Template marker (Local Templates only);
- Add XML output declaration (Snippets only);
- Split page in Assembler;
- Register new Variant with Content Type;
- Assign Page Selection Conditions to Split Page;
- Assign new Variant to Communities.

Creating Page Templates

Copying an Existing Page

Begin by copying of the existing p_EIEvent.HTML file which is stored in the Rhythmyx\rxs_Event_cas\src directory. Copy this file to a local folder and rename it, p_EIJobListing.html.

Preparing the Page for Editing

1 Open the p_EIJobListing.html file and begin by locating and removing the Sidebar and List Slots. Everything between and including the <!-- Start slot and <!-- End slot comments for both Slots can be deleted. Be sure not to modify the general structure of the page when removing these Slots.

We will focus the remaining edits for this page on the contents of the ArticleAbstractChar <div>. Continue by deleting everything in this <div> except for the <h1> and its contents, the span for the "body", and the Contact Slot.:

```
<div class="ArticleAbstractChar">
      <h1>
         <span psxedit="displaytitle" psxshow="psx-shared/
            displaytitle">Pick the right executor to handle your
            estate</span>
      </h1>
      <span psxedit="body" psxshow="psx-shared/
         body">Generic</span>
      <!-- start slot Contact -->
      <span class="Contact">
<!-- start snippet wrapper -->
      <span psxeditslot="yes" slotname="Contact">Contacts for
         press release</span>
<!-- end snippet wrapper -->
      </span>
<!-- end slot Contact-->
</div>
```

2 Next, change the content of the <h1> tag to display the Department instead of the displaytitle. Precede this dynamic data with the static text, Careers. In addition, put the Listing's position, and location in the line below the department:

```
      <h1>
         Careers - <span psxedit="department"
            psxshow="psx-department">Department for Listing</span>
         < br />
         <span psxedit="position" psxshow="psx-position">Position
            for Listing</span>
         <span>&nbsp;-&nbsp;</span>
         <span psxedit="location" psxshow="psx-location">Location
            for Listing</span>
      </h1>
```

3 The Listing's summary follows next. Place this markup between the <h1> tag and the body <span>:

```
<br />
   <span psxedit="summary" psxshow="psx-summary">Summary for
      Listing</span>
```

4 After the summary is the body text. This markup already exists in the page:

```
<br />
   <span psxedit="body" psxshow="psx-shared/body">Listing
      Body</span>
<br />
```

5 At the bottom of the page our Contacts Slot will be edited to say, "For more information about this position contact:" followed by the Contact Item.

```
<!-- start slot Contact -->
<span class="Contact">
   <span>For more information about this position contact:</span>
         <br />
<!-- start snippet wrapper -->
      <span psxeditslot="yes" slotname="Contact">Contacts for
         press release</span>
<!-- end snippet wrapper -->
   </span>
<!-- end slot Contact-->
```

6 The final page for the Job Listing Local Template will display all of our required content. The header of this document is already prepared to import our Global Template, corporate-global-template:

```
<!-- begin XSL -->
<xsl:output method="xml" omit-xml-declaration="yes"/>
<!-- end XSL -->
<html>
   <head psxglobaltemplate="*">
      <!-- begin XSL -->
      <xsl:call-template name="rxglobal_head" />
      <!-- end XSL -->
      <title>psx-shared/displaytitle</title>
   </head>
   <body>
      <!-- ===========  BREADCRUMBS START HERE =========== -->
      <!-- start slot nav_breadcrumb -->
      <div id="breadcrumbs">
         <span psxeditslot="no" slotname="nav_breadcrumb">nav
            breadcrumb slot</span>
      </div>
      <!-- end slot nav_breadcrumb -->
      <div id="ContentBox">
         <!-- ===========  CONTENT STARTS HERE =========== -->
         <div id="MainContent">
            <div class="ArticleAbstractChar">
               <h1>
         Careers - <span psxedit="department"
            psxshow="psx-department">Department for Listing</span>
               <br />
                  <span psxedit="position"
                     psxshow="psx-position">Position for
                     Listing</span>
                  <span>&nbsp;-&nbsp;</span>
                  <span psxedit="location"
                     psxshow="psx-location">Location for
                     Listing</span>
               </h1>
               <br />
               <span psxedit="summary" psxshow="psx-summary">
                  Summary for Listing</span>
               <br />
               <span psxedit="body" psxshow="psx-shared/body">
```

```
                        Listing Body</span>
                     <br />
                     <!-- ===========  CONTACT SLOT STARTS HERE
=========== -->
                     <!-- start slot Contact -->
                     <span class="Contact">
                        <span>For more information about this position
                           contact:</span>
                        <br />
                        <!-- start snippet wrapper -->
                        <span psxeditslot="yes" slotname="Contact">
                           Contacts for press release</span>
                        <!-- end snippet wrapper -->
                     </span>
                     <!-- end slot Contact-->
                  </div>
               </div>
            </div>
         </body>
      </html>
```

Note: The default Global Template is defined when setting up a Publishing Site. By adding, psxglobaltemplate="*" to the <html> tag, the page uses the Global Template dispatcher to select the appropriate template. If the Site Global Template is being overridden for an individual Variant, the following markup must replace the first block of XSL in the Page Template:

```
<!-- begin XSL -->
<xsl:import href="file:rx_resources/stylesheets/assemblers/
   rx_GlobalTemplates.xsl"/>
   <xsl:output method="xml" omit-xml-declaration="yes"/>
   <xsl:template match="/" priority="100">
      <!-- override inner template's root to invoke the outer
         template dispatcher -->
      <xsl:call-template name="psx-global-template-dispatcher"/>
</xsl:template>
<!-- end XSL -->
```

The name of the global template, corporate-global-template is used and the suffix is replaced by _root.

7 Save this page.

Note: Global Template precedence is as follows:

If specified, the Global Template explicitly defined in the Local Template is applied.

If a Global Template is not specified in the Local Template, the first Site Folder overriding the Site Global Template is used (Defined as a property of the individual Site Folder).

If no Site Folder override is found, the Site Global Template is used.

Applying the Page to the Job Listing Assembly Resource

1 Open the Workbench and open the ei_JobListing_cas application.

2 Select the Files tab in the Workbench navigation area and locate the recently created p_EIJobListing.html page. Select this page, drag, and drop the page onto the page_assembler resource. Select [Add Style Sheet] from the presented menu.

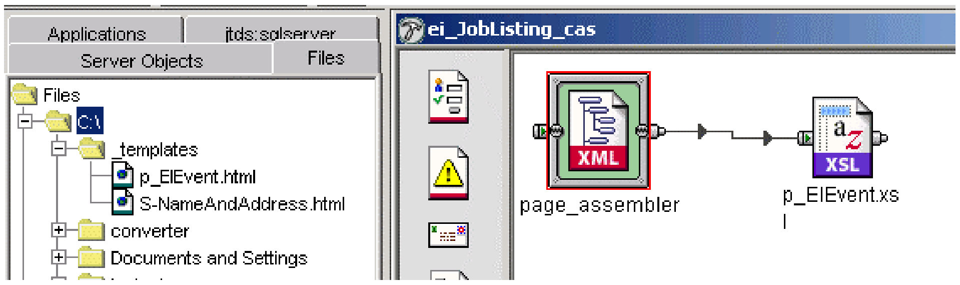


Figure 49: Job Listing Assembler and Page

3 Save (Application > Save) the application.

Registering the New Page Variant

1 With the ei_jobListing_cas open, right click the page_assembler resource and select [Request Properties].

2 At the bottom of the dialog press the Sample URL [Copy to Clipboard] button.

3 Log into the Content Explorer and navigate to the System Tab.

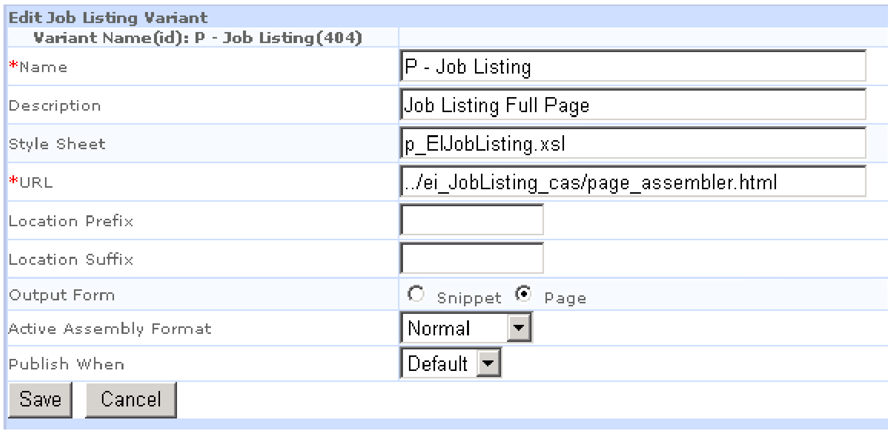
4 Under the Variants section, select [Job Listings].

5 Select [New Variant].

6 Enter the following data into the required fields:

- Name - P - Job Listing
- Description - Job Listing Full Page
- Style Sheet - p_EIJobListing.xsl
- URL - (paste the contents of the clipboard copied from the Assembly Resource here. Modify this URL to find the relative location of the application)
- Location Prefix -
- Location Suffix -

- Output Form - Page
- Active Assembly Format - Normal
- Publish When - Default



Edit Job Listing Variant
Variant Name(id): P - Job Listing(404)

*Name	P - Job Listing
Description	Job Listing Full Page
Style Sheet	p_EIJobListing.xsl
*URL	../ei_JobListing_cas/page_assembler.html
Location Prefix	
Location Suffix	
Output Form	Snippet [✓ Page]
Active Assembly Format	Normal
Publish When	Default

Figure 50: Job Listing Page Variant Registration

7 Select [Save] to register the new Variant.

Adding Slots to the New Page Variant

1 Re-Select the [Job Listing] Variant list.

2 Select the newly registered P - Job Listing Variant.

3 Select [Add Slot].

4 Select the Contacts Slot.

5 Repeat to add the following remaining Slots:

- nav_bottom
- nav_breadcrumb
- nav_left
- nav_preload
- nav_top

6 Select [Add Site].

7 Add the following Sites to the Variant:

- Corporate Investments
- Corporate Investments Mirror

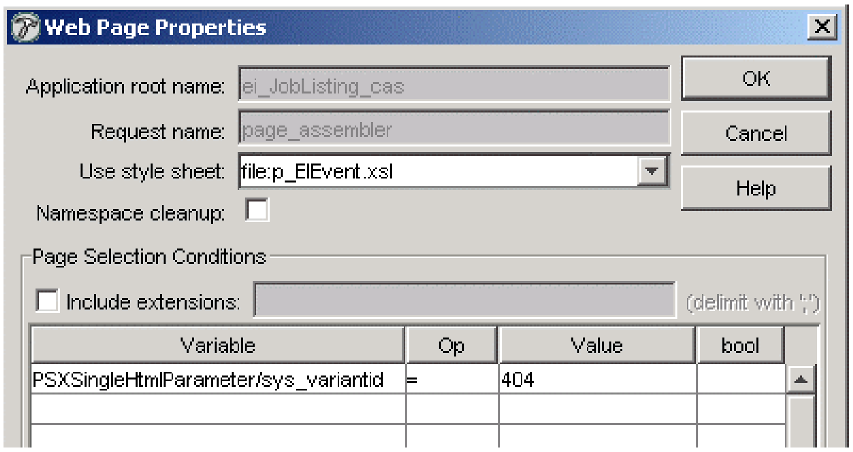
8 Make note of the Variant Id issued to this P - Job Listing Variant. This value is displayed at the top of the Variant's details page.

Finalizing the Job Listing Page Variant

1 Open the Workbench and open the ei_JobListing_cas application.

2 Right click the p_EIJoblisting.xsl page and select [Properties].

3 Add the Page Selection Condition for this Page for Variant ID (Variant ID of the P - Job Listing Variant recently registered):



Application root name: ei_JobListing_cas
Request name: page_assembler
Use style sheet: file:p_EIEvent.xsl
Namespace cleanup: ☐

Page Selection Conditions
☐ Include extensions: (delimit with ';')

Variable	Op	Value	bool
PSXSingleHtmlParameter/sys_variantid	=	404	

Figure 51: Adding Page Selection Conditions to a Page

4 Press [OK] to close the Web Page Properties dialog.

5 Save (Application > Save) the application.

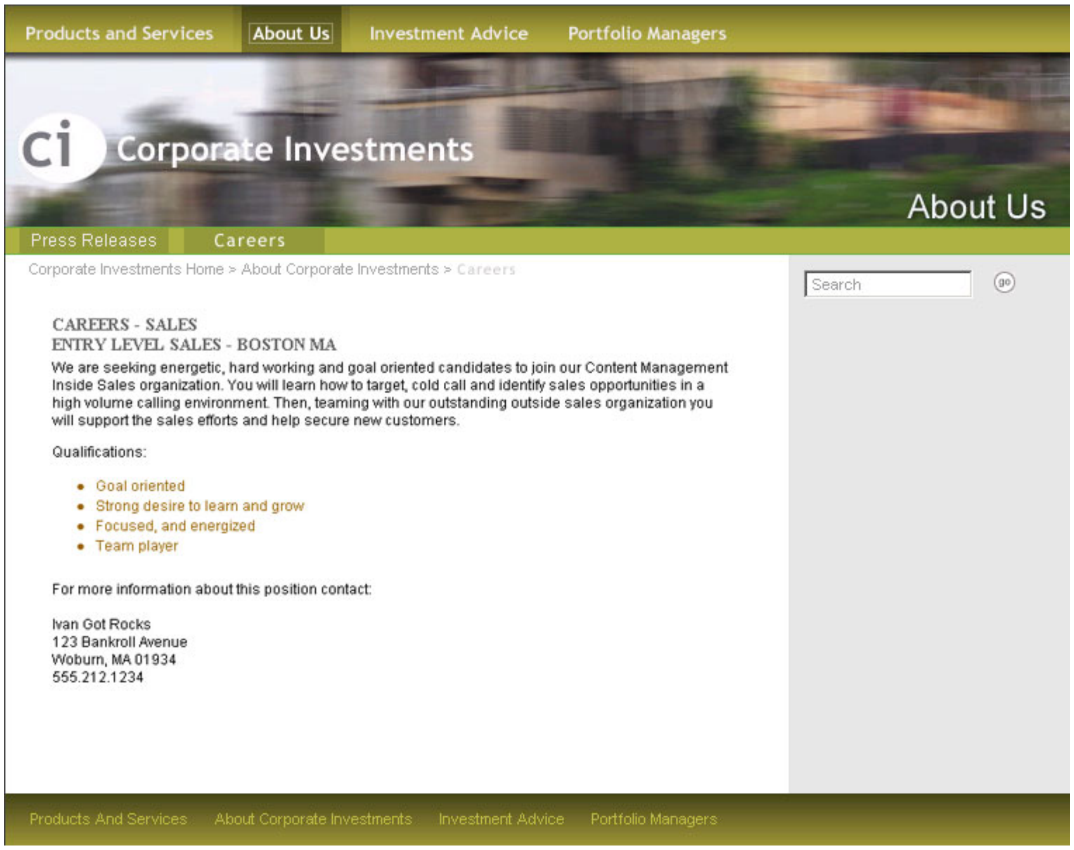
Testing the New Page Variant

1 Log into the Content Explorer and switch to the Corporate Investments Admin Community.

2 Locate the Test Folder Folder created recently for testing. This Folder should contain a test Item of the Job Listing Type. [Edit] the Item. The Default Variant drop-down should now contain the Variant P - Job Listing (When we registered this Variant we gave it the Publish When value Default).

3 Right click the Item in the Content Explorer and preview the Page Variant (Internet Preview > P - Job Listing).

4 The Item should be displayed with the Global Template as expected. Make modifications if the output does not conform to the specifications describe in the Development Plan and re-split the template accordingly.



CAREERS - SALES
ENTRY LEVEL SALES - BOSTON MA
We are seeking energetic, hard working and goal oriented candidates to join our Content Management Inside Sales organization. You will learn how to target, cold call and identify sales opportunities in a high volume calling environment. Then, teaming with our outstanding outside sales organization you will support the sales efforts and help secure new customers.

Qualifications:

- Goal oriented
- Strong desire to learn and grow
- Focused, and energized
- Team player

For more information about this position contact:

Ivan Got Rocks
123 Bankroll Avenue
Woburn, MA 01934
555.212.1234

Figure 52: Job Listing Page

Snippet Templates

Our Job Listing will show up as a short link on a Careers Index page on our site. The requirement for this page is to generate the position and location text as a hyperlink to the Page Variant. This Snippet will be simple.

Entry Level Sales - Boston, MA

Figure 53: Job Listing Position Location Link Snippet

We can take the bulk of this Snippet's markup from the page template. Best Practices tell us to create a separate Assembly Resource for our Snippets. Snippets are requested by the system more often than pages but often require less post document processing. Page Assemblers tend to incur more processing than Snippet Assemblers but are requested less often. As such, a separate Resource for Snippets will help improve overall performance during preview and publish of Items.

Create the Snippet Template

1 Locate the s_titlelink.html file (a copy can be found in the Rhythmyx\rxs_Shared_cas\src directory) and save a new copy of this file as s_EIJobListingPositionLink.html locally.

2 Edit the template to display the Job Listing position and location.

```
<!-- begin XSL -->
<xsl:output method="xml" omit-xml-declaration="yes" />
<!-- end XSL -->
<html>
   <head>
      <title>psx-shared/displaytitle</title>
      <link rel="stylesheet" psx-href="$rxs_css/rxs_styles.css"
         href="/web_resources/css/rxs_styles.css" type="text/
         css" />
   </head>
   <body>
      <span class="lead_snippet">
         <h1>
            <a href="psx-link" class="titlelink">
               <span psxedit="position" psxshow="psx-position">
                  Position for Listing</span>
               <span>&nbsp;-&nbsp;</span>
               <span psxedit="location" psxshow="psx-location">
                  Location for Listing</span>
            </a>
         </h1>
      </span>
   </body>
</html>
```

3 Save the updated template.

Note: Since we used an existing template, we did not need to add the <link> to the css file and the necessary markup for the <head> and the XML declaration at the top of the page.

Create the Snippet Assembly Resource

1 Open the Workbench and open both the ei_JobListing_cas and ei_JobListing_ce applications.

2 In the Content Editor application, right click the JobListing Resource and select [Copy Source XML].

3 Switch to the Assembly application, right click in the workspace, and select [Paste as Snippet Assembler].

4 Save (Application > Save) the updated application.

Apply the New Snippet Template to the Snippet Assembler

1 Open the Workbench and open the ei_JobListing_cas application.

2 Select the Files tab in the explorer column and locate the newly created s_JobListingPositionLink.html file. Select this page, drag, and drop the page onto the snippet_assembler resource. Select [Add Style Sheet] from the presented menu.

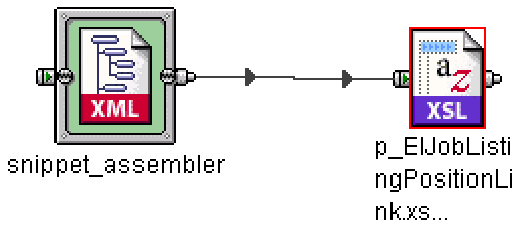


Figure 54: Snippet Assembler and Page for Job Listing Snippet

3 Save (Application > Save) the application.

Registering the New Snippet Variant

1 With the ei_jobListing_cas open, right click the snippet_assembler resource and select [Request Properties].

2 At the bottom of the dialog press the Sample URL [Copy to Clipboard] button.

3 Log into the Content Explorer and navigate to the System Tab.

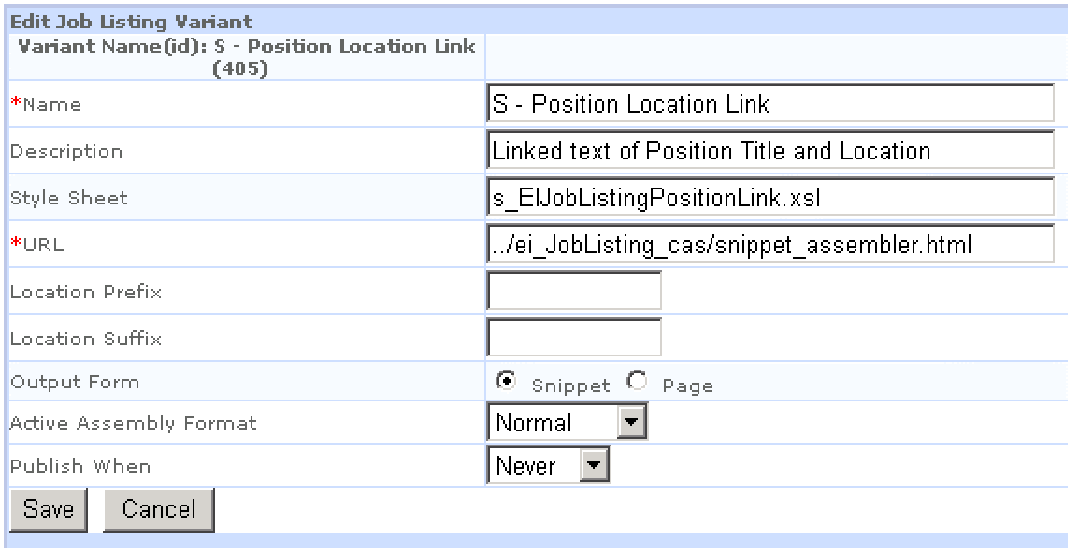
4 Under the Variants section, select [Job Listings].

5 Select [New Variant].

6 Enter the following data into the required fields:

- Name - S - Position Location Link
- Description - Job Listing Position and Location Link
- Style Sheet - s_EIJobListingPositionLink.xsl
- URL - (paste the contents of the clipboard copied from the Assembly Resource here. Modify this URL to find the relative location of the application)
- Location Prefix -
- Location Suffix -
- Output Form - Snippet

- Active Assembly Format - Normal
- Publish When - Never



Edit Job Listing Variant

Variant Name(id): S - Position Location Link (405)

*Name	S - Position Location Link
Description	Linked text of Position Title and Location
Style Sheet	s_EIJobListingPositionLink.xsl
*URL	../ei_JobListing_cas/snippet_assembler.html
Location Prefix	
Location Suffix	
Output Form	[✓ Snippet] Page
Active Assembly Format	Normal
Publish When	Never

Figure 55: Job Listing Snippet Registration

7 Select [Save] to register the new Variant.

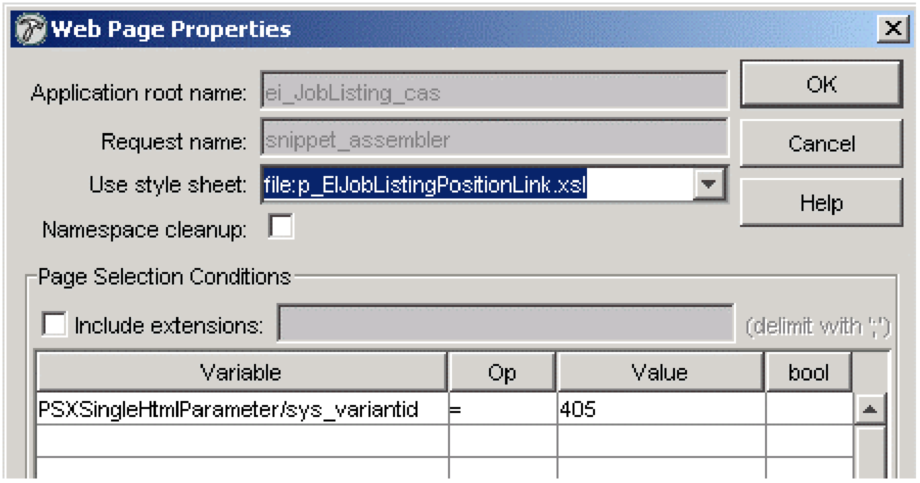
8 Make note of the Variant Id issued to this P - Job Listing Variant. This value is displayed at the top of the Variant's details page.

Finalizing the Job Listing Snippet Variant

1 Open the Workbench and open the ei_JobListing_cas application.

2 Right click the s_EIJoblistingPositionLink.xsl page and select [Properties].

3 Add the Page Selection Condition for this Page for Variant ID (Variant ID of the S - Position Location Link Variant recently registered):



Web Page Properties

Application root name: ei_JobListing_cas

Request name: snippet_assembler

Use style sheet: file:p_EIJobListingPositionLink.xsl

Namespace cleanup: ☐

Page Selection Conditions

☐ Include extensions: (delimit with ';')

Variable	Op	Value	bool
PSXSingleHtmlParameter/sys_variantid	=	405	

Figure 56: Page Selection Conditions for Job Listing Snippet

4 Press [OK] to close the Web Page Properties dialog.

5. Save (Application > Save) the application.

Testing the New Snippet Variant

1. Log into the Content Explorer and switch to the Corporate Investments Admin Community.
2. Locate the Test Folder Folder created recently for testing. This Folder should contain a test Item of the Job Listing Type.
3. Modify the Item if necessary and [Save] the changes.
4. Right click the Item in the Content Explorer and preview the Snippet Variant (Internet Preview > S - Position Location Link).
5. The Item should be displayed without the Global Template. In fact, is should appear quite plain. Make modifications as necessary to the template and re-split it accordingly.

Maintaining Global Templates

The FastForward implementation relies on the use of Global Templating to facilitate the reuse of page templates. Global Templating allows the developer to define a master template used to wrap Local Template items, ultimately normalizing redundant HTML. The locally templated items are designed with as little formatting as possible. When the Global template is applied to the locally templated item, the item takes on the look and feel as defined by this master template. This process allows developers to produce pages of many different types but process them through the single master template.

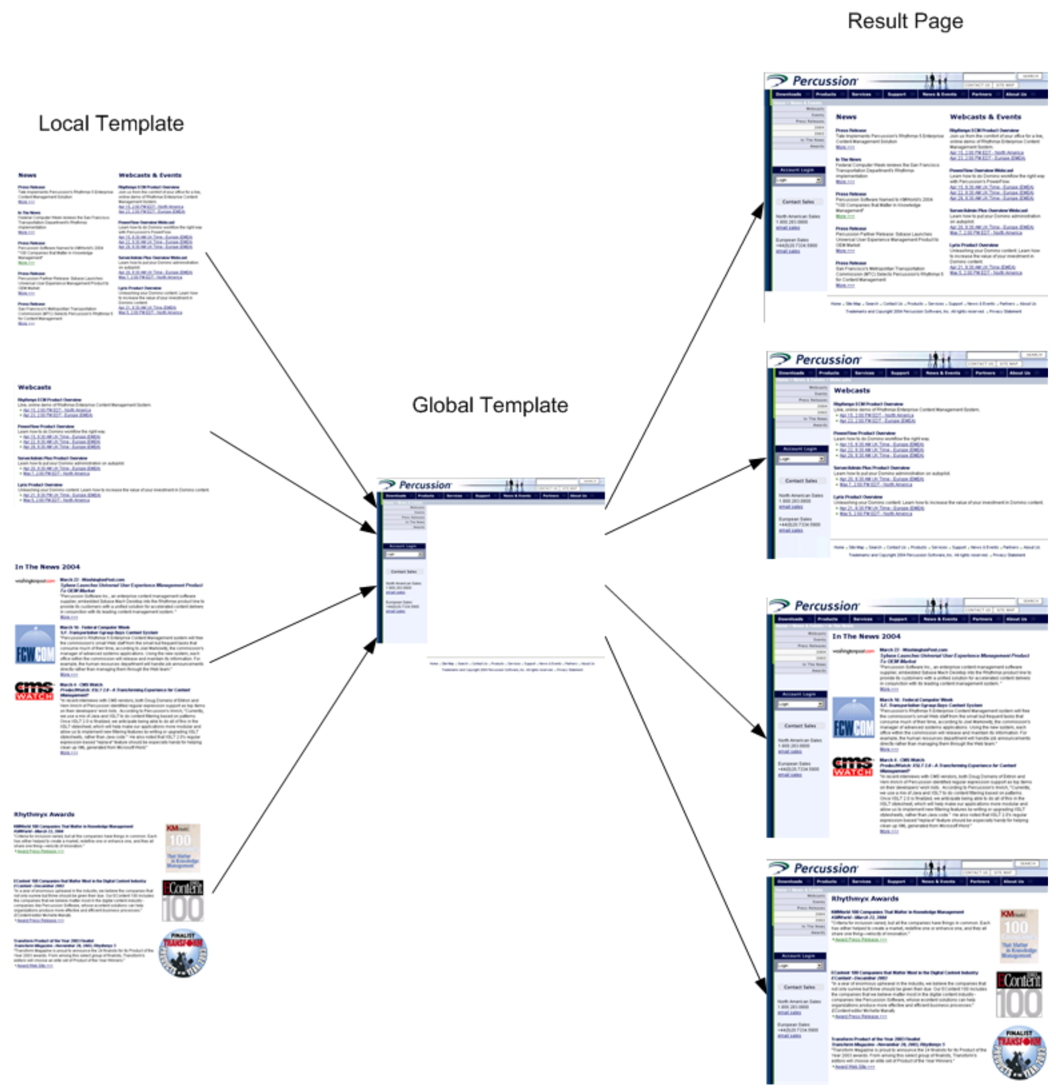


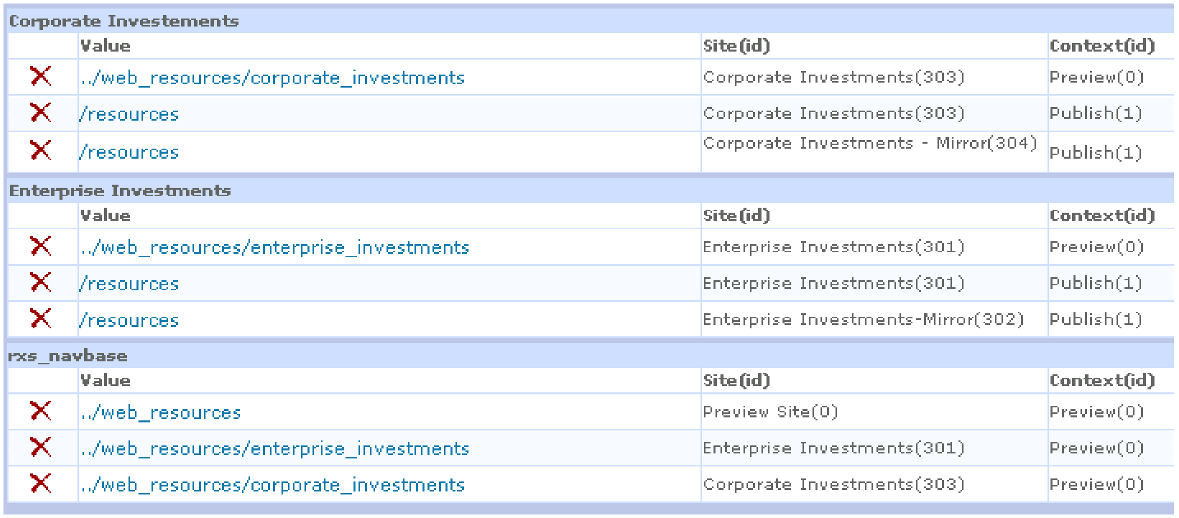
Figure 57: Global Templating of Local Templates

Which master template is selected depends on the following factors:

- If the Variant template calls a specific Global Template, this template is used.
- If the Variant does not call a template, the Site Folder containing the Item (or nearest folder up the tree) defining a Global Template is used.
- If neither the Variant nor any Folder defines a Global Template, the Site level defined Global Template is used.

- If none of the above objects defines a Global Template, no template is used and the Local Page Template is rendered alone.

In addition to Global and Local Templates, a cascading style sheet is tied to each page template.



Corporate Investements

	Value	Site(id)	Context(id)
✗	../web_resources/corporate_investments	Corporate Investments(303)	Preview(0)
✗	/resources	Corporate Investments(303)	Publish(1)
✗	/resources	Corporate Investments - Mirror(304)	Publish(1)

Enterprise Investments

	Value	Site(id)	Context(id)
✗	../web_resources/enterprise_investments	Enterprise Investments(301)	Preview(0)
✗	/resources	Enterprise Investments(301)	Publish(1)
✗	/resources	Enterprise Investments-Mirror(302)	Publish(1)

rxs_navbase

	Value	Site(id)	Context(id)
✗	../web_resources	Preview Site(0)	Preview(0)
✗	../web_resources/enterprise_investments	Enterprise Investments(301)	Preview(0)
✗	../web_resources/corporate_investments	Corporate Investments(303)	Preview(0)

Figure 58: Context Variables

The combination of Global Templating and CSS provides maximum control over style, design, and structure with minimal redundancy.

Customizing Global HTML Templates

You can modify both Global and Local Templates to create the look and feel you want for your site. In both cases, you should follow the specified process, but the process is especially important for Global Templates because of the inherent behavior of XSpLit when you split a template. By default, XSpLit adds import statements of several system and local templates, breaks parts of the page into separate templates (including both a root and a body template), and adds logic to apply Context Variables. When you install a Global Template, you will need to override some of these behaviors.

To create a Global Template:

1. Clean up your HTML to make it as compliant as possible with XHTML standards.
2. Add Slot formatting to your HTML.
3. Add the Local Template marker.
4. Add the Global Template to the Rhythmyx Global Template application.

Updating Global HTML Templates

If you modify a Global Template HTML, you must re-split it and clear the old version from the server's cache. To update a modified Global Template:

1. Open the Workbench.
2. Open the rxs_GlobalTemplates application.

3 Select the Files tab from the Workbench Navigation Panel and locate the updated Global HTML Template.

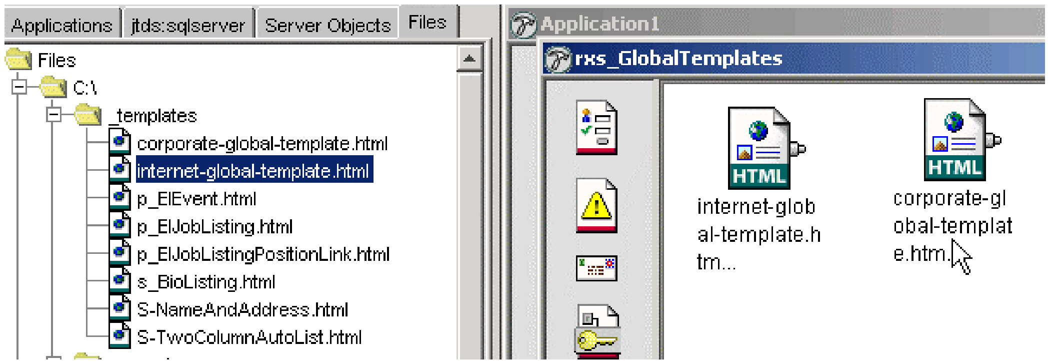


Figure 59: Updating the CI Global Template

4 Drag and drop the new template over the existing template and update the existing web page.

5 Save and close the application.

Deleting a Global HTML Template

Deleting a web page from the rxs_GlobalTemplate application in the workbench will delete the Global Template from the system and modify the rx_Globals.xsl file to remove the corresponding XSL import.

Note: All the local variant style sheets must be modified and re-split to remove the call to the deleted global template.

1 Open the Workbench.

2 Open the rxs_GlobalTemplates application.

3 Select the web page to be deleted.

4 From the Edit menu select [Cut] or press the Delete key.

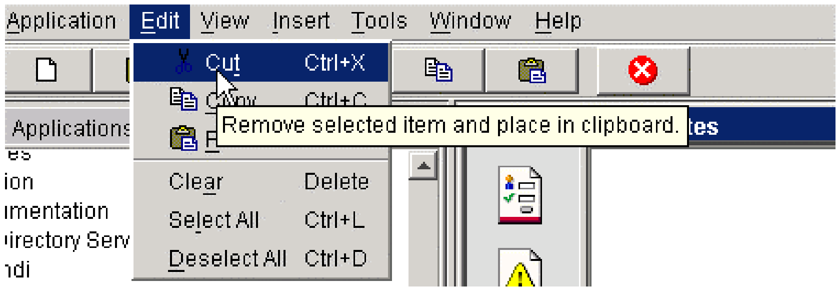


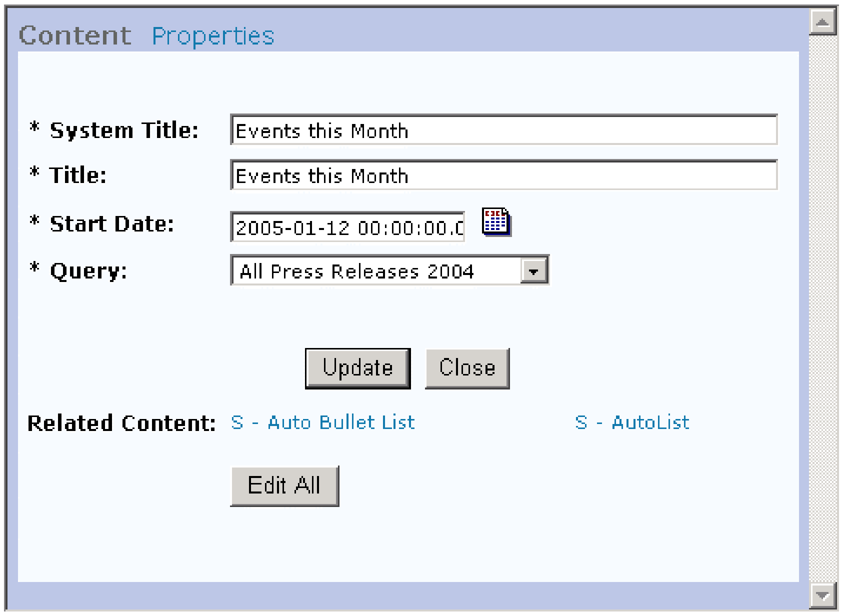
Figure 60: Deleting an Static Web Page

5 Save and close the application.

Automated Indexes
An Automated Index Snippet is an aggregation of other Snippets created as the result of a user selected query. This Snippet is generally placed into a Slot on a Page Variant and is "updated" each time the parent page is assembled. Common uses are lists of the ten most recent Press Releases or Events occurring in a particular month. More complex Automated Indexes can be Monthly Calendars and Navigation Menus.

Generally, a handful of Auto Index Snippets are created and then used and re-used by the general Content Contribution community.

The Content Editor for an Automated Index is the same as any other Editor. The AI Editor will be used to create Items of the Automated Index Type. The key piece of information gathered by Items of this type is the query. Generally, the query is a request URL pointing to an application and passing that application the arguments necessary to return a particular list of Snippets. In FastForward, the query field in the AI Content Editor is displayed in a drop down populated by a Keyword.



* System Title: Events this Month
* Title: Events this Month
* Start Date: 2005-01-12 00:00:00.0
* Query: All Press Releases 2004

Related Content: S - Auto Bullet List S - AutoList

Figure 61: Auto Index Content Editor

The URLs are queries to the rxs_PressRelease_auto application. The selector in this application requires a date and variant id. This is obtained from the **Query** selected in the Content Editor. The query is defined in the FF Auto Index Keyword category. The query is formatted as follows:

```
../rxs_PressRelease_auto/specifiedyear.xml?pr_year=2005
    &sys_variantid=309
```

The rxs_PressRelease_auto assembler uses this information to build a request URL to the rxs_PressRelease_cas assembler for the individual Variants (in this case Variant ID 309). An example URL might be:

```
../rxs_PressRelease_cas/snip.html?sys_revision=1&amp;sys_siteid=301&amp;
sys_authtype=0&amp;sys_contentid=501&amp;sys_variantid=309&amp;sys_comma
nd=&amp;slotname=&amp;sys_context=0
```

The result is an AutoIndex Snippet that can be placed in a Slot like the List Slot on a Generic Page or similar.

The necessary pieces for this process are:

- AutoIndex Content Editor (Exists in FastForward)
- AutoIndex Assembler (Exists in FastForward)
- Query Keyword (Must be slightly modified)
- Target Content Type Auto Assembler (Must be created)
- Target Content Type Assembler (Exists in FastForward)

Setting up a Keyword for Automated Indexes
FastForward is shipped with an AutoIndex Content Editor and Assembler. To create unique Auto Index Items, the only necessary modification to this set of applications is the addition of a new Keyword. The AutoIndex Assembler has two Variants, S - AutoList and S - Auto Bullet List. We can either display the Snippets returned by the target Content Type with one of these templates or create a new one. For this example we will assume that either of these two Variants will fulfill our needs.

Our new Automated Index will be a list of Event Items whose Event Start Date is within the current month. We will return the S - Date Range Snippet (Variant ID = 340) in a single column list either bulleted (S - Auto Bullet List) or not (S - AutoList).

To accomplish this we will only need to:

- Create a new Keyword,
- Create an ei_Event_auto Assembler to return the desired list of Items.

Creating a new Keyword

1. Log into the Content Explorer.
2. Navigate to the System Tab and select Keywords By Name.
3. Select the FF Auto Index Queries Keyword.
4. Select Add Choice.
5. Create a new Keyword with the following values:

- Choice Label - Events for the Current Month
- Description - Auto Index to return a List of Items based on the current month.
- Choice Value - ../ei_Event_auto/eventsforthismonth.html?sys_variantid=340
- Sort Order - 5

1 Select [**OK**] to save the Keyword.

Creating an Automated Index Query Assembler
We will now create the Assembly resource to return the list of the Event Item Snippets (Variant ID = 340) who's Start Date matches the current month and year.

Create the Assembler

1. Open the Workbench and open a new (Application > New) application.

2 In the Files tab, navigate to the Rhythmyx\DTD directory and locate the sys_AutoIndex.dtd. Drag and drop this file into the new workspace. Select [Query] when prompted.

3 Rename the resource to match the name we defined in our new AutoIndex Query, eventsforthismonth.

Add Pre-Processing Exits to the Assembler

We will need to accomplish a few things by using both pre and post exits on this Assembler:

- sys_AddCurrentTimeDate (Generic Request Pre-Processing Java Exit) will be used to pass the current month and year to the query. We will use separate instances of this exit to populate two custom html parameters, currentmonth and currentyear.

1 In the Workbench, select the Server Objects tab.

2 Open the Request Pre-Processing tree and expand the Generic folder. Drag the sys_AddCurrentDateTime Exit to the eventsforthismonth resource.

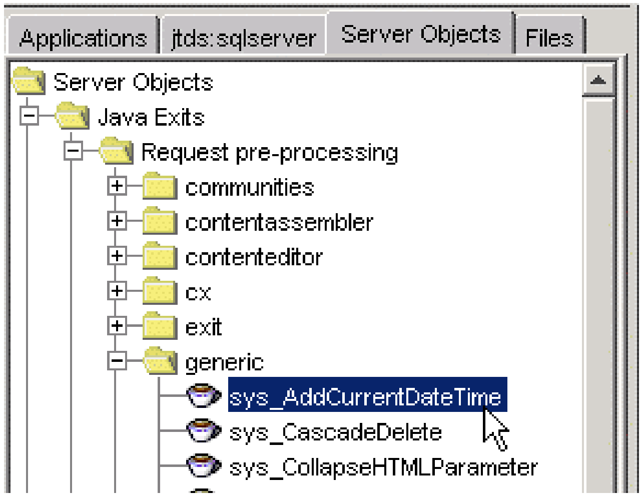
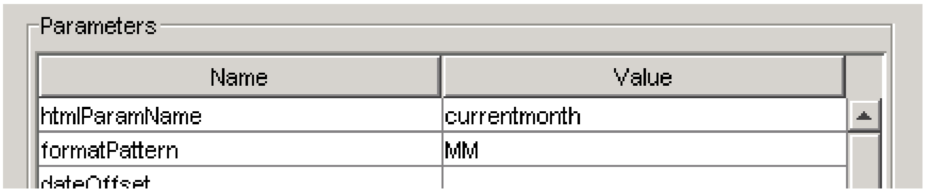


Figure 62: sys_AddCurrentDateTime Exit

3 Double click the Exit icon and select the inserted Exit as a pre-exit. Add the following parameters:

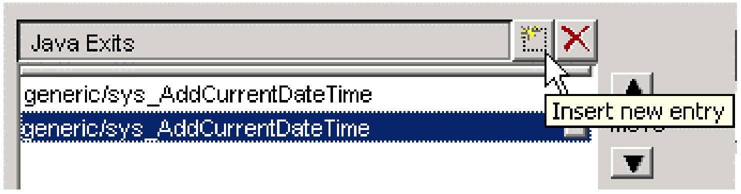
- Name - htmlParamName Value - currentmonth
- Name - formatPattern Value - MM


Parameters

Name	Value
htmlParamName	currentmonth
formatPattern	MM
dateOffset	

Figure 63: Adding Parameters to the sys_CurrentDateTime Exit

4 Select the New Entry button and add a second instance of the sys_AddCurrentDateTime Exit.



generic/sys_AddCurrentDateTime
generic/sys_AddCurrentDateTime

Figure 64: Adding a Second Entry

5 Select the new Exit and add the following parameters:

- Name - htmlParamName Value - currentyear
- Name - formatPattern Value - yyyy

6 Select [OK] to close the Exit Properties dialog.

Adding these two Exits will add two arguments HTML parameters to the request URL. Example: ?currentyear=2005&currentmonth=01

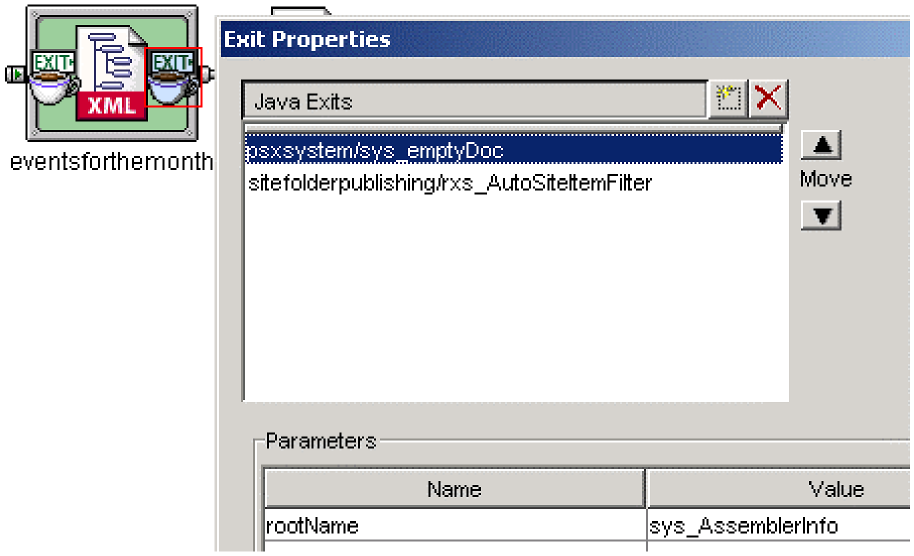
Add Result Document Exits to the Assembler

1 The next set of Exits are a Result Document Processing Exits. Expand the Result Document Processing tree and locate the sys_EmptyDoc Exit in the psxsystem tree. Drag and drop this Exit on the eventsforthismonth resource.

2 Double click the new Java Exit icon and select the sys_EmptyDoc Exit.

3 Add the following Parameter to the Exit:

4 Name - rootName Value - sys_AssemblerInfo



psxsystem/sys_emptyDoc
sitefolderpublishing/rxs_AutoSiteItemFilter

Name	Value
rootName	sys_AssemblerInfo

Figure 65: Configuring the sys_EmptyDoc Exit

This Exit handles situations when there are no results returned from a query. Instead of no XML being returned, an empty document with the root element <sys_AssemblerInfo /> will be returned.

5 Press the new entry button and add the sitefolderpublishing/rxs_AutoSiteItemFilter Exit.

6 Add the following Parameters to the Exit:

7 Name - ItemSiteFolderURL Value - ../rx_supportSiteFolderContentList/lookupSiteFolderRoot.xml

8 Select [OK] to close the Exit Properties dialog.

Add tables to Assembler and make joins

1 Open the eventsforthismonth Resource properties. Select the Data tab, locate, and drag and drop the following tables onto the datapipe:

- CONTENTSTATUS
- CONTENTVARIANTS
- RXS_CT_EVENT
- STATES
- RXS_CT_SHARED

2 Open the Backend Datatank and join the tables as follows:

Left	Right
RXS_CT_SHARED.CONTENTID	CONTENTSTATUS.CONTENTID
RXS_CT_SHARED.REVISIONID	CONTENTSTATUS.CURRENTREVISION
RXS_CT_EVENT.CONTENTID	CONTENTSTATUS.CONTENTID
RXS_CT_EVENT.REVISIONID	CONTENTSTATUS.CURRENTREVISION
STATES.STATEID	CONTENTSTATUS.CONTENTSTATEID
STATES.WORKFLOWAPPID	CONTENTSTATUS.WORKFLOWAPPID
CONTENTSTATUS.CONTENTTYPEID	CONTENTVARIANTS.CONTENTTYPEID

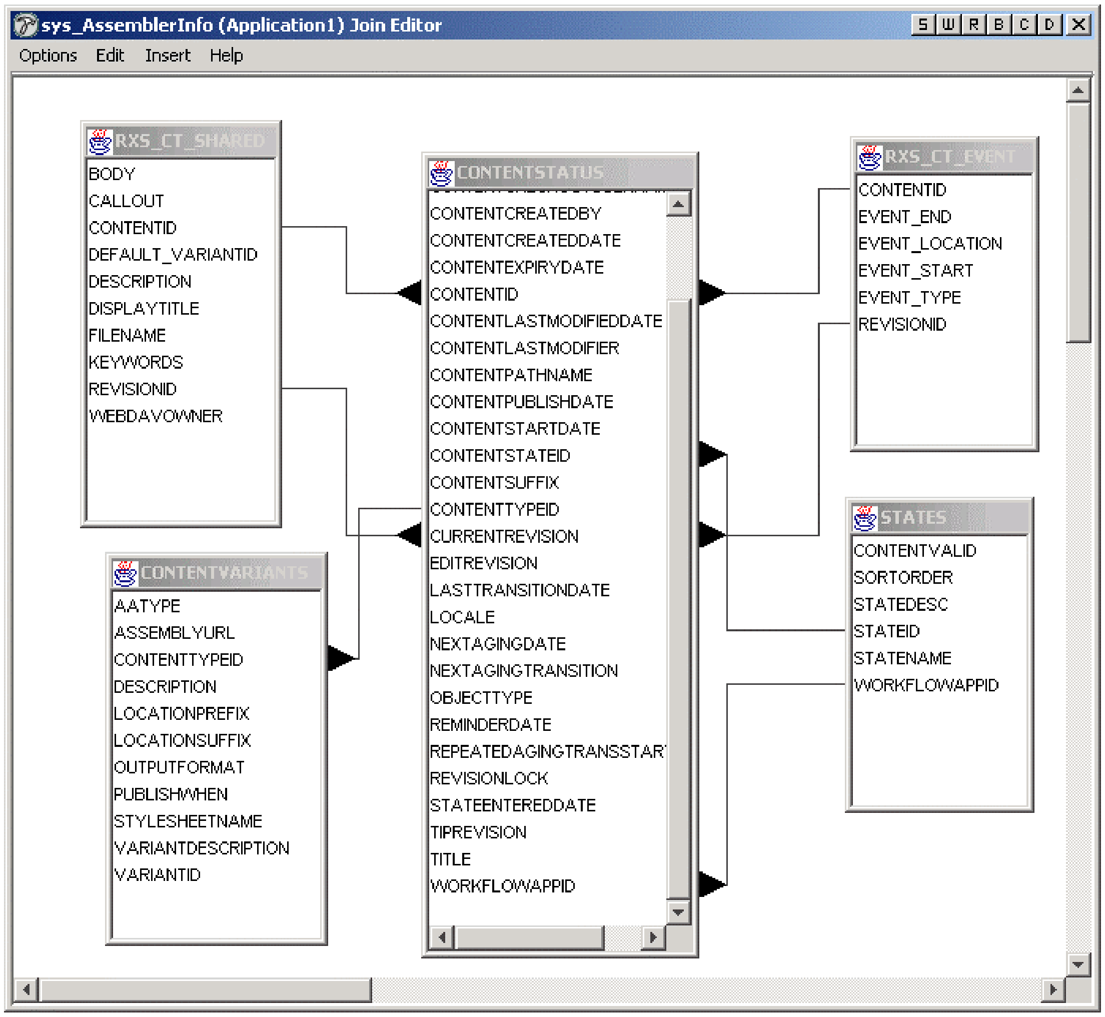


Figure 66: Table Joins for Event Auto Index

3 Close the Datatank and when prompted select [Yes] to save the changes.

Define Where clause

We only want to return Public Items who's Event Start Date shares a common Month and Year with the current date. Additionally, we only want Variants equal to the Variant ID defined in the All Events for the Current Month Keyword. We achieve this by defining a Where clause in our query

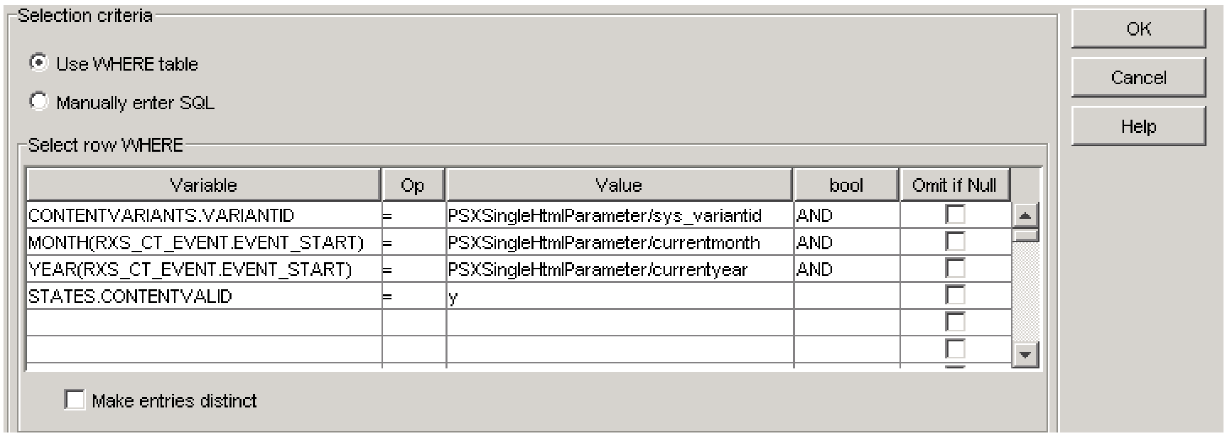
1 Open the eventsforthismonth resource Properties.

2 Open the Selector Properties.

3 Define the following in the Where Table:

Variable	**Operator**	**Value**	**Boolean**
CONTENTVARIANTS.VARIANTID	=	SingleHTMLParameter/sys_variantid	AND

Function - Month **Name** **Value** source RXS_CT_EVENT.EVENT_START	=	SingleHTMLParameter/currentmonth	AND
Function - Year **Name** **Value** source RXS_CT_EVENT.EVENT_START	=	SingleHTMLParameter/currentyear	AND



Use WHERE table [✓]

Variable	Op	Value	bool	Omit if Null
CONTENTVARIANTS.VARIANTID	=	PSXSingleHtmlParameter/sys_variantid	AND	☐
MONTH(RXS_CT_EVENT.EVENT_START)	=	PSXSingleHtmlParameter/currentmonth	AND	☐
YEAR(RXS_CT_EVENT.EVENT_START)	=	PSXSingleHtmlParameter/currentyear	AND	☐
STATES.CONTENTVALID	=	y		☐
				☐
				☐

Figure 67: Defining the Where Clause for the Event Auto Resource

4 Select [OK] to close the dialog.

Add a Result Pager to the Datapipe

The Result Page sorts the returned records on the key specified. We can also use the Pager to limit the number of records passed to the Mapper.

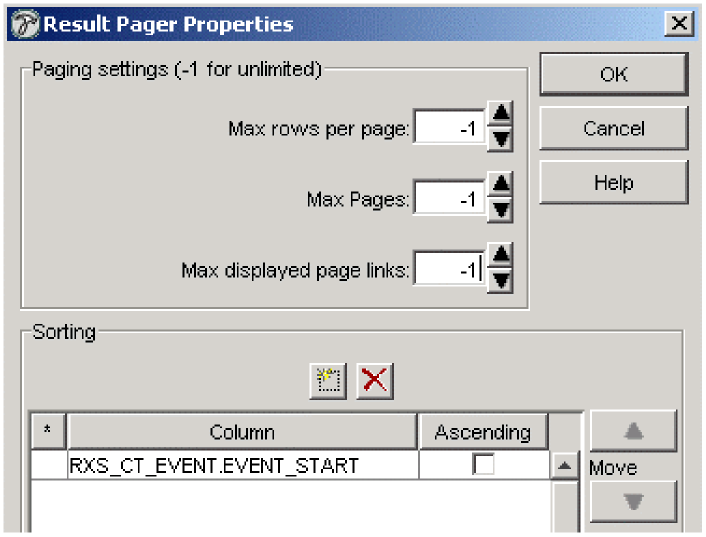
1 Open the eventsforthismonth properties to expose the Pipe.

2 From the menu insert a Result Pager (Insert > Result Pager).

3 Drag the Result Pager to the "bolt" between the Data Tank and the Mapper.

4 Open the Pager and press the new entry button.

5 Select RXS_CT_EVENT.EVENT_START for the sorting column and un-select Ascending.



Paging settings (-1 for unlimited)

Max rows per page: -1

Max Pages: -1

Max displayed page links: -1

*	Column	Ascending
	RXS_CT_EVENT.EVENT_START	☐

Figure 68: Result Pager for Event Items

6 Press [OK] to close the Properties dialog.

Map the required fields

At this point all of the required records should be returned (though we can't see them yet). The last step is to map the resulting data and build the Assembly URLs for the Snippets we want to send back to the AutoIndex Assembler. A few standard Mappings will be defined along with the use of the sys_MakeIntLink UDF to build the Assembly URL to the S - Date Range Snippet.

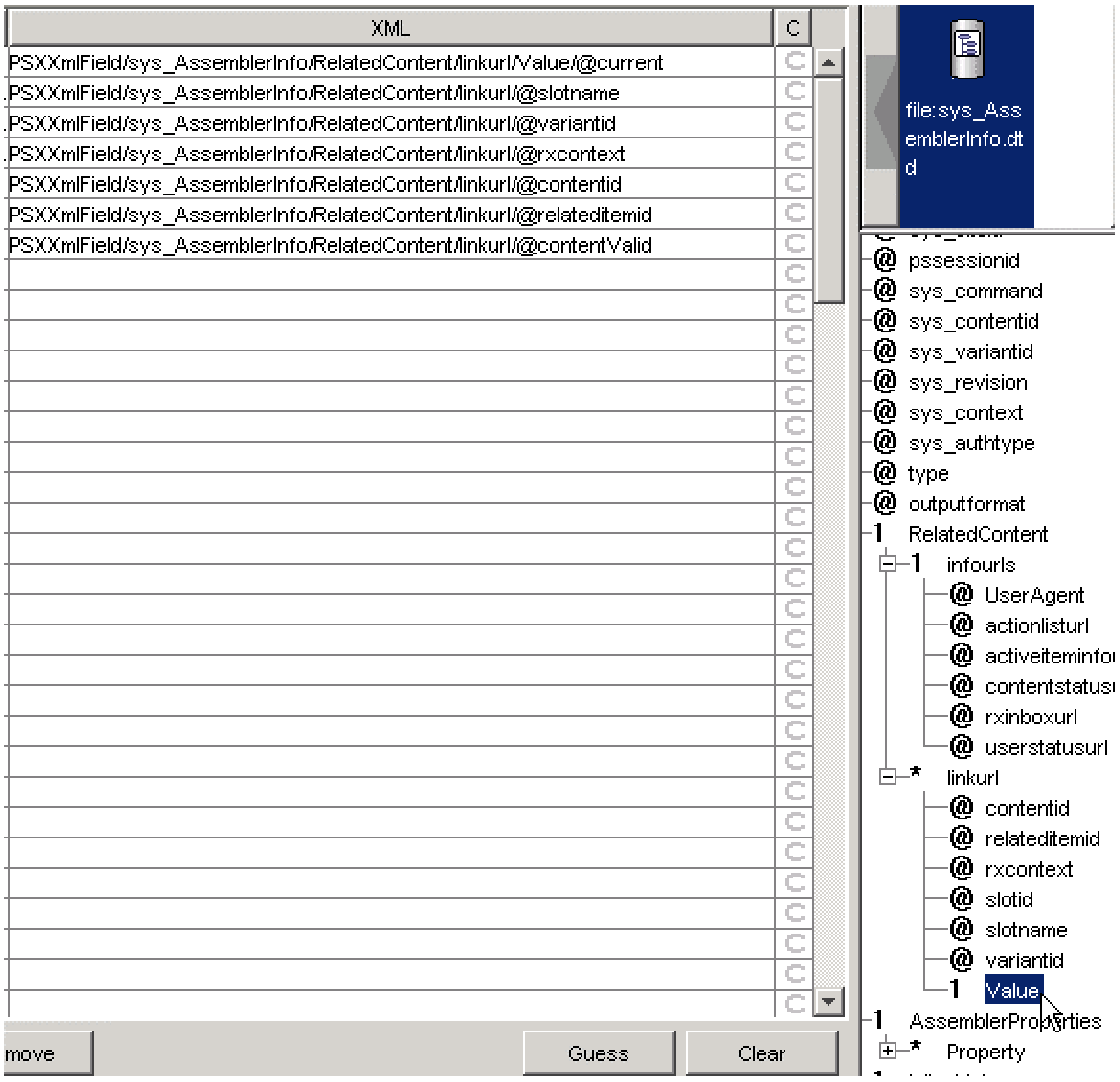
1 Open the eventsforthismonth properties to expose the Pipe.

2 Open the Mapper properties.

3 Under the linkurl node, you may not see the attributes that you want to map. Click in the field on the XML side of the mapper and click ... to see the attribute choices. Insert the following attributes to the XML side of the Mapper:

- slotname
- variantid
- rxcontext
- contentid
- relateditemid

- contentValid (Not in DTD. Must be typed in manually.)
- Value (after dropping this element, the attribute @current will need to be entered by hand)



```
XML
PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/Value/@current
PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@slotname
PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@variantid
PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@rxcontext
PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@contentid
PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@relateditemid
PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@contentValid
```

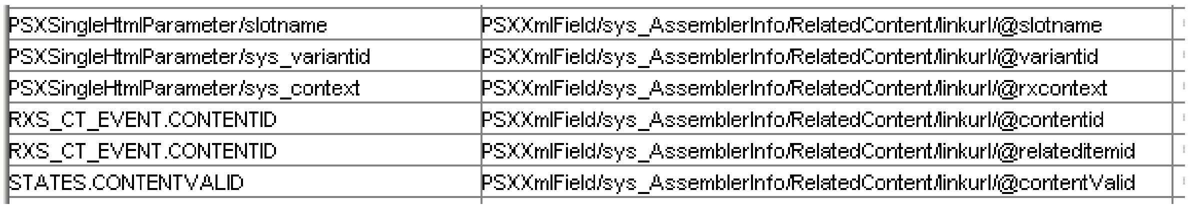
```
file:sys_AssemblerInfo.dtd
@ pssessionid
@ sys_command
@ sys_contentid
@ sys_variantid
@ sys_revision
@ sys_context
@ sys_authtype
@ type
@ outputformat
1 RelatedContent
  1 infourls
    @ UserAgent
    @ actionlisturl
    @ activeiteminfo
    @ contentstatus
    @ rxinboxurl
    @ userstatusurl
  * linkurl
    @ contentid
    @ relateditemid
    @ rxcontext
    @ slotid
    @ slotname
    @ variantid
    1 Value
1 AssemblerProperties
  * Property
```

Figure 69: Building the Result XML

4 Match these elements with the following Back-end values:

Backend	**XML**
Single HTML Parameter, slotname	@slotname
Single HTML Parameter, sys_variantid	@variantid
Single HTML Parameter, sys_context	@rxcontext
Backend Column, RXS_CT_EVENT.CONTENTID	@contentid
Backend Column, RXS_CT_EVENT.CONTENTID	@relateditemid

Backend	**XML**
Backend Column, STATES.CONTENTVALID	@contentValid



PSXSingleHtmlParameter/slotname	PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@slotname
PSXSingleHtmlParameter/sys_variantid	PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@variantid
PSXSingleHtmlParameter/sys_context	PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@rxcontext
RXS_CT_EVENT.CONTENTID	PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@contentid
RXS_CT_EVENT.CONTENTID	PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@relateditemid
STATES.CONTENTVALID	PSXXmlField/sys_AssemblerInfo/RelatedContent/linkurl/@contentValid

Figure 70: Mapping Back-end Values

5 The last piece is to build the Snippet URL being mapped to @current. Begin by dragging the Global User Defined Function, sys_MakeIntLink to match the @current element.

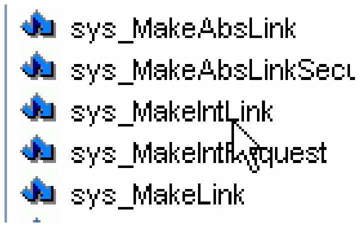
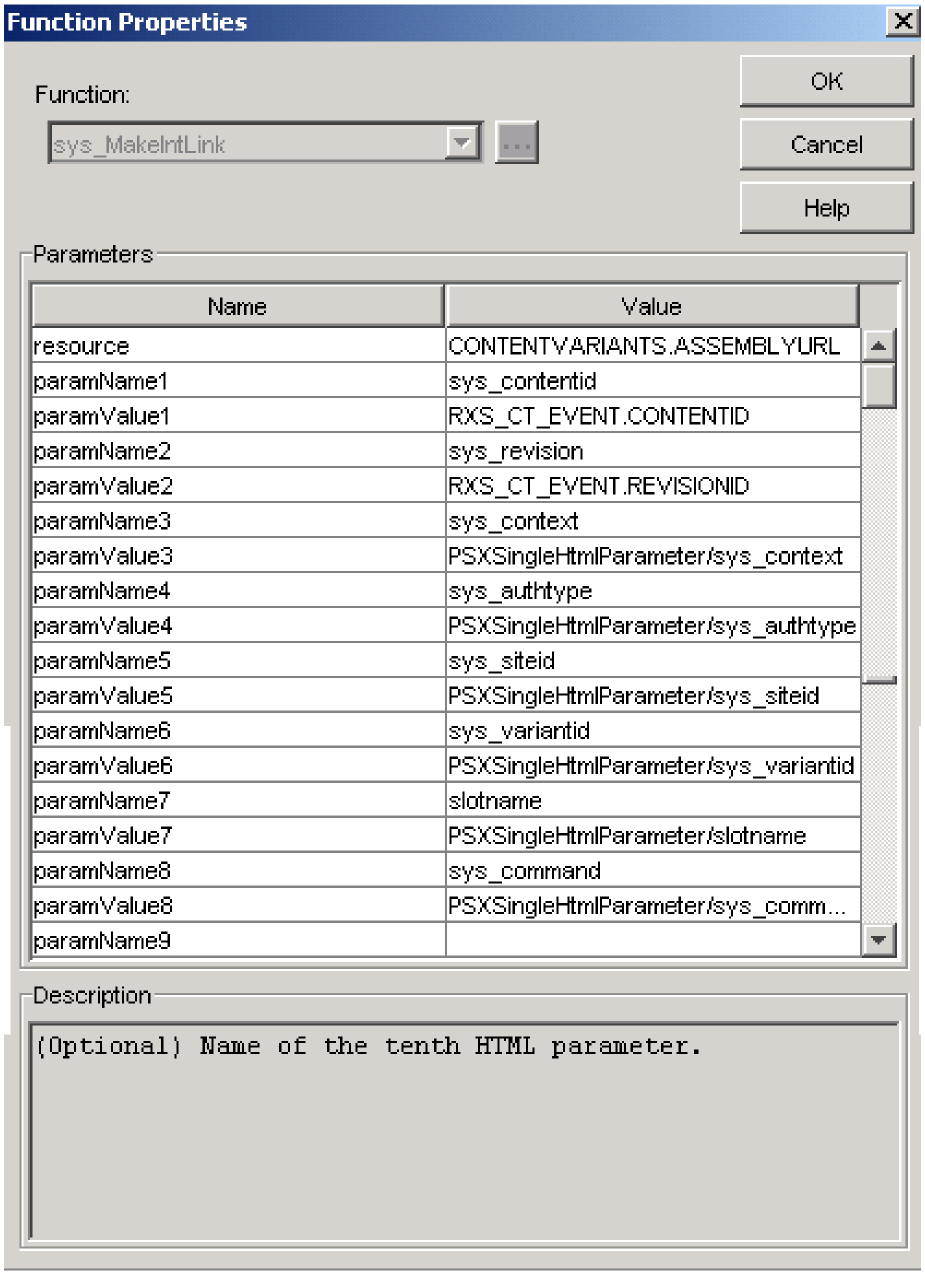


Figure 71: sys_MakeIntLink UDF

6 Define the following properties for the UDF:

paramName	**paramValue**
	Backend Column, CONTENTVARIANTS.ASSEMBLYURL
Literal, sys_contentid	Backend Column, RXS_CT_EVENT.CONTENTID
Literal, sys_revision	Backend Column, RXS_CT_EVENT.REVISIONID
Literal, sys_context	Single HTML Parameter, sys_context
Literal, sys_authtype	Single HTML Parameter, sys_authtype
Literal, sys_siteid	Single HTML Parameter, sys_siteid
Literal, sys_variantid	Single HTML Parameter, sys_variantid
Literal, slotname	Single HTML Parameter, slotname
Literal, sys_command	Single HTML Parameter, sys_command



Name	Value
resource	CONTENTVARIANTS.ASSEMBLYURL
paramName1	sys_contentid
paramValue1	RXS_CT_EVENT.CONTENTID
paramName2	sys_revision
paramValue2	RXS_CT_EVENT.REVISIONID
paramName3	sys_context
paramValue3	PSXSingleHtmlParameter/sys_context
paramName4	sys_authtype
paramValue4	PSXSingleHtmlParameter/sys_authtype
paramName5	sys_siteid
paramValue5	PSXSingleHtmlParameter/sys_siteid
paramName6	sys_variantid
paramValue6	PSXSingleHtmlParameter/sys_variantid
paramName7	slotname
paramValue7	PSXSingleHtmlParameter/slotname
paramName8	sys_command
paramValue8	PSXSingleHtmlParameter/sys_comm...
paramName9	

```
(Optional) Name of the tenth HTML parameter.
```

Figure 72: Properties for the MakeIntLink UDF

7 Press [OK] to close the Function Properties dialog.

8 Select the Return Empty XML checkbox at the bottom of the mapper to allow for an empty XML doc to be returned if no records are found.

9 Press [OK] to close the Mapper.

10 Close the datapipe.

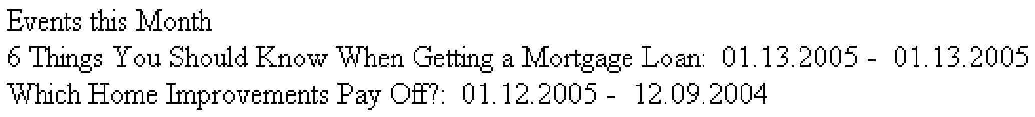
Save and name the application

1 Save the application (Application > Save) and name it ei_Event_auto. This name should match the application name we defined in our Events for the Current Month Keyword (../ei_Event_auto/eventsforthismonth.html?sys_variantid=340).

2 Start (Application > Start) the application.

Testing the Event Auto Index

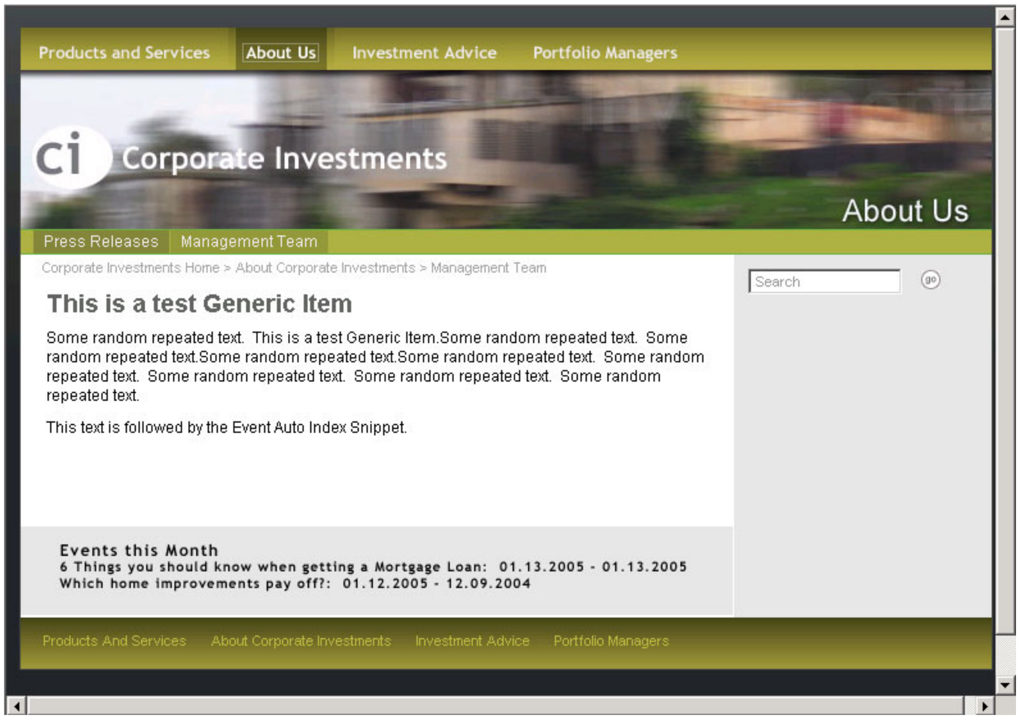
We have completed all the necessary pieces for our Event Auto Index. Some of the pieces were part of the FastForward package, we had to create the others. When used together we have the ability to create an Item of the AutoIndex Type selecting our new Keyword and the result is a Snippet that returns a list of Event Items occurring in the current month.



Events this Month
6 Things You Should Know When Getting a Mortgage Loan: 01.13.2005 - 01.13.2005
Which Home Improvements Pay Off?: 01.12.2005 - 12.09.2004

Figure 73: Event Auto Index Snippet

This Snippet can then be placed in any of the existing Slots (List or Sidebar).



Corporate Investments Home > About Corporate Investments > Management Team

This is a test Generic Item

Some random repeated text. This is a test Generic Item.Some random repeated text. Some random repeated text.Some random repeated text.Some random repeated text. Some random repeated text. Some random repeated text. Some random repeated text. Some random repeated text.

This text is followed by the Event Auto Index Snippet.

Events this Month
6 Things you should know when getting a Mortgage Loan: 01.13.2005 - 01.13.2005
Which home improvements pay off?: 01.12.2005 - 12.09.2004

Figure 74: Generic Page with Events Auto Index in List Slot

1 Log into the Content Explorer.

2 Navigate to the, "test folder, " right click it and create a new Item of the Auto Index Type (New Item > Auto Index). Select our newly created Events for the Current Month Query.

3 [Insert] the Item and [Close] the Content Editor.

4 Preview the new Auto Index Item. If no Items are returned, be sure to confirm the existence of Event Items with an Event Start Date occurring within the current month.

5 Once this Snippet is returning records, return back to the test folder and create a Generic Item.

6 Open the Table Editor for the Generic Item and add the new Auto Index Item to the List Slot.

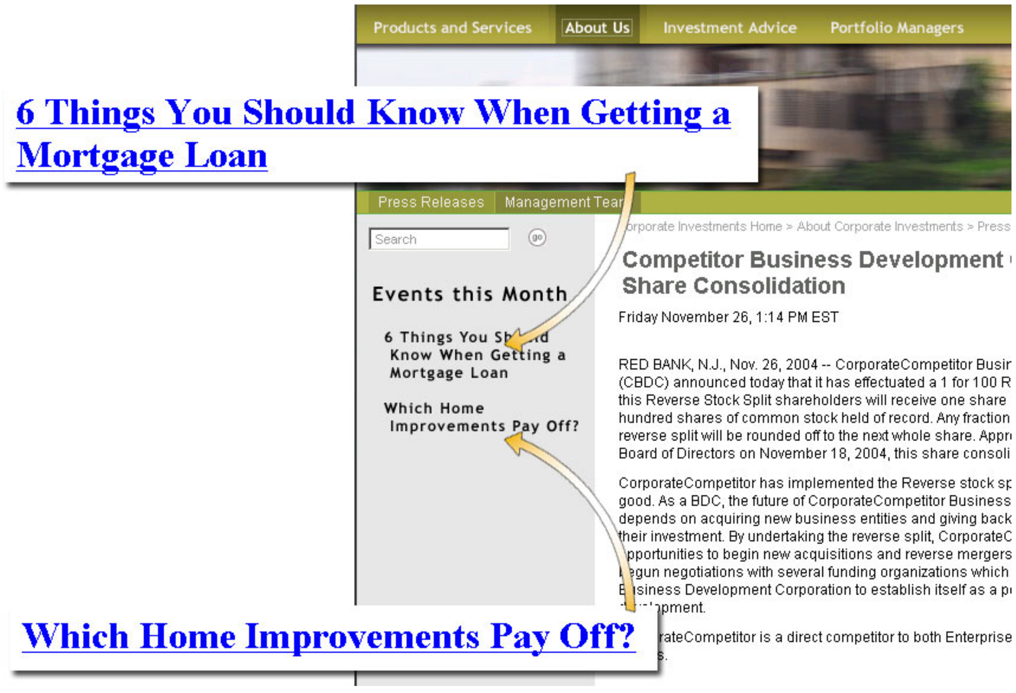
Note: The Generic Item may need to be dragged and dropped into a Site Folder to import the appropriate Global Template. This example shows the Generic Item in the Corporate Investments' Brief Site Folder.

---

7 Preview the Generic Item. The List Slot should be populated with the list of Snippets.

Automatically Populating a Slot using AutoRelatedContent

Our previous Auto Index example used either of the two out of the box Auto Index Snippets, S - AutoList and S - Bulleted Auto List. Once we created the Auto Index Item we had to manually select one of the two available Snippets and manually place it into a Slot on a page (our example placed it in the List Slot on a Generic Page). Our goal in this activity is to automatically populate all CI Event pages with this list of Events. We will create a new Slot (AutoIndexSlot) under the existing Sidebar Slot. The AutoIndexSlot already exists in the system and only needs to be added to the markup for the existing P - CI Event template. We will change the Snippets filling this Slot from S - Date Range Snippets to S - Title Link Snippets (Variant ID = 305).


Products and Services
About Us
Investment Advice
Portfolio Managers
6 Things You Should Know When Getting a Mortgage Loan
Press Releases
Management Tear
Search
go
Events this Month
6 Things You Should Know When Getting a Mortgage Loan
Which Home Improvements Pay Off?
Which Home Improvements Pay Off?

orporate Investments Home > About Corporate Investments > Press

Competitor Business Development
Share Consolidation

Friday November 26, 1:14 PM EST

RED BANK, N.J., Nov. 26, 2004 -- CorporateCompetitor Busir
(CBDC) announced today that it has effectuated a 1 for 100 R
this Reverse Stock Split shareholders will receive one share
hundred shares of common stock held of record. Any fraction
reverse split will be rounded off to the next whole share. Appr
Board of Directors on November 18, 2004, this share consoli

CorporateCompetitor has implemented the Reverse stock sp
good. As a BDC, the future of CorporateCompetitor Business
depends on acquiring new business entities and giving back
their investment. By undertaking the reverse split, CorporateC
pportunities to begin new acquisitions and reverse mergers
egun negotiations with several funding organizations which
siness Development Corporation to establish itself as a p
pment.

rateCompetitor is a direct competitor to both Enterprise
s.

Figure 75: Event Auto Index on Event Page

We will be using the sys_AutoRelatedContent Exit to insert the Snippets into a new Slot on the CI Event page (P - CI Event). Since contributors will not be involved in the process (it is automatic) we will not have to create Items of the Auto Index Type, will not need a new Keyword entry, and will not have to use Active Assembly to add the content to the Event page. When any Event page is assembled, the current month's list of Events will populate the Slot (AutoIndexSlot) listing events in with the S - Title Link Snippet.

Update the AutoIndexSlot Slot

The currently registered AutoIndexSlot only allows Press Release S - Title Link Snippets. We need to add the Event S - Title Link Snippets to this list.

1 Return to the Content Explorer and navigate to the System tab.

2 Select Slots [By Name].

3 Select the [AutoIndexSlot] Slot.

4 Add the S - Title Link Snippet of Event to the list of allowed content for this Slot.

Add AutoEventList Slot to P - CI Event Template

1 Locate the html file p_CIEvent.html in the Rhythmyx\rxs_Event_cas\src directory and save a copy locally. Do not change the template's name.

2 Open the template and find the existing Sidebar slot. Immediately following the <!-- end slot Sidebar Slot --> comment, add the new AutoIndexSlot template:

```
<!-- start slot AutoIndexSlot slot -->
<div id="Related">
   <div class="RelatedContent">
      <span class="relatedHeader">Events this month</span>
      <br />
      <br />
      <!-- start snippet wrapper -->
      <span psxeditslot="no" slotname="AutoIndexSlot">
         AutoIndexSlot</span>
      <!-- end snippet wrapper -->
   </div>
</div>
<!-- end slot AutoIndexSlot slot -->
```

3 Save the template.

4 Open the Workbench and open the rxs_Event_cas application. Drag the updated p_CIEvent.html template onto the p_CIEvent.xsl page.

5 Save the application.

Map the Query to the ei_Event_auto application

Instead of having the user create an Item of the Auto Index Type to pick the query, we will code this query directly into the Event assembler. The sys_MakeIntLink UDF will be used to generate the query for each Event Snippet. We will map this to the attribute, @query.

1 Open the rxs_Event_cas application.

2 Open the Page resource Properties.

3 Open the Mapper.

4 Add a new entry to the mappings:

- XML - XML Element, whitebox/@query

1 The second step is to build the Snippet URL being mapped to @query. Begin by dragging the Global User Defined Function, sys_MakeIntLink to match the @query element.

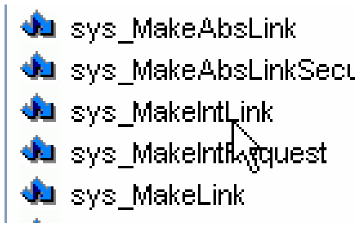
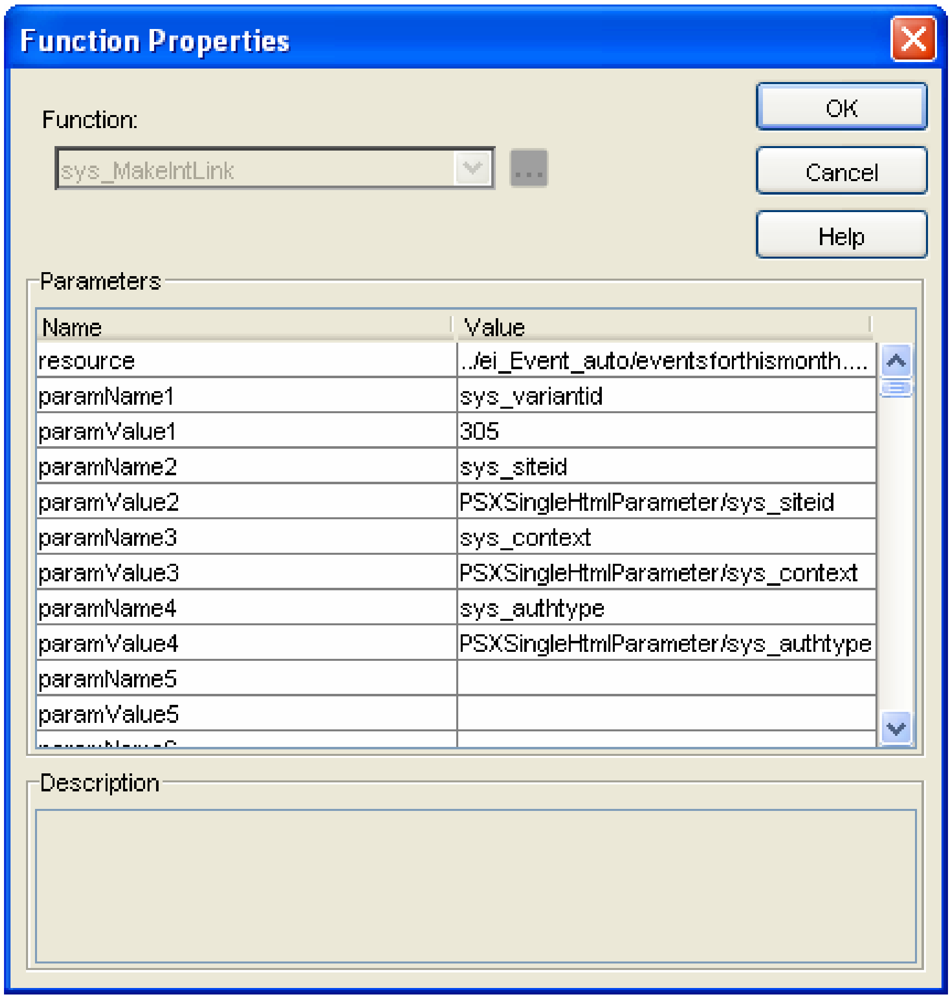


Figure 76: sys_MakeIntLink UDF

2 Define the following properties for the UDF:

paramName	**paramValue**
resource	Literal, ../ei_Event_auto/eventsforthismonth.xml
Literal, sys_variantid	Literal, 305
Literal, sys_siteid	SingleHTMLParameter, sys_siteid
Literal, sys_context	SingleHTMLParameter, sys_context
Literal, sys_authtype	SingleHTMLParameter, sys_authtype



Function: sys_MakeIntLink

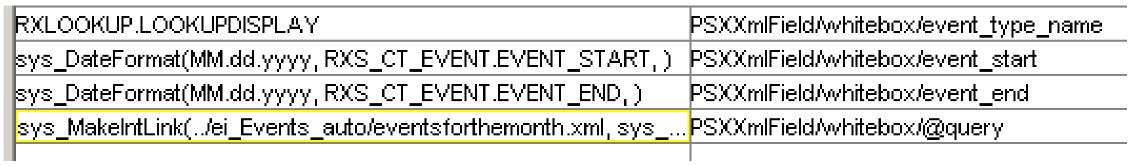
Parameters

Name	Value
resource	../ei_Event_auto/eventsforthismonth....
paramName1	sys_variantid
paramValue1	305
paramName2	sys_siteid
paramValue2	PSXSingleHtmlParameter/sys_siteid
paramName3	sys_context
paramValue3	PSXSingleHtmlParameter/sys_context
paramName4	sys_authtype
paramValue4	PSXSingleHtmlParameter/sys_authtype
paramName5	
paramValue5	

Description

Figure 77: Parameters for sys_MakeIntLink

Note: This is basically replacing the Keyword we used in our previous Auto Index activity.



RXLOOKUP.LOOKUPDISPLAY	PSXXmlField/whitebox/event_type_name
sys_DateFormat(MM.dd.yyyy, RXS_CT_EVENT.EVENT_START,)	PSXXmlField/whitebox/event_start
sys_DateFormat(MM.dd.yyyy, RXS_CT_EVENT.EVENT_END,)	PSXXmlField/whitebox/event_end
sys_MakeIntLink(../ei_Events_auto/eventsforthemonth.xml, sys_...	PSXXmlField/whitebox/@query

Figure 78: Mapping for sys_MakeIntLink

3 Press [OK] to close the Function Properties dialog.

4 Close the pipe

Add sys_AutoRelatedContent Exit to rxs_Event_cas page Assembly Resource

The last piece of this process is to automatically add the Auto Snippet into the AutoIndexSlot Slot.

1 Open the Java Exit object on the Page resource in the rxs_Event_cas application.

2 Use the [Insert Entry] button to add the contentassembler/sys_casAutoRelatedContent UDF.

Note: This Exit must occur AFTER sys_casAddAssemblerInfo

3 Define the following parameters for the UDF:

Name	Value
LinkUrl	query
SlotNameOverride	AutoIndexSlot

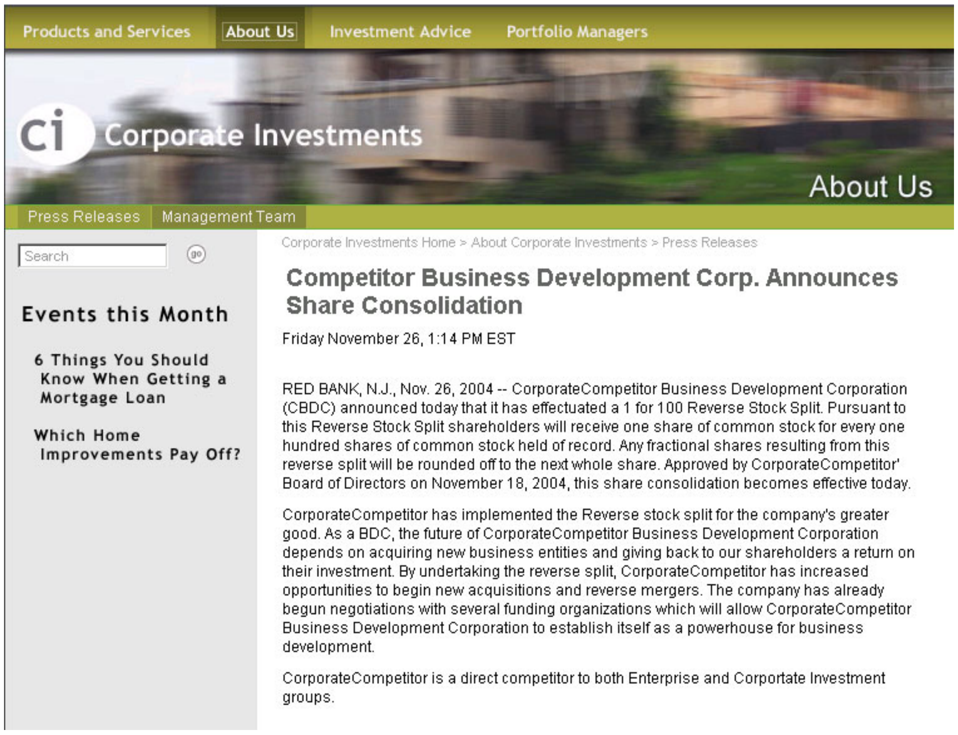
This basically tells the assembler to populate the AutoIndexSlot with the Snippets mapped to the @query attribute in the mapper.

4 Press [OK] to close the Exit Properties dialog.

5 Save (Application > Save) the application.

Test New Auto Slot on Event Pages

At this point, previewing any CI Event Page Item will present a list of Event S - Title Link Snippets below the Sidebar Slot. If Items do not show up in this Slot, confirm the existence of Event Items with Event Start Dates equal to the current month.



Events this Month

6 Things You Should Know When Getting a Mortgage Loan

Which Home Improvements Pay Off?

Competitor Business Development Corp. Announces Share Consolidation

Friday November 26, 1:14 PM EST

RED BANK, N.J., Nov. 26, 2004 -- CorporateCompetitor Business Development Corporation (CBDC) announced today that it has effectuated a 1 for 100 Reverse Stock Split. Pursuant to this Reverse Stock Split shareholders will receive one share of common stock for every one hundred shares of common stock held of record. Any fractional shares resulting from this reverse split will be rounded off to the next whole share. Approved by CorporateCompetitor' Board of Directors on November 18, 2004, this share consolidation becomes effective today.

CorporateCompetitor has implemented the Reverse stock split for the company's greater good. As a BDC, the future of CorporateCompetitor Business Development Corporation depends on acquiring new business entities and giving back to our shareholders a return on their investment. By undertaking the reverse split, CorporateCompetitor has increased opportunities to begin new acquisitions and reverse mergers. The company has already begun negotiations with several funding organizations which will allow CorporateCompetitor Business Development Corporation to establish itself as a powerhouse for business development.

CorporateCompetitor is a direct competitor to both Enterprise and Corportate Investment groups.

Figure 79: Event Page with Auto Related Content

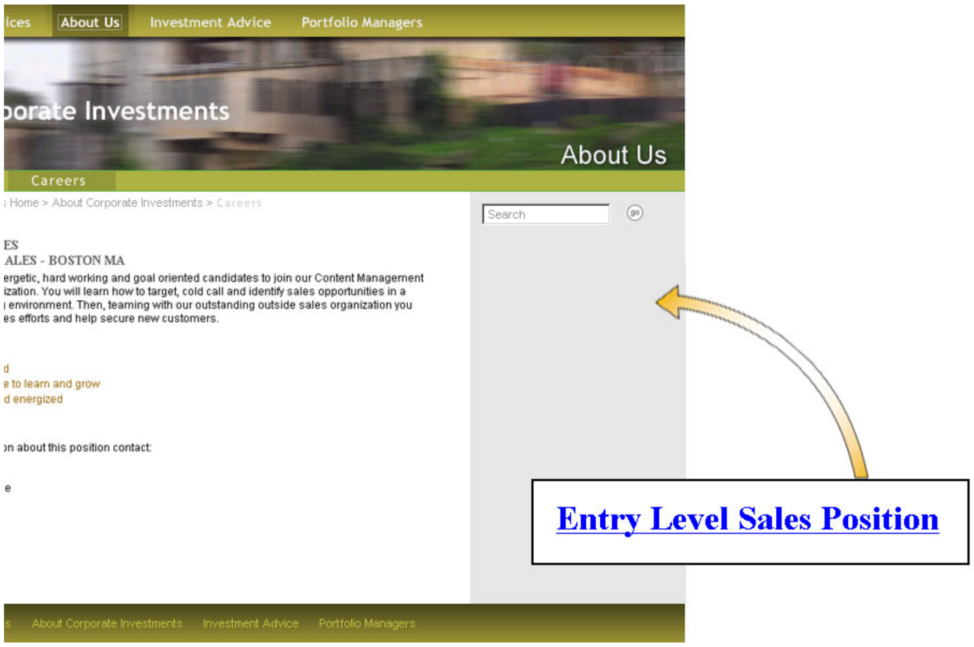
Page Markup Summary

The following is a summarized list of Rhythmyx specific tags used when marking up HTML templates for use with Rhythmyx.

Use	Markup
Dynamic Content Replacement (Local Fields)	<span psxedit="city" psxshow="psx-city" />
Dynamic Content Replacement (Shared Fields)	<span psxedit="body" psxshow="psx-shared/body" />
Slot	<!-- start slot sampleslot--> <!-- start snippet wrapper --> <span psxeditslot="yes/no" slotname="sampleslot" /> <!-- end snippet wrapper --> <!-- end slot sampleslot -->
Slot Using Slot Template	<span psxeditslot="yes" slotname="Home List" template="MultiColumnSlot" param1="'COLS', '3'" />
Context Variable (script tag)	<script psx-src="$rxs_navbase/js/mouseover.js" language="javascript" type="text/javascript">;</script>
Context Variable (link tag)	<link rel="stylesheet" psx-href="$rxs_navbase/css/rxs_styles.css" href="web_resources/css/rxs_styles.css" type="text/css" />
Context Variable (img tag)	<img psx-src="$rxs_navbase/images/go_black.gif" src="images/go_black.gif" alt="Go Image" />
Snippet Header	<!-- begin XSL --> <xsl:output method="xml" omit-xml-declaration="yes" /> <!-- end XSL -->
Shared field shorthand	<title>psx-shared/displaytitle</title>
Local Template Call to Global Template Dispatcher	<html psxglobaltemplate="*">

Shared Variants

Snippet and occasionally Page Variant templates can be shared across multiple Content Types. This reduces the redundancy of creating multiple instances of the same template. Our Job Listing Content Type will need an S - Title Link Snippet. This Snippet will display the Job Listing Item's display title as a link back to the Listing's full page.


ices
About Us
Investment Advice
Portfolio Managers
porate Investments
About Us
Careers
Home > About Corporate Investments > Careers
Search
Entry Level Sales Position
s About Corporate Investments Investment Advice Portfolio Managers

ES
ALES - BOSTON MA
ergetic, hard working and goal oriented candidates to join our Content Management
ization. You will learn how to target, cold call and identify sales opportunities in a
environment. Then, teaming with our outstanding outside sales organization you
es efforts and help secure new customers.

d
e to learn and grow
d energized

on about this position contact:

e

Figure 80: Snippet Linking to Page

To accomplish this, we simply need to copy the Variant from an existing Content Type to the Job Listing type.

1. Log into the Content Explorer and navigate to the System Tab
2. Under the Variants section, select [Job Listing].
3. Select [Copy Variant].
4. Locate and select any instance of the [S - Title Link] Variant.
5. Select [Save].

Testing the Shared Snippet Variant

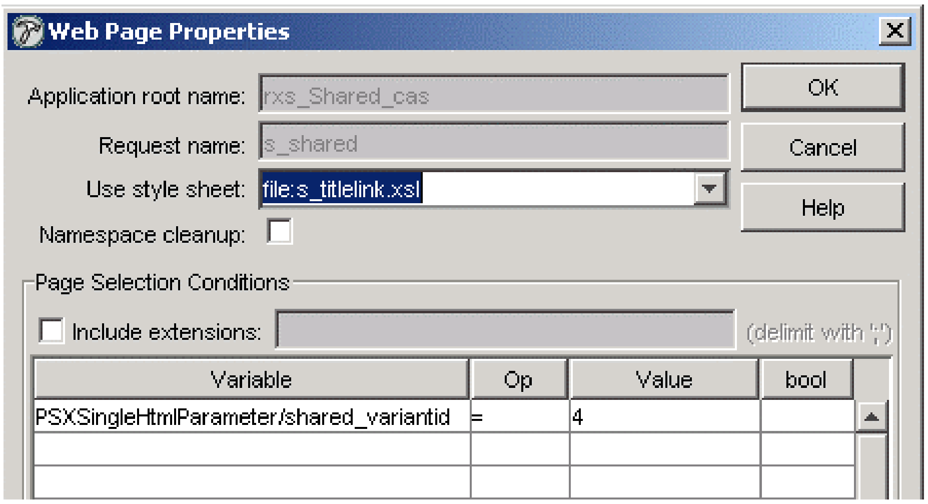
1. Log into the Content Explorer.
2. Locate the Test Folder created recently for testing. This Folder should contain a test Item of the Job Listing Type.
3. Modify the Item if necessary and [Save] the changes.
4. Right click the Item in the Content Explorer and preview the Snippet Variant (Internet Preview > S - Title Link).

5 The Item should be displayed without the Global Template. In fact, is should appear quite plain. Make modifications as necessary to the template and re-split it accordingly.

Understanding the Shared Variant

It is also possible to register a new Shared Variant in the event that the existing ones do not fit the requirements. The important factors to know about Shared Variants are:

- Shared Variants are built in either the rxs_Shared_cas, rxs_SharedImage_cas or rxs_SharedBinary_cas applications. Though this is not a requirement it does support a maintainable environment and is a Best Practice.
- Shared Variants are issued a unique shared_variantid. This ID must make it unique among Shared Variants in an application. This ID must be a part of the Variant URL for requests. The correct usage is shared_variantid={unique id}.
  `../rxs_Shared_cas/s_shared.html?shared_variantid=4`
- At the application level, this shared_variantid must be added as a Page Selection Condition instead of sys_variantid. An example URL for a Shared Variant is provided:



Application root name: rxs_Shared_cas
Request name: s_shared
Use style sheet: file:s_titlelink.xsl
Namespace cleanup: ☐
Include extensions: ☐

Variable	Op	Value	bool
PSXSingleHtmlParameter/shared_variantid	=	4	

Figure 81: Page Selection Condition for shared_variantid

- Data required by the new shared template must be mapped in the shared application's resource. The data presented in a Shared Variant generally resides in any of the shared tables (RX_CT_SHARED, RXS_CT_SHAREDBINARY or RXS_CT_SHAREDIMAGE) or systems tables (CONTENTSTATUS). It may be necessary to add new fields to these tables for particular Variants.
- Shared Variant names need not be unique between Content Types but must be unique within each Type.

C H A P T E R   6

Specialized Implementations

The fundamental concepts behind a specialized implementation can be carried into many different types of application. The Automated Index, for example, is very specialized as described in this chapter, but the concepts underlying how the index is generated can be applied to requirements very different than our example.

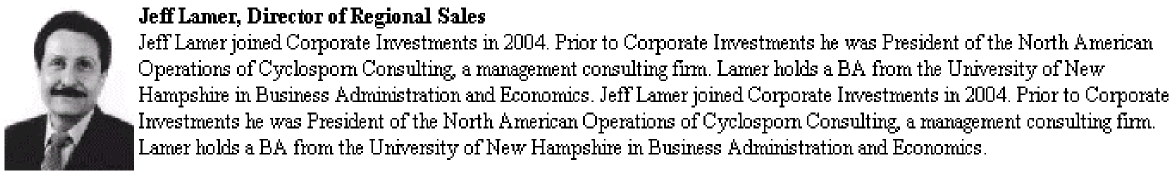
The idea behind these examples is to give you an understanding of the key concepts behind each specialized application so you can use the approach to solve your specific problems. Use Text Extraction, Automated Indexes, WebImageFX, and WebDAV to open the door to unique Rhythmyx Content Management solutions.

Implementing Text Extraction and Non-Text Assembly

The current set of FastForward applications does not meet one of Corporate Investment's requirements. This requirement states the need for a Content Type to collect and store Items of the Bio Type. Items of Bio Content Type will manage executive biographies and consist of the following fields:

- Employee's Name;
- Employee's Title;
- Bio;
- Image.

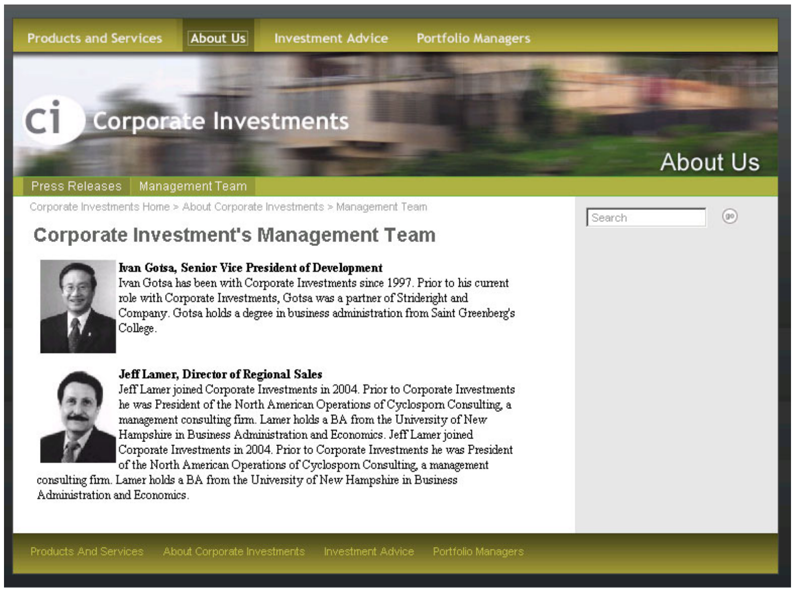
The Bio will initially reside in a Microsoft Word or Adobe PDF file. These files will be uploaded during the creation of the Bio Item and Text Extraction will be used to pull the content from these documents and store the data in a Biobody field. An image will also be uploaded to the Bio Item of the employee. Ultimately, Bio Items will render a Snippet displaying the Employee's Name, Title, some descriptive text, and an image of the Employee.



Jeff Lamer, Director of Regional Sales
Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclosporn Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics. Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclosporn Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics.

Figure 82: S - Bio Listing Snippet

This Content Type will not have a Page Variant. Instead, these Snippets will be manually added to a Generic or similar page to build a list of Employees.



Corporate Investments Home > About Corporate Investments > Management Team

Corporate Investment's Management Team

Ivan Gotsa, Senior Vice President of Development
Ivan Gotsa has been with Corporate Investments since 1997. Prior to his current role with Corporate Investments, Gotsa was a partner of Strideright and Company. Gotsa holds a degree in business administration from Saint Greenberg's College.

Jeff Lamer, Director of Regional Sales
Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclosporn Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics. Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclosporn Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics.

Figure 83: Generic Page with S - Bio Listing Snippet

The creation of this Content Type involves several steps:

1. Creating a Content Editor with Text Extraction and File upload controls.
2. Creating an Assembler.

The Assembler will make use of the existing Shared Image Assembler for the B - Image Variant.

Text Extraction

Rhythmyx's text extraction feature lets you extract text from binary files created in third-party applications (for example, Adobe Acrobat PDFs) to create Rhythmyx Content Items. A Content Editor extension extracts the text and metadata in these files, formats them as text or HTML markup, and inserts the formatted data into Content Item fields. You can attach additional extensions to perform data translations or to insert text into other Content Editor fields.

The text extraction feature uses functionality provided by the Convera RetrievalWare software included with Rhythmyx when it is licensed to include the full text search. In order to use text extraction, you must install the Full Text Search feature when you install Rhythmyx. For information about the Full Text Search feature, see "Searching for Content Using the Full Text Search Engine" in the online *Rhythmyx Content Explorer Help*.

Uploading External Binary Files into Rhythmyx

To apply text extraction to external binary files, add the sys_TextExtraction exit to the Content Editor application that will perform text extraction. The Content Editor must include a field that stores the binary file and a field that stores the data extracted from the file.

In some cases it is most efficient to upload files individually through a file upload control in a Content Editor. In other scenarios, it is most efficient to upload one or more binary files to Rhythmyx by inserting or storing them in a WebDAV-enabled folder. In this case, you must edit the WebDAV configuration file associated with the folder to convert the file into the Rhythmyx Content Type that performs text extraction.

Note: The sys_TextExtraction exit does not limit the size of text extracted. However, a user's browser, Web Server, database, ODBC driver, or sys_EditLive settings may limit the size of content in the field specified to store extracted text. An error message alerts users and prevents them from saving the Content Item if the text extracted exceeds the field's size limitation.

For information about adding a file-upload control to a field in a Content Editor, see the topic "sys_file" in the Workbench help set.

For information about using and configuring WebDAV with Rhythmyx see the document, *Implementing WebDAV in Rhythmyx.*

Creating a Content Editor that Extracts Text

To create a Content Editor that extracts text:

1. Create a Content Editor in the Rhythmyx Workbench.

2 Add or choose fields to store the uploaded file, the extracted data, an extraction error message, and optionally, the file type. Use the following guidelines:

- Create a field to upload and store a file. Use the properties:
  - **Control Name**: sys_file (to store binary files)
  - **Data Type**: binary
  - **Format**: max
- Create a field to store the file type. Note: The sys_FileInfo exit extracts and stores the file type; it requires the name of this field to be the name of the field that stores the uploaded file concatenated with _type (for example, fileupload and fileupload_type). Use the properties:
  - **Control Name**: sys_EditBox
  - **Data Type**: text
  - **Format**: 50 (This value is large enough to store long Mime Type names).
- (Optional) Create a field to hold error text. Use the properties:
  - **Control Name**: sys_EditBox
  - **Data Type**: text
  - **Format**: 255 (This value is large enough to store long error messages.)
- Create a field to hold the extracted text:

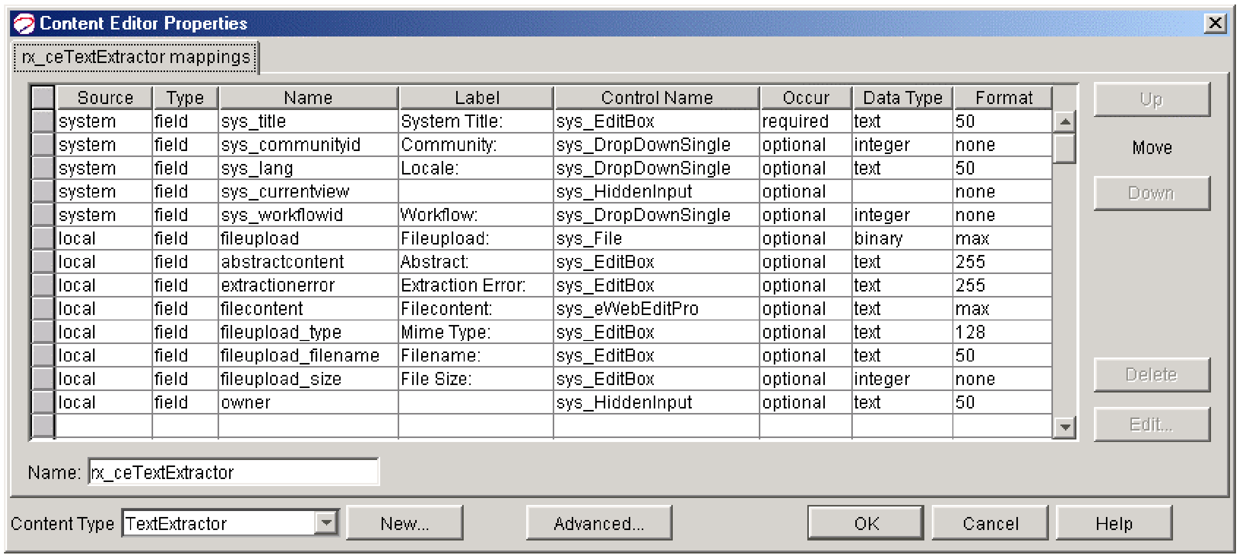
If you want the extracted text to include HTML formatting, use the properties:

  - **Control Name**: sys_EditLive (You can also use sys_TextArea. You may encounter size limitations if you use sys_EditBox.)
  - **Data Type**: text
  - **Format**: max (You can specify a numeric size if you add an exit that truncates the data to a specific size.)

If you want the extracted text to be formatted as text, use the properties:

- **Control Name**: sys_TextArea
- **Data Type**: text
- **Format**: max (You can specify a numeric size if you add an exit that truncates the data to a specific size.)

For examples of how output formats and Content Editor controls affect the appearance of the text in your Content Editor, see ***Displaying Extracted Text in a Content Editor*** (on page 109).



Source	Type	Name	Label	Control Name	Occur	Data Type	Format
system	field	sys_title	System Title:	sys_EditBox	required	text	50
system	field	sys_communityid	Community:	sys_DropDownSingle	optional	integer	none
system	field	sys_lang	Locale:	sys_DropDownSingle	optional	text	50
system	field	sys_currentview		sys_HiddenInput	optional		none
system	field	sys_workflowid	Workflow:	sys_DropDownSingle	optional	integer	none
local	field	fileupload	Fileupload:	sys_File	optional	binary	max
local	field	abstractcontent	Abstract:	sys_EditBox	optional	text	255
local	field	extractionerror	Extraction Error:	sys_EditBox	optional	text	255
local	field	filecontent	Filecontent:	sys_eWebEditPro	optional	text	max
local	field	fileupload_type	Mime Type:	sys_EditBox	optional	text	128
local	field	fileupload_filename	Filename:	sys_EditBox	optional	text	50
local	field	fileupload_size	File Size:	sys_EditBox	optional	integer	none
local	field	owner		sys_HiddenInput	optional	text	50

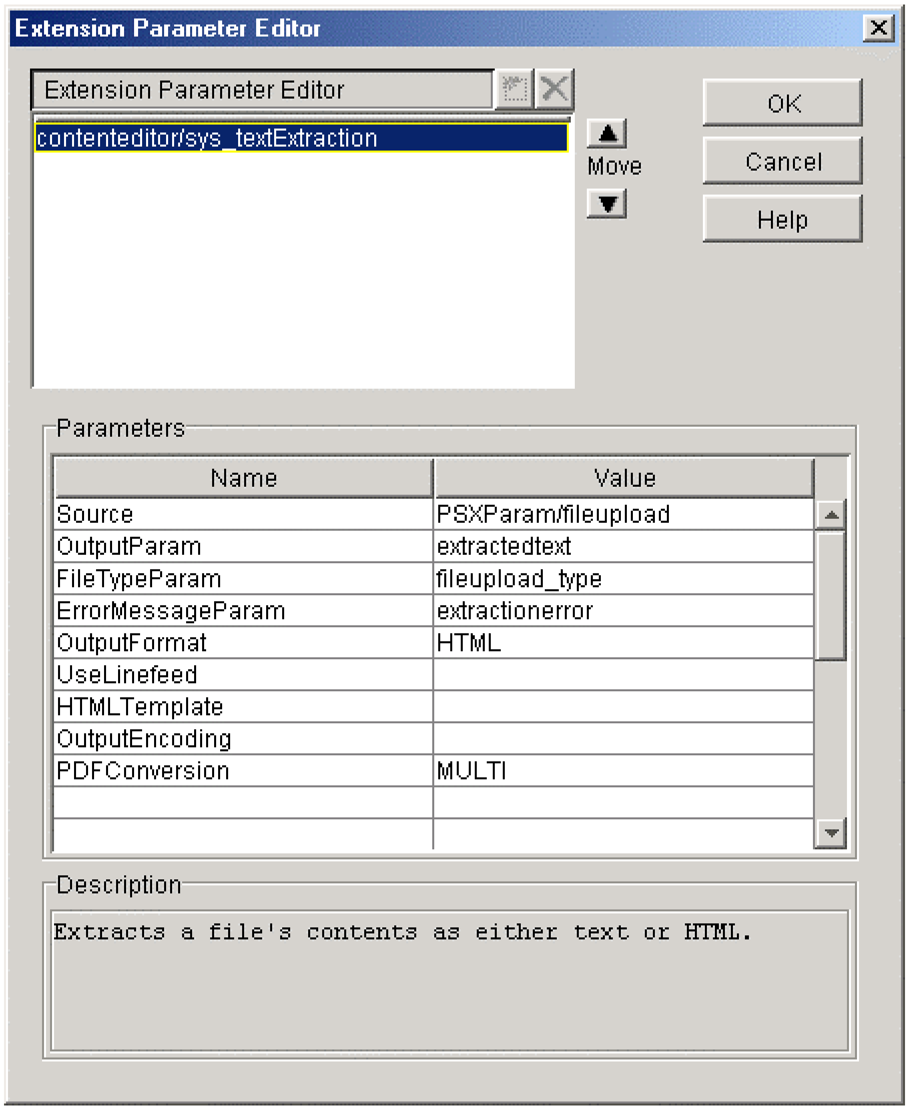
Name: rx_ceTextExtractor

Content Type: TextExtractor

Figure 84: Content Editor Properties for an example Text Extraction Content Type

3. Click [**Advanced**] to open the Content Editor Settings dialog.
4. In the Item Input Translation tab:

- Choose the sys_TextExtraction extension. Configure the extension parameters for the specific text extraction.



contenteditor/sys_textExtraction

Name	Value
Source	PSXParam/fileupload
OutputParam	extractedtext
FileTypeParam	fileupload_type
ErrorMessageParam	extractionerror
OutputFormat	HTML
UseLinefeed	
HTMLTemplate	
OutputEncoding	
PDFConversion	MULTI

Extracts a file's contents as either text or HTML.

Figure 85: Example configuration of the sys_TextExtraction exit

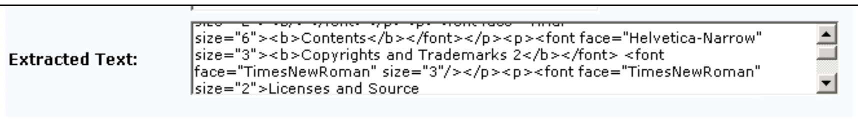
These parameter values correspond to the fields in the Content Editor shown in the graphic for Step 2.

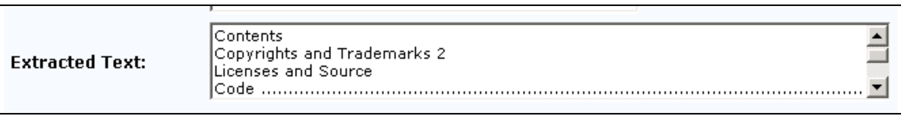
Optionally, choose additional extensions to translate data in the uploaded file before it is inserted into the Rhythmyx Content Item. When you add the sys_File control, Rhythmyx automatically includes the sys_FileInfo exit. This exit inserts the filename, file size, and file extension into Content Editor fields.

Displaying Extracted Text in a Content Editor

The way the extracted text appears in a Content Editor depends on the type of control that stores the extracted field and the output format specified in the sys_TextExtraction exit. The following table shows the same extracted text displayed using three different combinations of Content Editor controls and output formats.

Control Name for Extracted Text	Output Format for Extracted Text	Extracted Text in Content Editor
sys_EditLive	HTML	

sys_TextArea	HTML	 Extracted Text: size="6"><b>Contents</b></font></p><p><font face="Helvetica-Narrow"
		size="3"><b>Copyrights and Trademarks 2</b></font> <font
		face="TimesNewRoman" size="3"/></p><p><font face="TimesNewRoman"
		size="2">Licenses and Source

sys_TextArea	TEXT	 Extracted Text: Contents Copyrights and Trademarks 2 Licenses and Source Code ....

Creating the Bio Content Editor

We will begin by creating the Bio Content Editor. This CE will be based on the specifications described in the Statement of Work found earlier in this manual.

1 Open the Workbench.

2 Open a new application.

3 Create a new Content Editor Resource (Insert > Content Editor Resource).

4 Select the sys_Default template.

5 Build the following fields into the new Content Editor.

	Field/Label	Field Name	Control	Occur	Type(Format)
sys	System Title:	sys_title	sys_EditBox	Required	50
sys	Start Date:	sys_contentstartdate	sys_CalendarSimple	Required	none
sys	Expiration Date:	sys_contentexpirydate	sys_CalendarSimple	Optional	none
sh	Employee Name:	shared/displaytitle	sys_EditBox	Required	512
loc	Employee's Title:	employeetitle	sys_EditBox	Optional	50

	Field/Label	Field Name	Control	Occur	Type(Format)
loc	Bio Source Field:	bio	sys_File	Optional	max
loc	Bio Body:	biobody	sys_TextArea	Optional	max
sh	Image::	sharedimage/img1	sys_File	Required	max
sh	Image Filename:	sharedimage/img1_filename	sys_EditBox	Required	512
loc	Error Text:	errortext	sys_TextArea	Optional	max
sys	Locale:	sys_lang	sys_DropDownSingle	Optional	50
sys	Community:	sys_communityid	sys_DropDownSingle	Optional	none
sys	Workflow:	sys_workflowid	sys_DropDownSingle	Optional	none
sys		sys_currentview	sys_HiddenInput	Optional	none
sys		sharedimage/img1_ext	sys_HiddenInput	Optional	50
sh		sharedimage/img1_height	sys_HiddenInput	Optional	none
sh		sharedimage/img1_size	sys_HiddenInput	Optional	none
sh		sharedimage/img1_type	sys_HiddenInput	Optional	50
sh		sharedimage/img1_width	sys_HiddenInput	Optional	none

6 Name the local table EI_BIO_CT.

7 Register a new Content Type (New) named Bio.

8 Press the [Advanced] button and define the following:

- General - Allow fields to be searched and Enable Related Content
- Workflow - Standard Workflow and Allows Items to Enter Any Workflow
- Input Item Translation - sys_textExtraction as follows:

  Source - PSXParam/bio

  OutputParam - biobody

  ErrorMessageParam - errortext

  OutputFormat - TEXT

  PDFConversion - SINGLE

In addition, add a conditional to this Exit as follows:

Conditional - PSXParam/bio IS NOT NULL

Note: This pre-exit extracts the text and metadata in a binary file uploaded to a Rhythmyx Content Editor and inserts the extracted data into a Content Editor field (or fields). The exit formats the extracted text as text or HTML markup.

- Item Output Translation - N/A
- Item Validation - N/A

9 Press [OK] to close the Content Editor Table Editor.

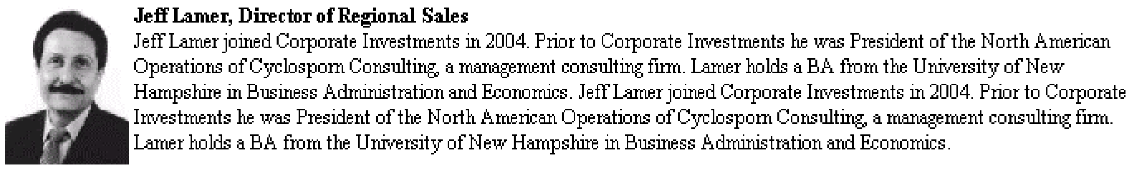
10 Right click the Content Editor Resource and select [Request Properties]. Rename the Resource, Bio.

11 Save (Application > Save) the application.

12 Start (Application > Start) the application.

Creating the Bio Assembly Application

Once we can create Items of the Bio Type, we need to generate the Snippets. These Snippets will be manually placed on a Generic page or similar Content Item. The S - BioListing Snippet will require two separate resources to render the output. A standard Query Resource will be used to render the text and will call the shared Non-Text Resource, Image, in the rxs_SharedImage_cas application. This Shared Resource will be responsible for the generation of the Image in the Snippet.



Jeff Lamer, Director of Regional Sales
Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclosporn Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics. Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclosporn Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics.

Figure 86: S - Bio Listing Snippet

Our tasks will be as follows:

- Create the Snippet HTML Template;
- Generate the Snippet Assembler;
- Register the S - Bio Listing Snippet as a Variant of the Bio Content Type.

Creating the s_BioListing.html template

The s_BioListing.html template is going to be very similar to an existing Snippet, S - Title, Callout, Link.

1 Locate a copy of the original template, s_titlecalloutlink.html (\Rhythmyx\rxs_Shared_cas\src) and save a copy of it locally as s_BioListing.html.

2 Open the file and modify it to match the sample below:

```
<!-- begin XSL -->
<xsl:output method="xml" omit-xml-declaration="yes" />
<!-- end XSL -->
<html>
   <head>
      <title>psx-shared/displaytitle</title>
      <link rel="stylesheet" psx-href="$rxs_css/rxs_styles.css"
         href="/web_resources/css/rxs_styles.css" type="text/
         css" />
   </head>
   <body>
      <div class="lead_snippet">
         <img src="psx-imagelink" alt="psx-shared/displaytitle"
            width="75" />
         <h1><span psxedit="displaytitle" psxshow="psx-shared/
```

```
               displaytitle">Employee Name goes here</span>
            <span>,&nbsp;</span>
            <span psxedit="employeetitle"
                psxshow="psx-employeetitle">Employee Title goes
               here</span></h1>
            <p>
               <span psxedit="body" psxshow="psx-body">Body of bio
                  goes here</span>
            </p>
         </div>
      </body>
   </html>
```

Note the following: shared/displaytitle will be the employee's name. This will be used for both the page title and as content within the page. The image source will be mapped to the element imagelink. This will be defined when we create the Snippet Assembler. The remaining values; employeetitle and body will provide the bulk of the content for the Snippet. These values, too, will be mapped when we create the Snippet Assembler.

3 Save the file.

Generating the Snippet Assembler

1 Open the Workbench and create a new empty application.

2 Open the original ci_Bio_ce application.

3 Right click and copy the [Source XML] from the Bio Content Editor Resource.

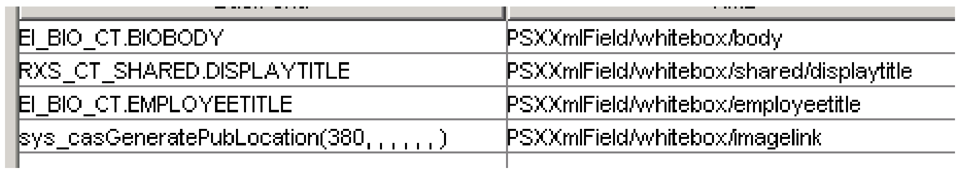
4 Switch over to the new empty application, select the workspace, right click and select [Paste as Snippet Assembler].

5 Save (Application > Save) the application as ci_Bio_cas.

6 Open the new Resource's properties and open the Mapper.

7 Map the following values:

- CI_BIO_CT.BIOBODY - PSXmlField/whitebox/body
- RXS_CT_SHARED.DISPLAYTITLE - PSXmlField/whitebox/shared/displaytitle
- sys_casGeneratePubLocation (variantid = VID of B - Image Variant of Image) - PSXmlField/whitebox/imagelink



EI_BIO_CT.BIOBODY	PSXXmlField/whitebox/body
RXS_CT_SHARED.DISPLAYTITLE	PSXXmlField/whitebox/shared/displaytitle
EI_BIO_CT.EMPLOYEETITLE	PSXXmlField/whitebox/employeetitle
sys_casGeneratePubLocation(380, , , , , ,)	PSXXmlField/whitebox/imagelink

Figure 87: Bio Snippet Mappings

8 Click [OK] to close the Mapper.

9 Save (Application > Save) the application.

10 Select the Files navigation tab in the Workbench and locate the s_BioListing.html template.

11 Drag and drop the template [As a StyleSheet] allowing for updates to the Resource's DTD.

12 Save (Application > Save) the application.

Register the S - Bio Listing Snippet

1 Log into the Content Explorer and navigate to the System Tab.

2 Under the Variants list, select Bio.

3 Click [Add a New Variant].

4 Register the new Bio Snippet as follows:

- Name - S - Bio Listing
- Description - Renders Image, Name, Title, and extracted text.
- Style Sheet - s_BioListing.xsl
- URL - ../ci_Bio_cas/snippet_assembler.html
- Location Prefix -
- Location Suffix -
- Output Form - Snippet
- Active Assembly Format - Normal
- Publish When - Never

5 [Save] the new Registration.

6 Note the Variant ID issued to the Snippet.

Assign Selection Condition to s_BioListing.xsl Page

1 In the Workbench, open the s_BioListing.xsl Page properties in the ci_Bio_cas application.

2 Define the following Selection Condition:

- PSXSingleHTMLParameter/sys_variantid = (Variant ID of the S - Bio Listing Snippet)

3 Press [OK] to close the Page Properties dialog.

4 Save (Application > Save) the application.

Creating a Non-Text Assembler

Our ci_Bio_cas application contains a Query Resource, snippet_assembler, which contains a reference to the Variant, B - Image. This Snippet is generated by the Non-Text Resource, Image, in the rxs_SharedImage_cas. This special type of Assembly Resource retrieves the binary data from the Rhythmyx repository and generates the Binary File. Our call to Image will yield a src value targeting the rendered image.

A Non-Text Resource is created as follows:

1 Open the Workbench.

2 Create a new empty Application.

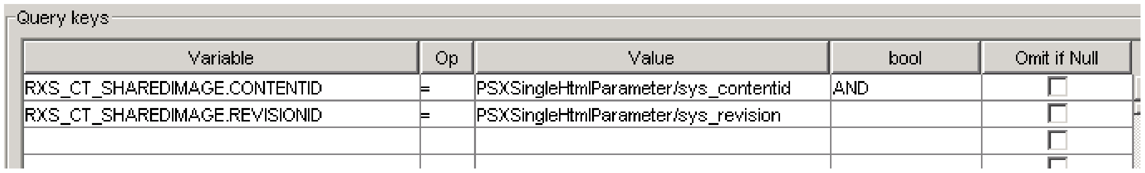
3 Insert a Non-Text Resource into the application by locating the table through the Workbench Data Tab where the Image is stored. Our example will use the table rxs_SharedImage_cas.

4 Open the Resource's [Request Properties] and rename the object. We will call this object, *Image*.

5 Press [OK] to close the Request Properties dialog.

6 Open the object's [Properties] dialog. Most of the initial database data information will be pre-defined. Select the column where the binary is stored. For this table the full size image is located in the IMG1 column.

7 Define query keys based on sys_contentid and sys_revision (as with normal Assemblers).

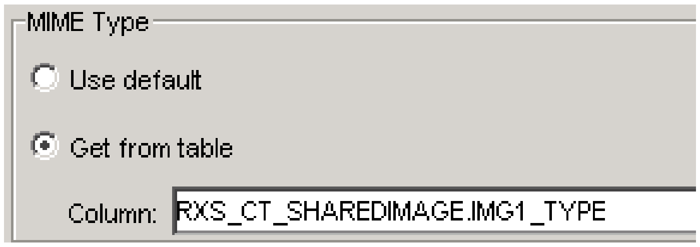


Query keys

Variable	Op	Value	bool	Omit if Null
RXS_CT_SHAREDIMAGE.CONTENTID	=	PSXSingleHtmlParameter/sys_contentid	AND	☐
RXS_CT_SHAREDIMAGE.REVISIONID	=	PSXSingleHtmlParameter/sys_revision		☐
				☐

Figure 88: Query Keys for Non-Text Resources

8 The MIME type for the binary is stored in the IMG_TYPE column. Provide this information to the Resource.



MIME Type

○ Use default

◉ Get from table

Column: RXS_CT_SHAREDIMAGE.IMG1_TYPE

Figure 89: Defining MIME Type for a Non-Text Resource

9 Press [OK] to save the Properties.

10 Save (Application > Save) the application.

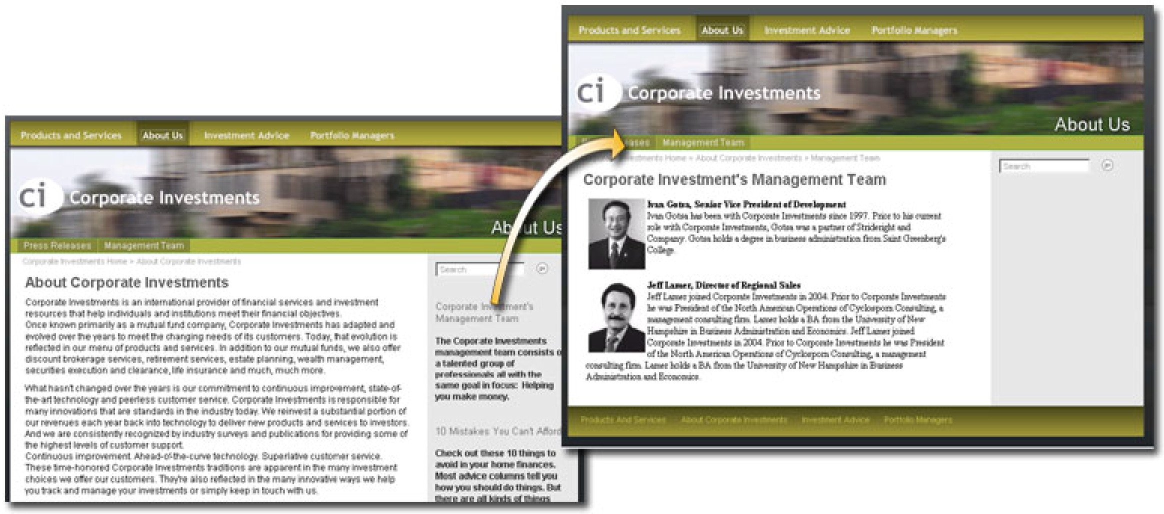
Register the Binary Variant

When registering the Non-Text Variant for a Content Type use the following configuration:

- Name - Standard naming convention prefaces the Variant name with B - to denote the Variant as Binary
- Style Sheet - {leave this blank] (we will not need to transform the binary)
- URL - ../{Application Name}/{Resource Name without suffix}
- Output Form - Page (For it to be published)
- Active Assembly Format - Non-Text (Since there is no Assembly possible on an Image)
- Publish When - Default

Testing the Bio Content Type

The template for the Snippet, the Content Editor to gather content, and the Assembler to generate the output HTML is complete. Putting all of these pieces together will allow our content contributors to create pages displaying images, extracted data from Microsoft Word or PDF documents, and additional data. Let's set the scenario for the creation of a Management Team page that will be linked to on our About Us page.


Products and Services
About Us
Investment Advice
Portfolio Managers
Corporate Investments
About Us
Press Releases
Management Team
Corporate Investments Home > About Corporate Investments
Search
Corporate Investment's Management Team
10 Mistakes You Can't Afford
Corporate Investments Home > About Corporate Investments > Management Team
Products And Services
About Corporate Investments
Investment Advice
Portfolio Managers

About Corporate Investments

Corporate Investments is an international provider of financial services and investment resources that help individuals and institutions meet their financial objectives.
Once known primarily as a mutual fund company, Corporate Investments has adapted and evolved over the years to meet the changing needs of its customers. Today, that evolution is reflected in our menu of products and services. In addition to our mutual funds, we also offer discount brokerage services, retirement services, estate planning, wealth management, securities execution and clearance, life insurance and much, much more.

What hasn't changed over the years is our commitment to continuous improvement, state-of-the-art technology and peerless customer service. Corporate Investments is responsible for many innovations that are standards in the industry today. We reinvest a substantial portion of our revenues each year back into technology to deliver new products and services to investors. And we are consistently recognized by industry surveys and publications for providing some of the highest levels of customer support.
Continuous improvement. Ahead-of-the-curve technology. Superlative customer service. These time-honored Corporate Investments traditions are apparent in the many investment choices we offer our customers. They're also reflected in the many innovative ways we help you track and manage your investments or simply keep in touch with us.

The Coporate Investments management team consists of a talented group of professionals all with the same goal in focus: Helping you make money.

Check out these 10 things to avoid in your home finances. Most advice columns tell you how you should do things. But there are all kinds of things

Corporate Investment's Management Team

Ivan Gotsa, Senior Vice President of Development
Ivan Gotsa has been with Corporate Investments since 1997. Prior to his current role with Corporate Investments, Gotsa was a partner of Strideright and Company. Gotsa holds a degree in business administration from Saint Greenberg's College.

Jeff Lamer, Director of Regional Sales
Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclospora Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics. Jeff Lamer joined Corporate Investments in 2004. Prior to Corporate Investments he was President of the North American Operations of Cyclospora Consulting, a management consulting firm. Lamer holds a BA from the University of New Hampshire in Business Administration and Economics.

Figure 90: About Us Linking to Management Team

For this activity you will need the following:

- Access to the new Bio Content Type
- Images for each Management Team member. The image will be scaled down by our template to 75 pixels wide.
- Individual Microsoft Word documents for each team member. This Word file will contain the biography data for the particular person.

Create the Bio Content Items

1. Log into the Content Explorer.
2. Navigate to the Sites > Corporate Investments > About Us Site Folder.
3. Create a new Item (New Item > Bio) of the Bio type. Our example will use the following information:
    - System Title - Jeff Lamer's Bio
    - Employee Name - Jeff Lamer
    - Employee's Title - Director of Sales
    - Bio Source File - {upload the Word file with Jeff's summary}
    - Bio Body - {This field will be populated with the contents of the Word file attached in the Bio Source File field during insert}

- Image - {Upload the image of Jeff Lamer}
- Image File Name - Will be automatically populated with the name of the file uploaded to the Image field.
- Error Text - Will be automatically populated with any error messages received during text extraction.

4 [Insert] the Item.

5 Create additional Items as necessary.

6 Preview the Bio Items (S - Bio Listing). If necessary, [Edit] the Bio Items to produce the desired Snippet output.

Create a Generic Page for the Bio Snippets

1 The Bio Listing Snippets must be assembled on a Page Variant for display on our site. In the About us folder, create an Item of the Generic Type. Name the page Corporate Investment's Management Team. Provide some brief Callout text for use when linking to the page. Leave the Body field blank for now or enter a short amount of text introducing the entire team.

2 [Insert] the Generic Item and close the form.

Assemble the Management Team Page

With both the parent Generic Page and the child Bio Items created, we can now add the Bio Snippets to the Generic page.

1 Right click the Generic Management Team Item and select [Active Assembly Table Editor].

2 Add the S - Bio Listing Snippets of the Bio Content Type to the List Slot.

Note: If the Bio Content Type is not available during this Item search, be sure the S - Bio Listing Snippets are defined as Allowed Content for the List Slot. See the topic, Slots, in the Workbench help set for more information.

3 Close the Table Editor dialog.

4 Preview the Management Team Generic page to assure the content is rendering as expected.

Link the About Us page to the new Management Team page

1 In the same About Us folder, locate the Generic page Page - About Corporate Investments.

2 Right click the About page Item and open the [Active Assembly Table Editor].

3 Add the new Management Team Generic page to the Sidebar Slot as related Content. Use the S - Title Callout Link Snippet to make the link match existing Related Content in the Slot.

4 Close the Table Editor.

5 Select and preview the About Us Page. The link to the Management Team page should now populate the Sidebar Slot. Edit the order of the links in the Slot if necessary.

Continuous Conversion Example

In continuous conversion, Rhythmyx publishes content that a user has created in another application and wants to continue modifying in the other application. When the file is downloaded to Rhythmyx the body data is extracted as text and inserted into a Content Editor field. However, the file contents are not modified in Rhythmyx; they are always modified in the third-party application. After modification, Rhythmyx reloads the files and updates the original Content Items.

In our example, a Marketing Department maintains a library of PDF documents for customers. On the company Web Site, the Department wants to include a list of the documents that links to their contents. When the content of a PDF changes, the Department uploads the updated PDF to Rhythmyx again. Rhythmyx automatically updates the content of the original Content Item.

As an implementer in the Marketing Department, you use the following procedure to implement continuous conversion of the PDFs:

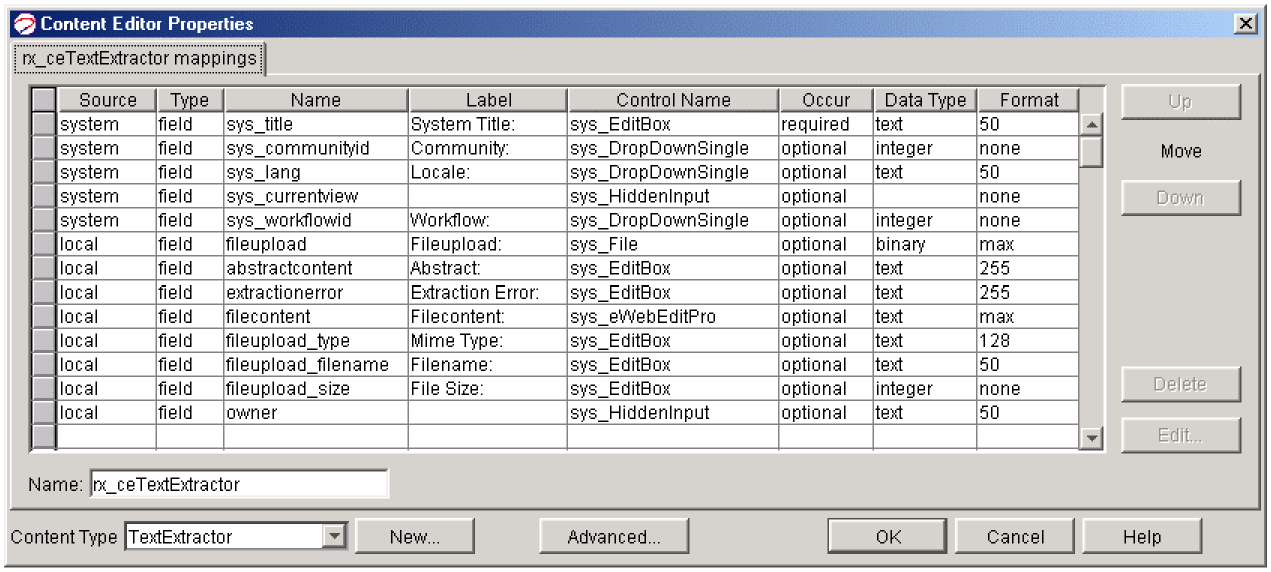
1 In the Rhythmyx Workbench, create a Content Editor for a Content Type to store the uploaded PDF files. It includes fields for the uploaded file, the extracted text, the file type, and an error message if file upload fails. The procedure for creating the Content Editor is as follows:

   a) Choose the sys_Default.xml template and name the new resource rx_TextExtract.

   b) Double-click the template to open the Content Editor Properties dialog and add the fields:

      - fileupload – The sys_TextExtraction exit will use *fileupload* to store the uploaded file.
        Properties: Control Name=sys_file, Data Type=binary, Format=max.
      - fileupload_type – The sys_TextExtraction exit and WebDAV will use *fileupload_type* to store the mime type.
        Properties: Control Name=sys_EditBox, Data Type=text, Format=50.
      - extractionnerror - The sys_TextExtraction exit will use *extractionerror* to store the text of the first error encountered during extraction.
        Properties: Control Name=sys_editBox, Data Type=text, Format=255.
      - filecontent - The sys_TextExtraction exit will use *filecontent* to store the extracted text.
        Properties: Control Name=sys_EditLive, Data Type=text, Format=max.
      - owner - WebDAV will use *owner* to store the user that has the file locked.
        Properties: Name=sys_HiddenInput, Data Type=text, Format=50.
      - fileupload_size – WebDAV will use *fileupload_size* to hold the file size.
        Properties: Control Name=sys_EditBox, Data Type – integer, Format=none.
      - abstractcontent – This field can display an abstract of the Content Item; it is not required by sys_TextExtraction or WebDAV. Use either a custom input translation exit to fill it or allow the user to fill it through the Content Editor.
        Properties: Control Name=sys_EditBox, Data Type=text, Format=255.
      - fileupload_name – This field displays the uploaded file name. Include it for your own reference. It is populated by the sys_file control.
        Properties: Control Name=sys_EditBox, Data Type=text, Format=50.

2 Click [**New**] and specify the Content Type TextExtractor.



Source	Type	Name	Label	Control Name	Occur	Data Type	Format
system	field	sys_title	System Title:	sys_EditBox	required	text	50
system	field	sys_communityid	Community:	sys_DropDownSingle	optional	integer	none
system	field	sys_lang	Locale:	sys_DropDownSingle	optional	text	50
system	field	sys_currentview		sys_HiddenInput	optional		none
system	field	sys_workflowid	Workflow:	sys_DropDownSingle	optional	integer	none
local	field	fileupload	Fileupload:	sys_File	optional	binary	max
local	field	abstractcontent	Abstract:	sys_EditBox	optional	text	255
local	field	extractionerror	Extraction Error:	sys_EditBox	optional	text	255
local	field	filecontent	Filecontent:	sys_eWebEditPro	optional	text	max
local	field	fileupload_type	Mime Type:	sys_EditBox	optional	text	128
local	field	fileupload_filename	Filename:	sys_EditBox	optional	text	50
local	field	fileupload_size	File Size:	sys_EditBox	optional	integer	none
local	field	owner		sys_HiddenInput	optional	text	50

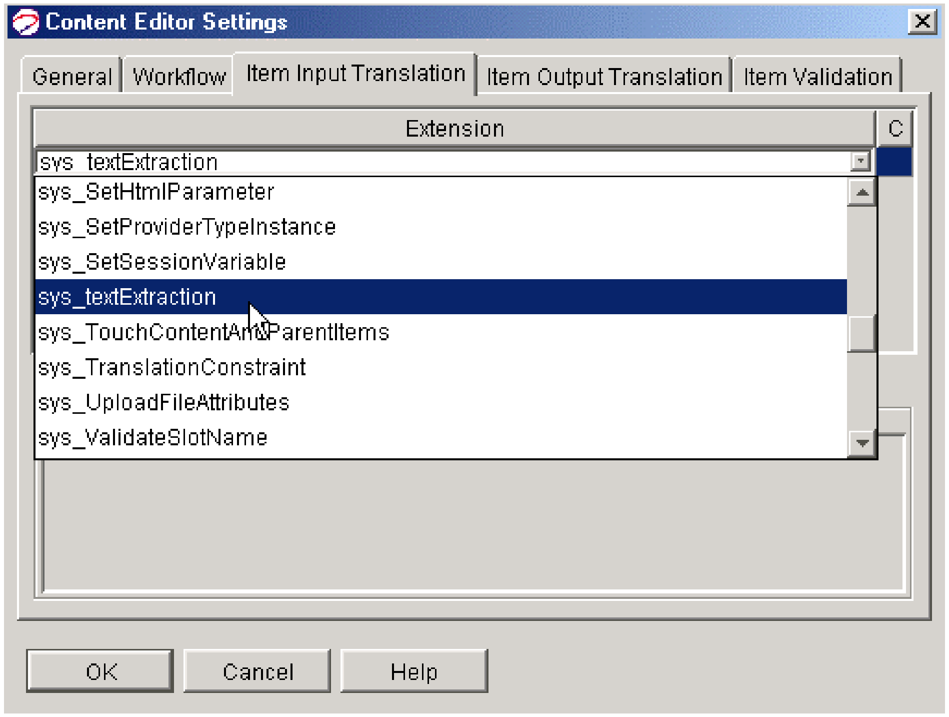
Name: rx_ceTextExtractor

Content Type TextExtractor

Figure 91: Content Editor Properties for Text Extractor

c) Click [**Advanced**] to open the Content Editor Settings dialog.

d) On the Item Input Translation tab, choose the sys_textExtraction extension.

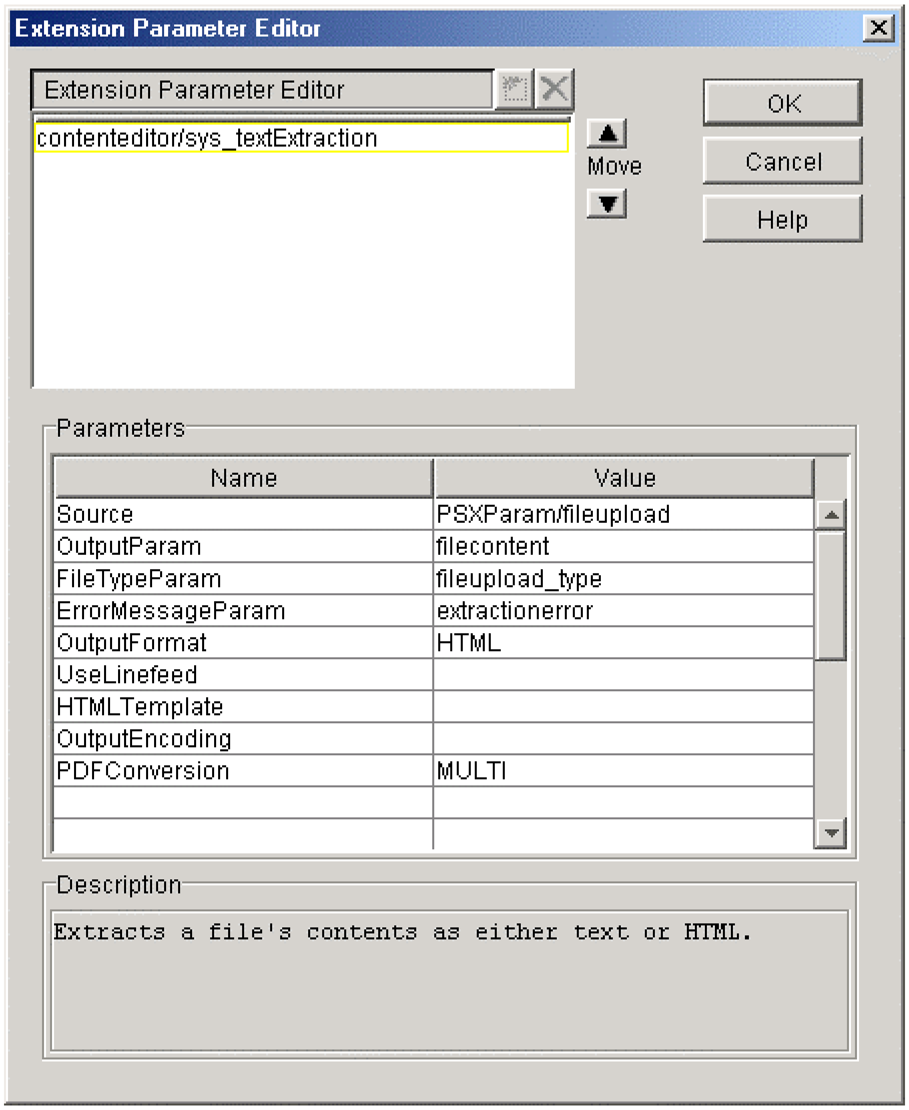


sys_textExtraction

sys_SetHtmlParameter
sys_SetProviderTypeInstance
sys_SetSessionVariable
sys_textExtraction
sys_TouchContentAndParentItems
sys_TranslationConstraint
sys_UploadFileAttributes
sys_ValidateSlotName

Figure 92: Choosing sys_textExtraction in the Content Editor Settings dialog

Set the following values for the parameters:



contenteditor/sys_textExtraction

Name	Value
Source	PSXParam/fileupload
OutputParam	filecontent
FileTypeParam	fileupload_type
ErrorMessageParam	extractionerror
OutputFormat	HTML
UseLinefeed	
HTMLTemplate	
OutputEncoding	
PDFConversion	MULTI

```
Extracts a file's contents as either text or HTML.
```

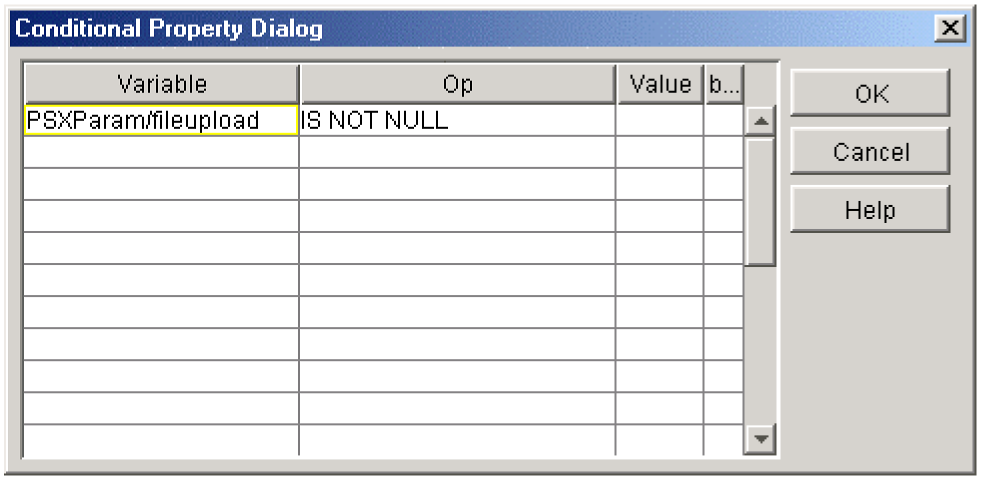
Figure 93: Parameter Settings for sys_textExtraction

Name	Value	Description
Source	PSXParam/fileupload	The field in the Content Editor that holds the uploaded file. May be expressed as any value type.
OutputParam	filecontent	The field in the Content Editor that stores the extracted data.
FileTypeParam	fileupload_type	The field in the Content Editor that stores the file type.

Name	Value	Description
ErrorMessageParam	extractionerror	The field in the Content Editor that stores the error message if data does not extract. By inserting the error message into a field in the Content Editor, you let the text extraction process convert all files, even if some do not convert properly.
OutputFormat	HTML	Specifies that converted data is formatted as HTML. Since the converted data is inserted into an HTML editor control, HTML is necessary so the control can format the data as closely as possible to its original appearance.
UseLineFeed	---	Not used because OutputFormat is HTML.
HTMLTemplate	---	Not used because standard HTML template is not overridden.
OutputEncoding	---	Not specified because default character encoding is used.
PDFConversion	Multi	multi is specified so OutputFormat can be HTML

Add a condition to the exit specifying that the exit only runs when a file is uploaded. If you do not include this condition, if a user edits metadata fields and saves the Content Item, the sys_textExtraction exit attempts to run and results in an error.

The condition specifies that the field that stores the uploaded file IS NOT NULL:



Variable	Op	Value	b...
PSXParam/fileupload	IS NOT NULL		

Figure 94: Conditional setting for sys_textExtraction

e) At this point, you can add custom input translation exits before or after the sys_textExtraction exit to perform additional processing. For example, you may add an exit to parse the first sentence from the extracted data and insert the parsed text into the abstractcontent field.

f) Save the Content Editor as rx_ceTextExtractor and close it.

NOTE: The new Content Type is automatically registered in Rhythmyx when you save the Content Editor, but you must associate it with the Communities that you want to have access to it.

3 Define a servlet named *Marketing PDFS* in the WebDAV deployment descriptor, <Rhythmyx root>/AppServer/webapps/rxwebdav/WEB-INF/web.xml:

```
  <servlet>
    <servlet-name>Marketing PDFs</servlet-name>
    <display-name>Rhythmyx WebDAV Router</display-name>
    <description>Rhythmyx WebDAV Router</description>
    <servlet-class>com.percussion.webdav.PSWebdavServlet</servlet-
class>
    <init-param>
      <param-name>RxWebDAVConfig</param-name>
      <param-value>/RxWebdavConfig.xml</param-value>
      <description>The webdav configuration file path, which is
         relative to the current web application</description>
    </init-param>
  </servlet>
```

Also, add a servlet-mapping element for the new servlet in the WebDAV deployment descriptor:

```
   <servlet-mapping>
       <servlet-name>Marketing PDFs</servlet-name>
       <url-pattern>/Marketing PDFs/*</url-pattern>
   </servlet-mapping>
```

4 Edit the WebDAV configuration file, <Rhythmyx root>\AppServer\webapps\rxwebdav\RxWebdavConfig.xml, as follows. Only a default Content Type is necessary because all of the input files will be PDFs and they will all be converted to TextExtract Content Types.

```
<?xml version="1.0" encoding="UTF-8"?>
<PSXWebdavConfigDef root="//Folders/Marketing PDFs"
   communityname="default" communityid="10" locale="en-us">
   <PSXWebdavContentType id="301" name="TextExtractor"
      contentfield="fileupload" ownerfield="owner" default="true">
      <PropertyMap>
         <PSXPropertyFieldNameMapping name="getcontenttype">
            <FieldName>fileupload_type</FieldName>
         </PSXPropertyFieldNameMapping>
               <PSXPropertyFieldNameMapping
                  name="getcontentlength">
            <FieldName>fileupload_size</FieldName>
         </PSXPropertyFieldNameMapping>
      </PropertyMap>
   </PSXWebdavContentType>
   </PSXWebdavConfigDef>
```

5 Set up WebDAV-enabled folders in Rhythmyx Content Explorer and Windows Explorer for storing the PDFs.

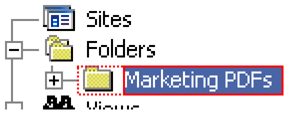


Figure 95: WebDAV-enabled Folder in Content Explorer

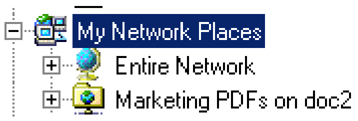
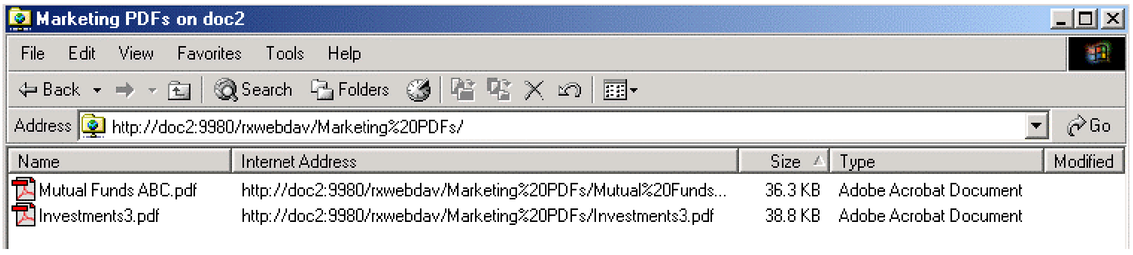


Figure 96: Web Folder in Windows Explorer

6 Copy the PDFs into the WebDAV-enabled folder on Windows Explorer.

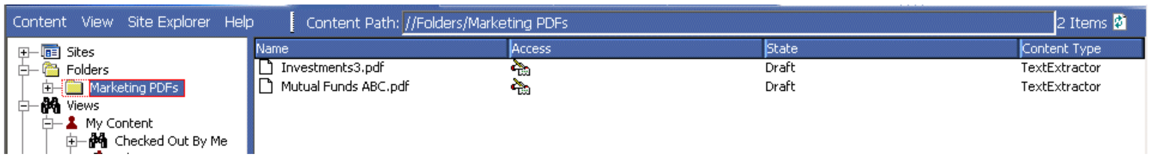


http://doc2:9980/rxwebdav/Marketing%20PDFs/

Name	Internet Address	Size	Type	Modified
Mutual Funds ABC.pdf	http://doc2:9980/rxwebdav/Marketing%20PDFs/Mutual%20Funds...	36.3 KB	Adobe Acrobat Document	
Investments3.pdf	http://doc2:9980/rxwebdav/Marketing%20PDFs/Investments3.pdf	38.8 KB	Adobe Acrobat Document	

Figure 97: PDF files in Web Folder

They now exist in the WebDAV-enabled folders in Content Explorer, and Rhythmyx automatically converts them into TextExtractor Content Items.

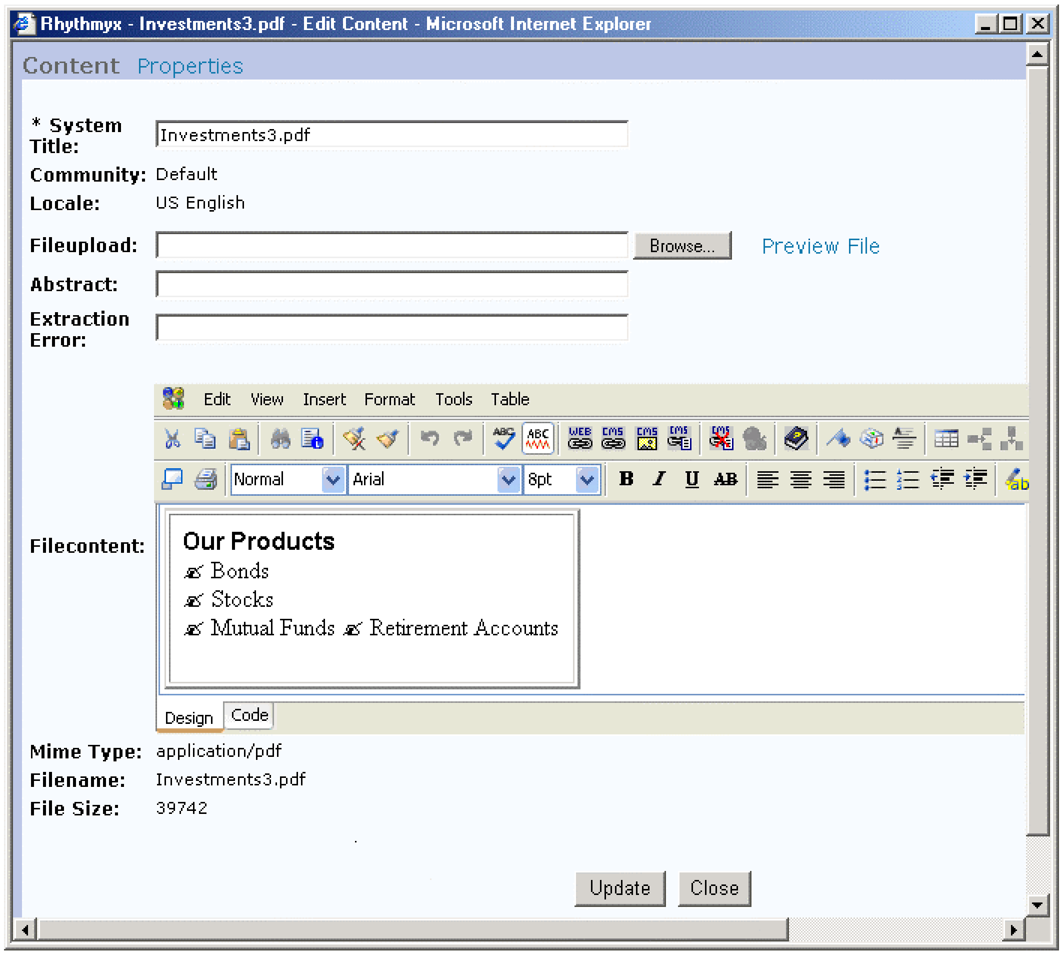


Content Path: //Folders/Marketing PDFs 2 Items

Name	Access	State	Content Type
Investments3.pdf		Draft	TextExtractor
Mutual Funds ABC.pdf		Draft	TextExtractor

Figure 98: Content Items extracted from the PDF files

An opened TextExtractor Content Item appears as:

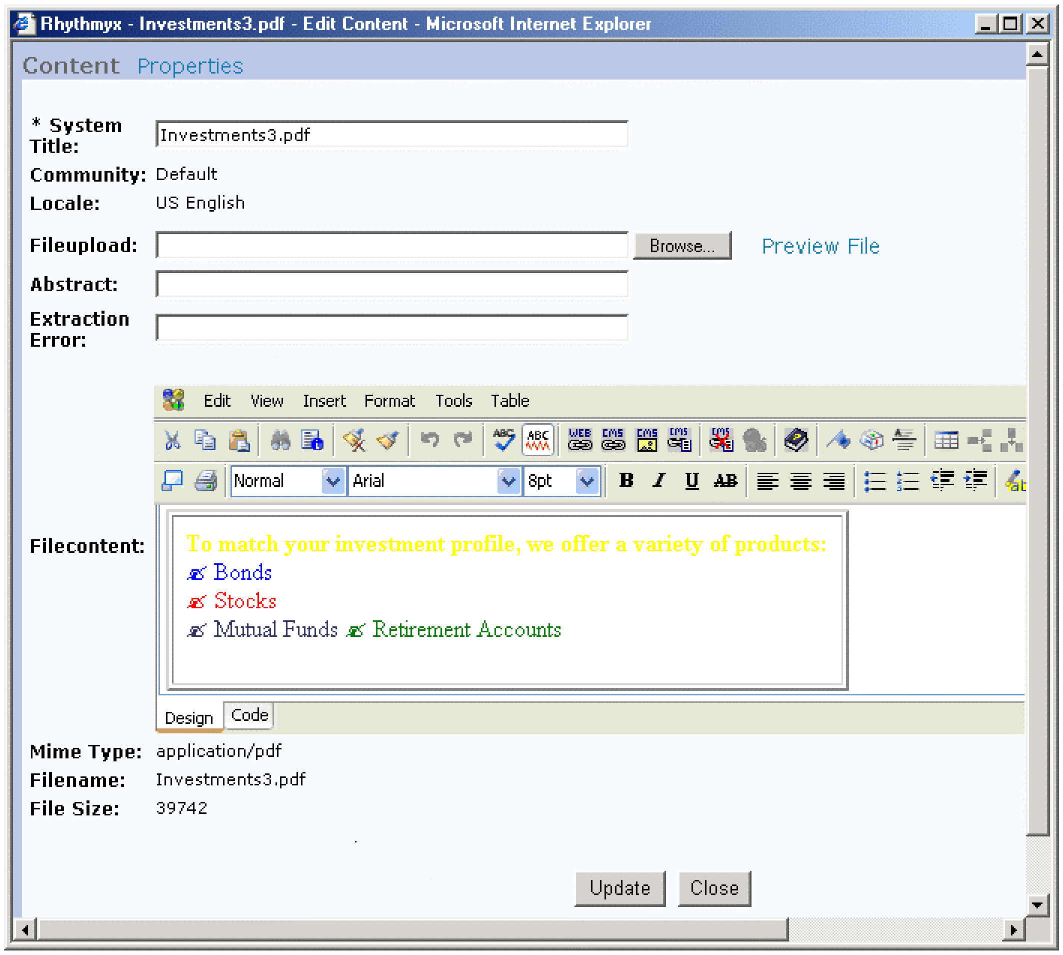


* System Title:	Investments3.pdf
Community:	Default
Locale:	US English
Fileupload:	
Abstract:	
Extraction Error:	
Filecontent:	Our Products ✍ Bonds ✍ Stocks ✍ Mutual Funds ✍ Retirement Accounts
Mime Type:	application/pdf
Filename:	Investments3.pdf
File Size:	39742

Figure 99: Opened Text Extractor Content Item

7 In this example, the body content of files is not edited in Rhythmyx (depending on your system requirements, you may choose to edit metadata fields). The original file is updated in its native application and re-uploaded to Rhythmyx, where it will automatically overwrite the originally uploaded Content Item.

If a content creator updates the file saved as Investments3.pdf in the original application, and recopies the file into the Marketing PDFs folder under My Network Places in Windows Explorer, the changed file overwrites the original Investments3.pdf. When you open it in Rhythmyx, the Filecontent field displays the changed content:



* System Title:	Investments3.pdf
Community:	Default
Locale:	US English
Fileupload:	
Abstract:	
Extraction Error:	
Filecontent:	To match your investment profile, we offer a variety of products: Bonds Stocks Mutual Funds Retirement Accounts
Mime Type:	application/pdf
Filename:	Investments3.pdf
File Size:	39742

Figure 100: Content updated in original application

Migration Example

In migration, text extraction transforms binary files that existed prior to Rhythmyx implementation into XML Content Items. When the files are uploaded to Rhythmyx, some of the data is extracted and inserted into Content Editor fields. In the future, these Content Items will be edited in Rhythmyx; they will not be returned to the original third-party application for editing.

In our example, a company has stored its employee profiles in Word doc files prior to implementing Rhythmyx. Now that the company has implemented Rhythmyx, it wants to convert these files into Rhythmyx Content Items and have the ability to process them in the future as Rhythmyx Content Items.

The procedure for implementing text extraction for migration of the Employee Profiles is essentially the same as the procedure for implementing continuous conversion of Marketing PDFs:

1. Follow Step 1 of the ***Continuous Conversion Example*** (on page 117) to set up a TextExtractor Content Editor to store the uploaded Word files. Since users may edit the Filecontent field in Rhythmyx, errors will result if you do not include the conditional described in this step. *Note:* You can use the same procedure because the TextExtractor Content Editor does not specify a specific type of file.
2. Follow Step 2 of the ***Continuous Conversion Example*** (on page 117) [link] to define a Servlet in the WebDAV deployment descriptor. Give the Servlet the name Employee Profiles.
3. Follow Step 3 of the ***Continuous Conversion Example*** (on page 117) to edit the WebDAV configuration file, but set *root* to:

   ```
   root="//Folders/Employee Profiles"
   ```
4. Set up WebDAV-enabled folders in Rhythmyx Content Explorer and Windows Explorer for storing the Word files.

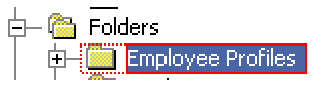
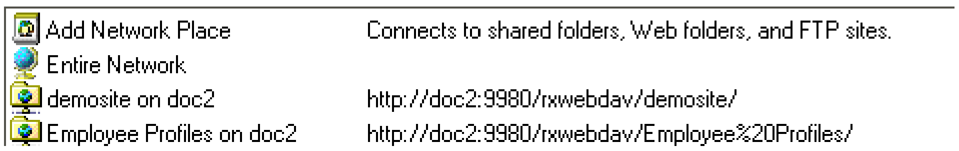


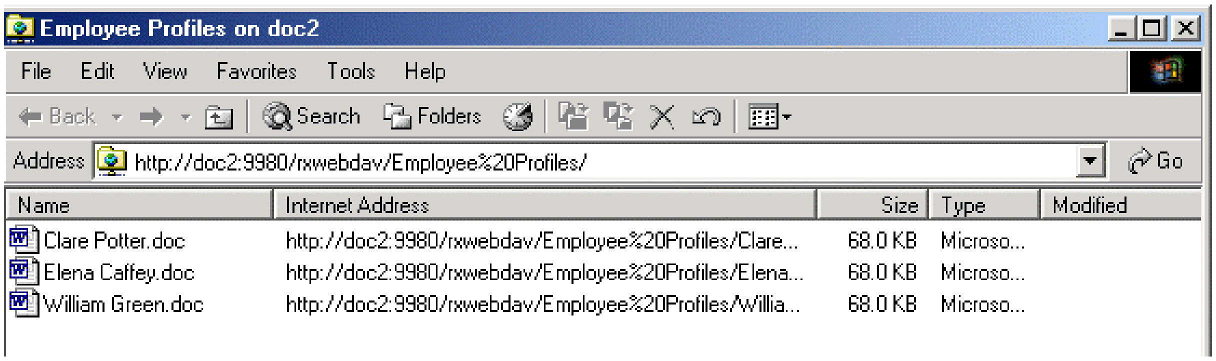
Figure 101: WebDAV-enabled Folder in Content Explorer



```
Add Network Place              Connects to shared folders, Web folders, and FTP sites.
Entire Network
demosite on doc2               http://doc2:9980/rxwebdav/demosite/
Employee Profiles on doc2      http://doc2:9980/rxwebdav/Employee%20Profiles/
```

Figure 102: Web Folders in Windows Explorer

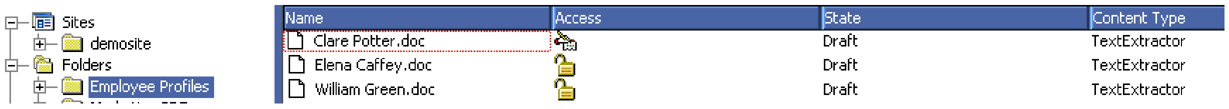
5 Copy the Word docs into the Web Folder on Windows Explorer.


Employee Profiles on doc2
File Edit View Favorites Tools Help
Back Search Folders
Address http://doc2:9980/rxwebdav/Employee%20Profiles/
Go

Name	Internet Address	Size	Type	Modified
Clare Potter.doc	http://doc2:9980/rxwebdav/Employee%20Profiles/Clare...	68.0 KB	Microso...	
Elena Caffey.doc	http://doc2:9980/rxwebdav/Employee%20Profiles/Elena...	68.0 KB	Microso...	
William Green.doc	http://doc2:9980/rxwebdav/Employee%20Profiles/Willia...	68.0 KB	Microso...	

Figure 103: Word document in Web Folder on Windows Explorer

They now exist in the WebDAV-enabled folders in Content Explorer, and Rhythmyx automatically converts them into TextExtractor Content Items.



Name	Access	State	Content Type
Clare Potter.doc		Draft	TextExtractor
Elena Caffey.doc		Draft	TextExtractor
William Green.doc		Draft	TextExtractor

Figure 104: WebDAV-enabled Folder in Rhythmyx

6 In this example, the file has become a Rhythmyx Content Item, so its body content is edited in Rhythmyx.

A Content Creator may open the file in Rhythmyx, edit it, and save it.

WebImageFX

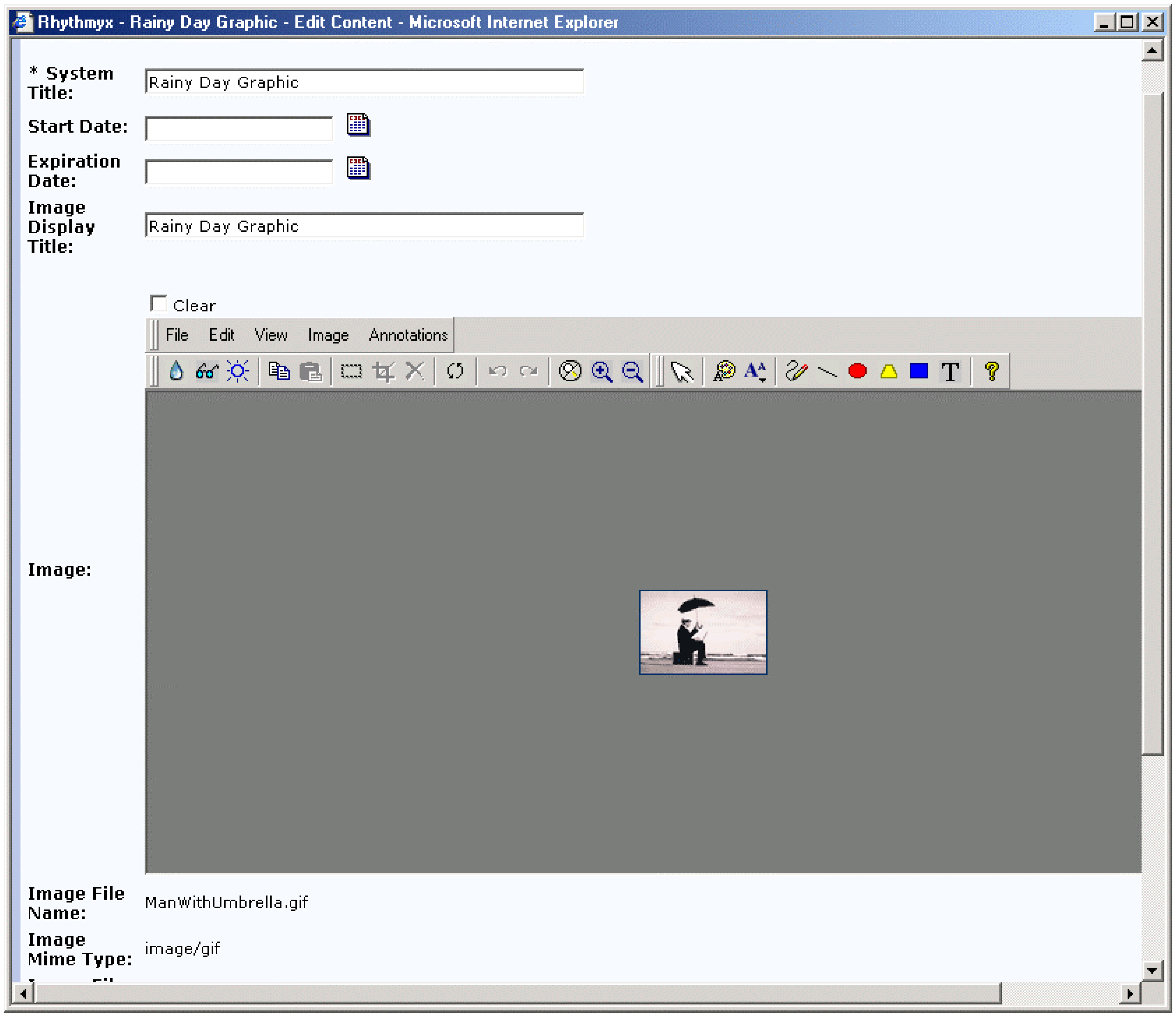
sys_WebImageFX and the WebImageFX Editor

Your Rhythmyx license may include Ektron's WebImageFX graphics editor which includes a variety of tools for creating and editing graphics files. With the WebImageFX editor, Rhythmyx includes the WebImageFX control. The control uploads a graphics file and displays it in a Content Editor using the WebImageFX editor.

An XML configuration file (ImageEditConfig.xml) defines the WebImageFX controls and styles available to the end user. You can customize this configuration file to add new functionality or to remove existing functionality. By default, the WebImageFX editor lets you upload, create, or paste (from Windows clipboard) images to edit in its window.

During installation, Rhythmyx installs a copy of WebImageFX to `Rhythmyxroot/sys_resources/webimagefx and checks the version of WebImageFx in Rhythmyxroot/rx_resources/webimagefx`. If the version in rx_resources is earlier than the current version (or there is no version file), Rhythmyx backs up the copy of WebImageFX in rx_resources (by adding a time stamp to the directory name, for example, webimagefx__0301_1538, and installs the current version into it.

The following Content Editor uses the sys_WebImageFX control to upload and display images.


Rhythmyx - Rainy Day Graphic - Edit Content - Microsoft Internet Explorer
Clear
File
Edit
View
Image
Annotations

* System Title: Rainy Day Graphic

Start Date:

Expiration Date:

Image Display Title: Rainy Day Graphic

Image:

Image File Name: ManWithUmbrella.gif

Image Mime Type: image/gif

Figure 105: Content Editor with sys_WebImageFX control

sys_WebImageFX Control

The sys_WebImageFX control functions almost identically to the sys_File control. It includes most of the same properties as the sys_File control, and like the sys_file control, it is a file upload element that allows the user to supply a file as the input, and it corresponds to a single, one-dimensional field. The main difference between the sys_WebImageFX control and the sys_File control is that the sys_WebImageFX control appears in a Content Editor with the WebImageFX image editor.

When hand-coding a Content Editor that includes a sys_WebImageFX control, include sys_FileInfo as a pre-exit. When you create a Content Editor using the Content Editor Properties dialog, Rhythmyx adds this exit for you automatically. The sys_FileInfo exit searches for attached files in a content item's HTML and returns values for file name, MIME type, character length and file encoding. The exit returns the values to field names formed by combining the filename (the <FieldRef> value) with descriptive suffixes. If you want to return information about a file, such as the filename or size, refer to the sys_FileInfo exit documentation in the *Workbench Online Help* for the correct syntax.

The sys_WebImageFX control displays a WebImageFX editor that allows a user to modify an image. For details about the standard features of WebImageFX, see the developer's guide at ***http://www.ektron.com/webimagefx.aspx*** (http://www.ektron.com/webimagefx.aspx). Note: This control has been tested with Internet Explorer.

Parameters:

Parameter	Data Type	Parameter Type	Description	Default
id	String	Generic	This parameter assigns a name to an element. This name must be unique in a document.	None
class	String	Generic	This parameter assigns a class name or set of class names to an element. Any number of elements may be assigned the same class name or names. Multiple class names must be separated by white space characters.	datadisplay
style	String	Generic	This parameter specifies style information for the current element. The syntax of the value of the style attribute is determined by the default style sheet language.	None
width	Number	Generic	This parameter tells the user agent the initial width of the control. The width is given in pixels.	800

Parameter	Data Type	Parameter Type	Description	Default
height	Number	Generic	This parameter tells the user agent the initial width of the control. The width is given in pixels.	400
config_src_url	String	Generic	This parameter specifies the location of the config.xml that will the control will use for configuration.	../sys_resources/ webimagefx/ ImageEditConfig.xml
cleartext	String	custom	This parameter determines the text that will be displayed along with a checkbox when the field supports being cleared.	Clear

Example Field Definition

```
<PSXField clearBinaryParam="yes" forceBinary="yes"
modificationType="user" name="uploadfilephoto" showInPreview="yes"
showInSummary="no" type="local" userCustomizable="yes"
userSearchable="yes">
   <DataLocator>
      <PSXBackEndColumn id="0">
         <tableAlias>RXWEBIMAGEFX</tableAlias>
            <column>IMGDATA</column>
            <columnAlias/>
      </PSXBackEndColumn>
   </DataLocator>
   <DataType>binary</DataType>
   <DataFormat>max</DataFormat>
   <OccurrenceSettings delimiter=";" dimension="optional"
      multiValuedType="delimited"/>
   <PSXPropertySet>
      <PSXProperty locked="no" name="mayHaveInlineLinks">
         <Value type="Boolean">no</Value>
      </PSXProperty>
      <PSXProperty locked="no" name="cleanupBrokenInlineLinks">
         <Value type="Boolean">no</Value>
      </PSXProperty>
   </PSXPropertySet>
</PSXField>
```

Example UI Definition

```
<PSXDisplayMapping>
   <FieldRef>uploadfilephoto</FieldRef>
      <PSXUISet>
         <Label>
            <PSXDisplayText>Image:</PSXDisplayText>
         </Label>
         <PSXControlRef id="1477" name="sys_webImageFx"/>
         <ErrorLabel>
```

```
            <PSXDisplayText>Uploadfilephoto:</PSXDisplayText>
         </ErrorLabel>
      </PSXUISet>
</PSXDisplayMapping>
```

Adding the sys_WebImageFX Control to a Content Editor

To create a Content Editor that uses WebImageFX:

1 Follow the procedure in the document *Implementing Content Editors* for creating a new Content Editor.

2 Include a field with the **Field Name** *uploadfilephoto* and the **Control Name** *sys_WebImageFX*.

3 When you choose sys_WebImageFX as the **Control Name**, Rhythmyx automatically includes the sys_FileInfo exit, which fills in the uploaded file's name, mime type, extension, and size into the proper Content Editor fields if you provide them. Add any of these fields to the Content Editor. See sys_FileInfo for required naming conventions for these fields.

4 Complete the standard procedure for creating the Content Editor.

To add the WebImageFX editor to a content editor:

1 In the Rhythmyx Workbench, access the Content Editor Properties dialog for the Content Editor.

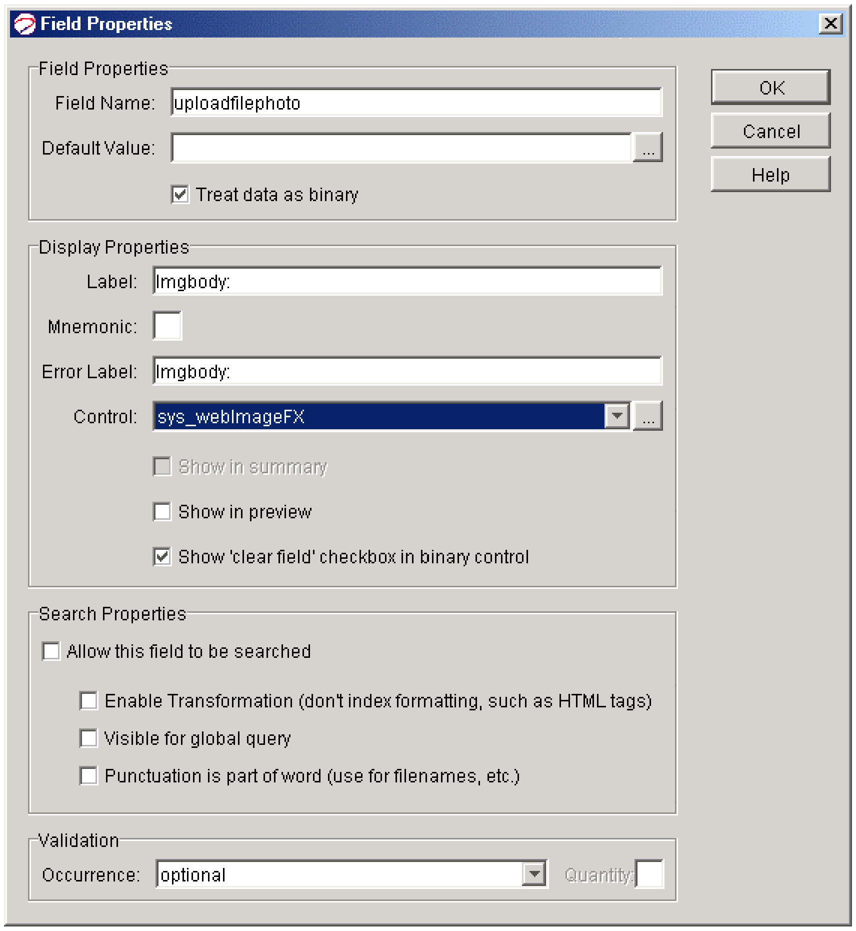
2 Select the field that you want to associate with WebImageFX and click [**Edit**].

   Rhythmyx displays the Field Properties dialog. (If you are defining a new field, Rhythmyx displays the New Field Properties dialog.)

3 Change the **Field Name** to *uploadfilephoto.* If you do not use this name, the WebImageFX control cannot upload the file.

4 Check **Treat Data as Binary**.

5 Select *sys_WebImageFX* as the **Control**.



Field Name: uploadfilephoto

Default Value:

☑ Treat data as binary

Label: Imgbody:

Mnemonic:

Error Label: Imgbody:

Control: sys_webImageFX

☐ Show in summary

☐ Show in preview

☑ Show 'clear field' checkbox in binary control

☐ Allow this field to be searched

☐ Enable Transformation (don't index formatting, such as HTML tags)

☐ Visible for global query

☐ Punctuation is part of word (use for filenames, etc.)

Occurrence: optional Quantity:

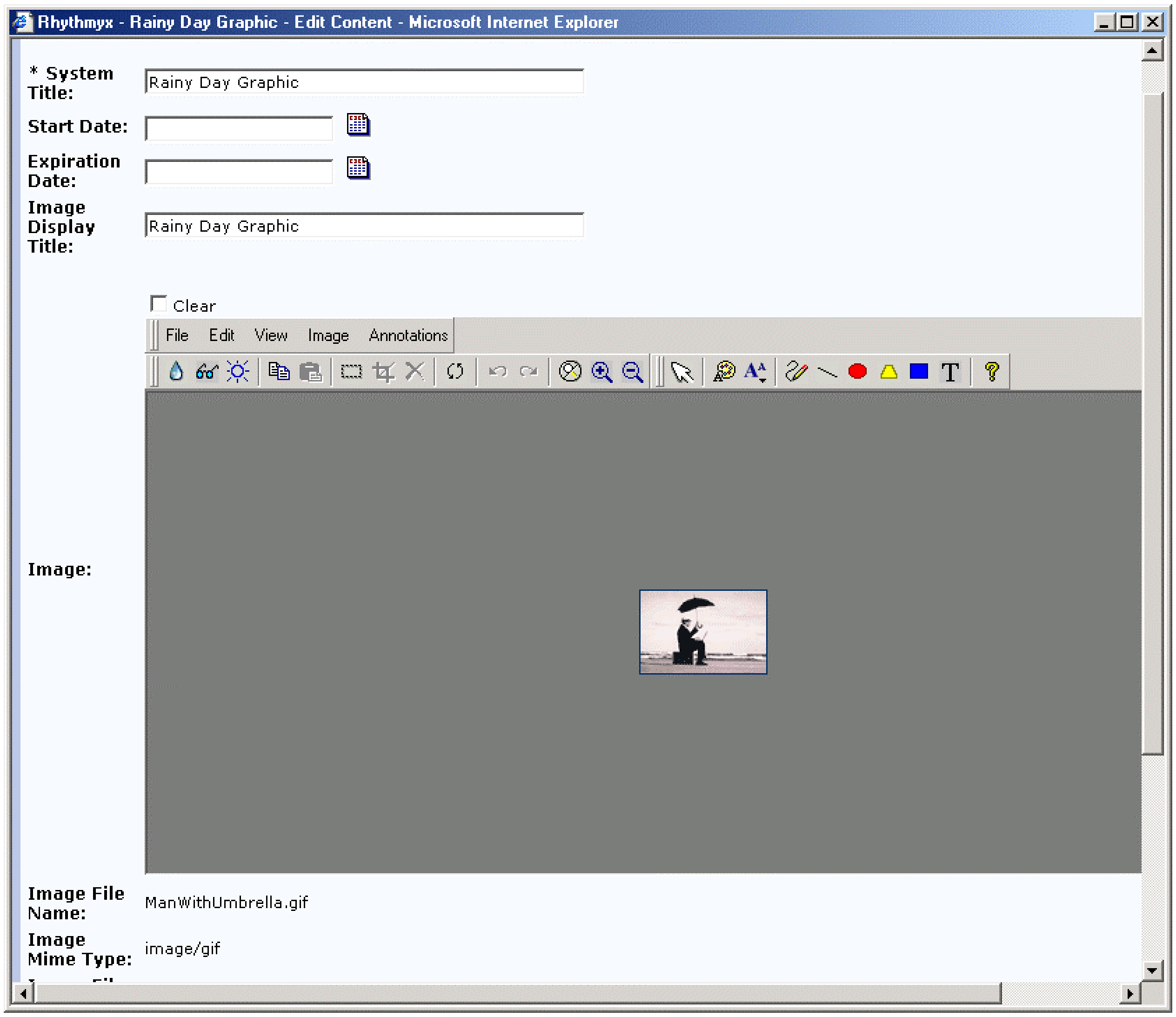
Figure 106: Field Properties Dialog for a field that uses sys_WebImageFX control

6 On the Field Properties dialog, click [**OK**].

7 When you choose sys_WebImageFX as the **Control Name**, Rhythmyx automatically includes the sys_FileInfo exit, which fills in the uploaded file's name, mime type, extension, and size into the proper Content Editor fields if you provide them. Add any of these fields to the Content Editor. See sys_FileInfo for required naming conventions for these fields.

8 On the Content Editor Properties dialog, click [**OK**].

The changes will take effect the next time you start your application. To see your changes, stop and restart the application, log in to Rhythmyx, and activate the editor.


Rhythmyx - Rainy Day Graphic - Edit Content - Microsoft Internet Explorer

* System Title:	Rainy Day Graphic
Start Date:	
Expiration Date:	
Image Display Title:	Rainy Day Graphic
Image:	
Image File Name:	ManWithUmbrella.gif
Image Mime Type:	image/gif

Figure 107: Content Editor with sys_WebImageFX control

The following limitations apply to all Content Editors that use this control:

- The name of the field containing the sys_WebImageFX control must be *uploadfilephoto.*
- Because the name of a field containing the sys_WebImageFX control must be *uploadfilephoto*, a Content Editor cannot have more than one sys_WebImageFX control. If it does, the additional controls will not be able to upload images.

- The names of fields in the Content Editor that sys_FileInfo updates (filename, type, size, and extension) must be prefixed with uploadfilephoto. For example, uploadfilephoto_filename, uploadfilephoto_type, uploadfilephoto_size, uploadfilephoto_ext. A Content Editor that contains a sys_WebImageFX control cannot also contain a sys_File control; if it does the sys_File control will not be able to upload a file.

NOTE: The first time you open a Content Editor that uses the sys_WebImageFX control in your Web browser, a dialog will prompt you to install WebImageFX. Follow the installation instructions in the dialog. After you initially install WebImageFX, you will not have to install it again.

You can customize both the parameters of the sys_WebImageFX control and the configuration files of the WebImageFX editor itself.

For guidance on customizing (and localizing) the WebImageFX editor, consult the `WebImageFX Developer's Reference Guide`, at ***http://www.ektron.com/webimagefx.aspx*** (http://www.ektron.com/webimagefx.aspx).

Most customizations of the WebImagFX editor involve modifications to the configuration file (ImageEditConfig.xml). Do not modify the default configuration file, which is located in the `Rhythmyxroot/rx_resources/WebImageFX` directory. Instead, customize shared or local definition files. If you only use one customized configuration file, best practice is to use a shared configuration file.

In the default ImageEditConfig.xml used in Rhythmyx, the upload and exit options are disabled because these actions cannot function in Rhythmyx; do not enable these options when you edit copies of the ImageEditConfig.xml file.

Several instances of the control can use the same configuration XML file (shared configuration file), or you can use a local configuration file for each instance of the editor; you can also use a shared configuration file for some instances and a local configuration file for other instances. The files must be stored in the following manner:

- The default configuration file is stored in the directory `rx_resources/WebImageFX`. This configuration file should not be modified.
- Shared configuration files should be stored in a directory with the path `rx_resources/[path]/WebImageFX`, where `[path]` is the path to a subdirectory that logically categorizes the file. For example, you might want to use the name of your project as part of the path; for a project with the name *sample*, the path would be `rx_resources/sample/WebImageFX`.
- Local configuration files should be stored in a subdirectory of the Content Editor application. For example, if you have a local configuration file for a Press Release content editor, the configuration file would be stored in the subdirectory `Rhythmyxroot/pressrelease/WebImageFX`.

To define an instance of the sys_WebImageFX control to use a customized configuration file:

1. Open the Content Editor in the Rhythmyx Workbench and access the Content Editor Properties dialog.
2. Select the field that uses the WebImagFX editor and click [**Edit**] to open the Field Properties dialog.

3. Click the browse button (...) next to the **Control** field.

   Rhythmyx displays the Display Control Properties for <field> dialog.

4. Click in the **Param name** column and choose *config_src_url*.
5. Click in the **Value** column of the same row and enter the relative URL of the configuration file you want to use for this instance of the control as a literal value.
6. On the Display Control Properties for <field> dialog, click [**OK**].
7. On the Field Properties dialog, click [**OK**].
8. On the Content Editor Properties dialog, click [**OK**].

The changes will take effect the next time you start your application. To see your changes, stop and restart the application, log in to Rhythmyx, and activate the editor.

Best Practices: sys_WebImageFX

To simplify maintenance and promote effective technical support, observe the following Best Practices when working with the WebImageFX editor and the sys_WebImageFX control:

- Keep shared configuration files (configuration files used by more than one instance of the control) in directories with the name `Rhythmyxroot/rx_resources/[path]/webimagefx`, where `[path]` defines a category (such as a the name of a project or customer). For example, if you are working on a project named `sample`, the directory should be `Rhythmyxroot/rx_resources/sample/webimagefx`.
- If only one editor is going to use a configuration file, store the file in a subdirectory of the editor application directory. If you decide to use this configuration file for other editors, move it to a shared directory and update the `SRC` parameters of the instances of the control that use that configuration file.
- When disabling a command or parameters of a command (such as lists of fonts or font sizes), hide the disabled elements first by commenting them out (`<!-- text --!>`), then test and refine your development. Remove the disabled commands and parameters when testing is complete to minimize clutter in the files and simplify future modification.

Implementing WebDAV in Rhythmyx's Installation of Tomcat

Rhythmyx includes a default WebDAV Web application. You must customize the files in the default application to implement your system's WebDAV setup.

This topic lists the procedure for implementing WebDAV in Rhythmyx's installation of Tomcat. For specific details about performing each step, see ***Simple Example Implementation of WebDAV*** (on page 144).

NOTES:
The default implementation of Rhythmyx's WebDAV specifies Folders installed with FastForward. If you have installed both FastForward and Tomcat, you can access the default Web Folders with the users and passwords *rxuser/demo* and *admin1/demo*.

The sample WebDAV configuration file `<Rhythmyx root>\AppServer\webapps\rxwebdav\RxWebdavConfig_for_OldSample.xml` refers to the previous Rhythmyx samples which can be installed with the archive file `<Rhythmyx root>/Samples/DeprecatedSamples.pda.` If you choose to use this file, you must modify the RxWebDAV application deployment descriptor, web.xml, to refer to the Servlet that it configures.

For information about installing and using FastForward, see the FastForward documentation.

To implement WebDAV in Rhythmyx's installation of Tomcat:

1. To function in Rhythmyx, WebDAV requires a Web Server Security Provider. New installations of Rhythmyx include a Web Server Security Provider automatically. If you are upgrading from a previous version of Rhythmyx and have not already implemented a Web Server Security Provider, you must define one. For details about implementing a Web Server Security Provider, see the topic "Web Server" in the *Rhythmyx Server Administrator Online Help*. NOTE: If you are not using other security providers included in Rhythmyx, remove them to avoid possible errors.
2. The Roles that you want to have access to your WebDAV servlet must be defined in Rhythmyx, the rxwebdav deployment descriptor (<Rhythmyx root>\AppServer\webapps\rxwebdav\WEB-INF\web.xml), and Tomcat.
   a) If the Roles are not already defined in Rhythmyx, add them. Associate the Roles with the Community that you want to have access to the WebDAV-enabled Folder. In addition add them to the Workflows assigned to your Content Types. Assign each Role to all States in its Workflows as an assignee.
   b) Back up the default rxwebdav Web application deployment descriptor, `<Rhythmyx root>\AppServer\webapps\rxwebdav\WEB-INF\web.xml. In your current copy`, add these Roles if they are not already included in the <SecurityConstraint> element.
   c) In the file `<Rhythmyx root>\AppServer\conf\tomcat-users.xml`, define the Roles if they are not already defined and associate them with the usernames and passwords that you want to use with WebDAV.

3 Create the folders that you want to associate with WebDAV under the Sites and Folders nodes in the Content Explorer navigation pane. In the Create Folder dialog **Folder Community** field, choose the value of the Community that you want to have access to your Servlet.

a) Choose *All Communities* or a specific Community (which must be the same as the Community entered in the rxwebdav configuration file).

b) Give the Roles that you want to have access to the Folder in Rhythmyx Write or Admin permission to the Folder.

4 Create a WebDAV Web application deployment descriptor, and configure the default WebDAV servlet and add additional WebDAV servlets:

a) Back up the original deployment descriptor, `<Rhythmyx root>\AppServer\webapps\rxwebdav\WEB-INF\web.xml` and open the current copy. There are four <servlet> nodes that contain the default WebDAV servlets, *CorporateInvestmentImages*, *CorporateInvestmentFiles*, *EnterpriseInvestmentImages* and *EnterpriseInvestmentFiles.*

b) In one of the default <servlet> nodes, change <servlet-name> to a custom name for your servlet, and change the name in the <servlet-mapping> element to the custom name. Change the name of the config file in <init-param><param-value> to a custom xml filename for your servlet. Be sure to change the name of the physical config file, (either `RxWebDAVConfigEIImages.xml, RxWebDAVConfigEIFiles.xml, RxWebDAVConfigCIImages.xml, or RxWebDAVConfigCIFiles.xml` to match the new value. Delete servlets and servlet mappings that you are not using.

c) If you are not using the default Rhythmyx Port, 9992, enter the correct value in the port init-param.

To add additional WebDAV servlets, see ***Adding Additional WebDAV Servlets*** (see "Adding Additional WebDAV Servlets in Rhythmyx's Installation of Tomcat" on page 141).

5 Configure a WebDAV configuration file for each WebDAV servlet. The configuration files for the default WebDAV servlets are:

```
<Rhythmyx
root>\AppServer\webapps\rxwebdav\RxWebdavConfigEIImages.xml
<Rhythmyx
root>\AppServer\webapps\rxwebdav\RxWebdavConfigEIFiles.xml
<Rhythmyx
root>\AppServer\webapps\rxwebdav\RxWebdavConfigCIImages.xml
and
<Rhythmyx
root>\AppServer\webapps\rxwebdav\RxWebdavConfigCIFiles.xml
```

Back up one of these files. Rename the current copy with the configuration file name you specified in your deployment descriptor. Edit this file to specify the Rhythmyx Folder and Community you want to make available through your WebDAV servlet, and specify the Content Types you want to manage using this servlet.

For each additional WebDAV servlet in the WebDAV application, make a copy of the configuration file in the same location, give it the name you specified in the <init param><param value> element for the servlet, and modify the Rhythmyx Folder, Community and Content Types.

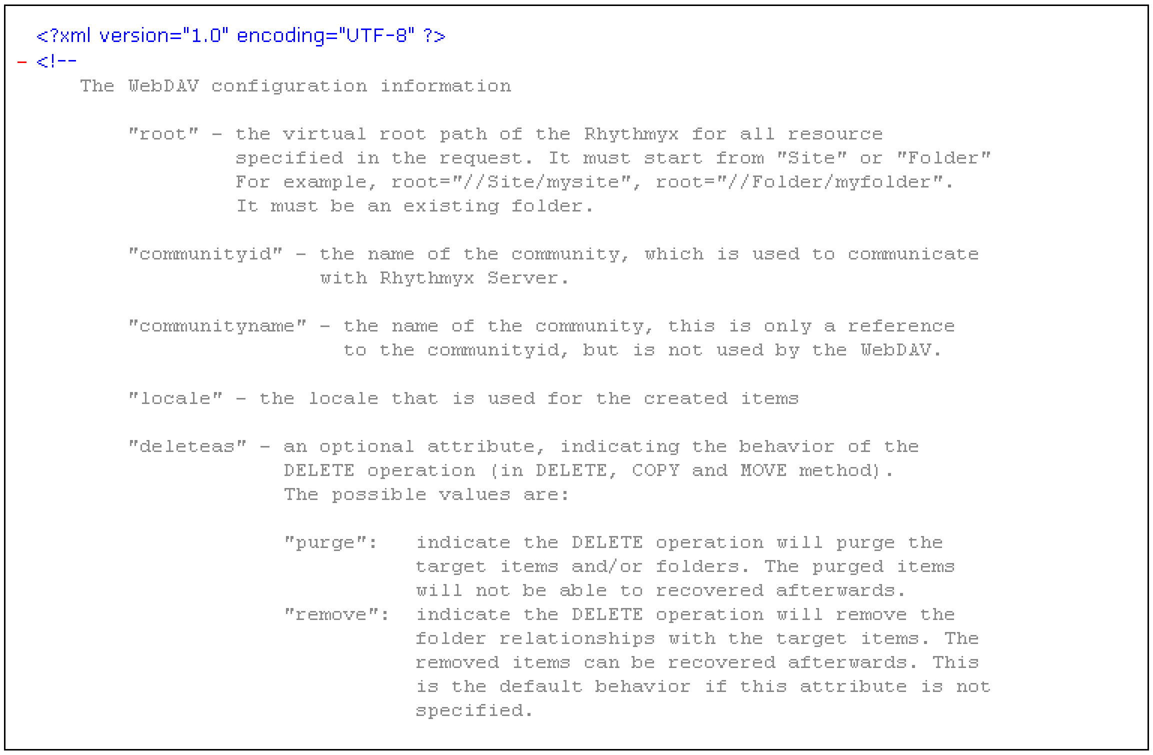
For more information about editing the configuration file, see ***Editing a WebDAV Configuration File*** (on page 142).

6 To create an additional WebDAV application in Tomcat, you must copy and rename the `rxwebdav.war` used for your Web application server and deploy it to a different folder in `<Rhythmyx root>\AppServer\webapps`. Then you must make the changes listed in the above steps to the deployment descriptor and WebDAV config file(s).

7 After you make changes to your WebDAV configuration, restart your Web application server to apply them.

8 To confirm that implementation is correct, restart the Tomcat Server. In your Web browser:

- Enter `http://hostname:9980/rxwebdav/[ServletURLPattern]?sys_command=getConfig`.
  If the WebDAV servlet is configured correctly in the application server the browser displays the contents of the WebDAV configuration file.



```
  <?xml version="1.0" encoding="UTF-8" ?>
- <!--
    The WebDAV configuration information

        "root" - the virtual root path of the Rhythmyx for all resource
                 specified in the request. It must start from "Site" or "Folder"
                 For example, root="//Site/mysite", root="//Folder/myfolder".
                 It must be an existing folder.

        "communityid" - the name of the community, which is used to communicate
                        with Rhythmyx Server.

        "communityname" - the name of the community, this is only a reference
                          to the communityid, but is not used by the WebDAV.

        "locale" - the locale that is used for the created items

        "deleteas" - an optional attribute, indicating the behavior of the
                     DELETE operation (in DELETE, COPY and MOVE method).
                     The possible values are:

                     "purge":   indicate the DELETE operation will purge the
                                target items and/or folders. The purged items
                                will not be able to recovered afterwards.
                     "remove":  indicate the DELETE operation will remove the
                                folder relationships with the target items. The
                                removed items can be recovered afterwards. This
                                is the default behavior if this attribute is not
                                specified.
```

Figure 108: Get Configuration screen

- Enter `http://hostname:9980/rxwebdav/[ServletURLPattern]?sys_command=validateConfig`.
  If the WebDAV servlet's references to Rhythmyx elements are correct, the screen displays the message "WebDAV configuration successfully validated".

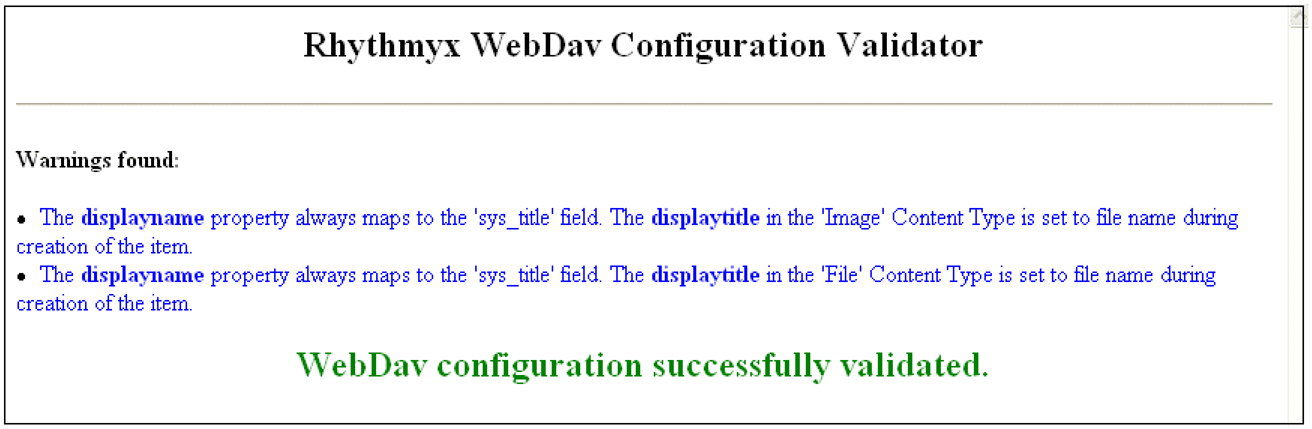
The warnings displayed on the following screen appear if WebDAV is uploading a file into a Content Type with a required displaytitle field to avoid confusion over which field WebDAV displays as the item's title. If displaytitle is required, Rhythmyx WebDAV does the following when it uploads a file into Rhythmyx:

- It sets the Rhythmyx Content Item's displaytitle field to the file name.
- It sets the Rhythmyx Content Item's sys_title field to the file name.

During processing after upload, WebDAV:

- Does not use the Rhythmyx Content Item's displaytitle field.
- Maps the Rhythmyx Content Item's sys_title field to WebDAV's displayname property. When the value in one changes, the other sets itself to the new value.

Therefore, changing the Content Item's displaytitle field does not change the displayname that WebDAV shows to users. Explain this to your users who work with Rhythmyx WebDAV.



Rhythmyx WebDav Configuration Validator

Warnings found:

- The **displayname** property always maps to the 'sys_title' field. The **displaytitle** in the 'Image' Content Type is set to file name during creation of the item.
- The **displayname** property always maps to the 'sys_title' field. The **displaytitle** in the 'File' Content Type is set to file name during creation of the item.

WebDav configuration successfully validated.

Figure 109: Validation screen

Note: In the WebDAV configuration file, the servlet URL pattern is the value of the *root* parameter in the <PSXWebdavConfigDef> element.

For example, in the sample WebDAV configuration file, the <PSXWebdavConfigDef> element appears as:

```
<PSXWebdavConfigDef root="//Sites/EnterpriseInvestments/Files"
communityname="Enterprise Investments" communityid="1002"
locale="en-us" deleteas="remove" publicValidTokens="y"
QEValidTokens="i">
```

Also, in the WebDAV deployment descriptor, <RhythmyxRoot>\AppServer\webapps\rxwebdav\WEB-INF\web.xml, the servlet URL pattern is specified in the <servlet-mapping> element for the servlet in the <url-pattern> sub-element.

To use the getConfig command, enter:

```
http://hostname:9980/rxwebdav/Sites/EnterpriseInvestments/Files
?sys_command=getConfig.
```

9 Open the folders in Windows Explorer or Internet Explorer, and if you plan to access the folders through a third-party application, configure the application as specified in its documentation.

WebDAV is now configured in your Rhythmyx system. You may now create Rhythmyx Content Items in Windows Explorer and your other third-party applications.

Adding Additional WebDAV Servlets in Rhythmyx's Installation of Tomcat

One WebDAV Web application can support multiple WebDAV servlets. To add an additional servlet to a WebDAV Web application:

1 Back up one of the default configuration files. Rename the new file for the servlet you are creating. For example, create a servlet named *USMarketing* with a configuration file named `RxWebdavConfigMarketing.xml`.

2 Open the deployment descriptor (<RhythmyxRoot>\AppServer\webapps\rxwebdav\WEB-INF\web.xml) of the WebDAV Web application.

3 Find the servlet definition:

```
<servlet>
   <servlet-name>EnterpriseInvestmentImages</servlet-name>
   <display-name>Rx WebDAV for EnterpriseInvestment Images
      folder</display-name>
   <description>Rhythmyx WebDAV for EnterpriseInvestment Images
      folder</description>
   <servlet-class>com.percussion.webdav.PSWebdavServlet</servlet-
class>
   <init-param>
      <param-name>RxWebDAVConfig</param-name>
      <param-value>/RxWebDAVConfigEIImages.xml</param-value>
      <description>The webdav configuration file path, which is
       relative to the current web application</description>
   </init-param>
</servlet>
```

Copy this code and paste the copy immediately below the original.

4 Change `<servlet-name>EnterpriseInvestmentImages</servlet-name>` to `<servlet-name>USMarketing</servlet-name>`

5 Change the value of the RxWebDAVConfig parameter to the name of the WebDAV configuration file you want to implement with this servlet. For example:

```
<servlet>
    <servlet-name>USMarketing</servlet-name>
    <display-name>Rx WebDAV for Marketing folder</display-name>
    <description>Rx WebDAV for Marketing folder</description>
    <servlet-class>com.percussion.webdav.PSWebdavServlet</servlet-
class>
    <init-param>
      <param-name>RxWebDAVConfig</param-name>
      <param-value>/RxWebdavConfigMarketing.xml</param-value>
      <description>The webdav configuration file path, which is
         relative to the current web application</description>
    </init-param>
  </servlet>
```

6 Find the <servlet-mapping> elements:

```
<servlet-mapping>
   <servlet-name>CorporateInvestmentImages</servlet-name>
   <url-pattern>/Sites/CorporateInvestments/Images/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
   <servlet-name>CorporateInvestmentFiles</servlet-name>
   <url-pattern>/Sites/CorporateInvestments/Files/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
   <servlet-name>EnterpriseInvestmentImages</servlet-name>
   <url-pattern>/Sites/EnterpriseInvestments/Images/*</url-
pattern>
</servlet-mapping>
<servlet-mapping>
   <servlet-name>EnterpriseInvestmentFiles</servlet-name>
   <url-pattern>/Sites/EnterpriseInvestments/Files/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
   <servlet-name>Rhythmyx</servlet-name>
   <url-pattern>/Rhythmyx/*</url-pattern>
</servlet-mapping>
```

and add one for the new servlet:

```
<servlet-mapping>
   <servlet-name>USMarketing</servlet-name>
   <url-pattern>/USMarketing/*</url-pattern>
</servlet-mapping>
```

7. Repeat steps 3 to 5 for each servlet you want to implement.
8. Restart the Web application.
9. The URL for accessing the new Servlet is `http://host:port/rxwebdav/USMarketing`.

Editing a WebDAV Configuration File

A WebDAV configuration file specifies the Rhythmyx Folder and Community the WebDAV servlet makes available, and the Content Types that users can manage through WebDAV. The WebDAV configuration file DTD is RxWebDAVconfig.dtd, and the default WebDAV configuration files in Rhythmyx are `RxWebDAVConfigEIImages.xml, RxWebDAVConfigEIFiles.xml, RxWebDAVConfigCIImages.xml` and `RxWebDavConfigCIFiles.xml`. They are located in `<Rhythmyx root>\AppServer\webapps\rxwebdav`.

A WebDAV configuration file consists of the following nodes:

PSXWebdavConfigDef - specifies the Rhythmyx Folder and Community the WebDAV servlet makes available. This node includes the following attributes:

root - specifies the Rhythmyx Folder. The value of this attribute must begin with either "//Sites/" (if the Folder you want to make available is a Site Folder) or "//Folders/" (if the Folder you want to make available is a child of the Folders node).

communityname - specifies the name of the Community you want to make available. NOTE: The Community property of the root folder can be either *All Communities* or the Community you specify here.

communityid - specifies the ID of the Community (Rhythmyx assigns the ID when you save the Community; you can find it in parentheses next to the Community Name on the Communities Editor on the System tab of Content Explorer).

locale – Locale for Content Items created in this Folder.

deleteas – How the delete function for items and Folders behaves. Possible values are *remove* and *purge*. Note that if a Content Item is in either Public or Quick Edit State, Rhythmyx does not perform any action on that item. If the Content Item is in any other state, Rhythmyx executes he action normally.

- remove – Default. If a WebDAV-enabled Folder is deleted in Rhythmyx or a third-party application, permanently delete the Folder in Rhythmyx and remove the Content Items from the Folder, but save them in Rhythmyx.
- Purge – If a WebDAV-enabled Folder is deleted in Rhythmyx or a third-party application, permanently delete the Folder and the Content Items in Rhythmyx.

publicValidTokens - specifies a list of flags for Public States in the Workflow. Used with the attribute QEValidTokens to implement automatic checkout of Content Items in the WebDAV Folder. Default value is *y*.

QEValidTokens - specifies a list of flags for Quick Edit States in the Workflow. Used with the publicValidTokens attribute to implement automatic checkout of Content Items in the WebDAV Folder. Default value is *i*.

PSXWebdavContentType – One or more of these nodes may be present. Each node specifies a Content Type that users can maintain in the Folder the servlet makes available.

id - specifies the ID of the Content Type; you can find it in parentheses next to the Content Type Name on the Content Types Editor on the System tab of Content Explorer.

name - specifies the name of the Content Type.

contentfield - specifies the Content Type field that stores the data file for the Content Type.

ownerfield - specifies the Content Type field that stores the lock owner data from WebDAV.

default - specifies whether the Content Type is the default Content Type for the servlet. The default Content Type is the Content Type that processes any MIME Types not specified for another Content Type in the configuration. Only one Content Type in the configuration can be specified as the default Content Type. Values are *true* or *false*.

MIMETypes - specifies the MIME types of files that will be processed by the Content Type. Each MIME type is specified in a separate Mime Type child node.

PropertyMap - stores the mappings of WebDAV properties to the Rhythmyx fields. The name attribute of the PSXPropertyFieldNameMapping element specifies the WebDAV property. The FieldName child of the PSXPropertyFieldNameMapping specifies the Rhythmyx Content Type field that stores the data for the specified WebDAV property.

PSXPropertyFieldNameMapping name="displayname" - Required for FastForward Content Types. Specifies the display title of the Content Item. WebDAV passes the filename as the display title.

PSXPropertyFieldNameMapping name="getcontenttype" - Required. Specifies the MIME type of the content.

PSXPropertyFieldNameMapping name="getcontentlength"– Required. Specifies the length, in bytes, of the content.

ExcludeFolderProperties – specifies which Rhythmyx custom properties of the parent WebDAV Folder should not be inherited by sub-folders created in WebDAV. Note that the Folder Name and Description are automatically not inherited (you must specify a new Folder Name and Description is always blank).

PropertyName – sys_pubFileName is listed because it specifies the name of the folder to publish to. This name would most likely differ in the sub-folder.

NOTE: Fields specified to store data for a WebDAV configuration must be from a System, Shared, or Local definition. You cannot use child fields to store WebDAV data.

Simple Example Implementation of WebDAV

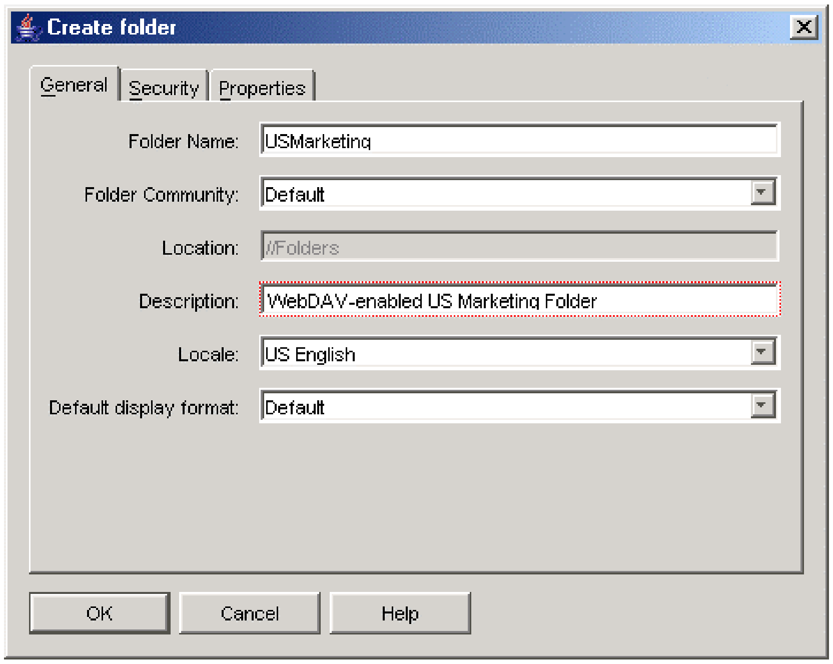
To illustrate a typical implementation using WebDAV, let us configure two WebDAV servlets to support the Rhythmyx Folders for two departments in a company. The two WebDAV servlets will reside in the same Web application and run on the default Rhythmyx Web Application Server (Tomcat).

We will configure:

- a WebDAV servlet to support the USMarketing Folder for the Default Community. The folder is accessible to the Marketing_Author and Marketing_Artist Roles as well as the Default Role.
- a WebDAV servlet to support the USEditorial Folder for the Editorial Community. The folder is accessible only to Editor and Administrator Roles.

To set up our sample WebDAV configuration:

1 First, as an Administrator, we create the folders in Content Explorer that we want to WebDAV-enable. We create the USMarketing Folder and assign it to the Default Community.



Folder Name: USMarketing
Folder Community: Default
Location: //Folders
Description: WebDAV-enabled US Marketing Folder
Locale: US English
Default display format: Default

We give the Role `Admin` Read, Write, and Admin permission, and we give the Roles `Marketing_Author`, `Marketing_Artist`, and `Default` Read and Write permissions:

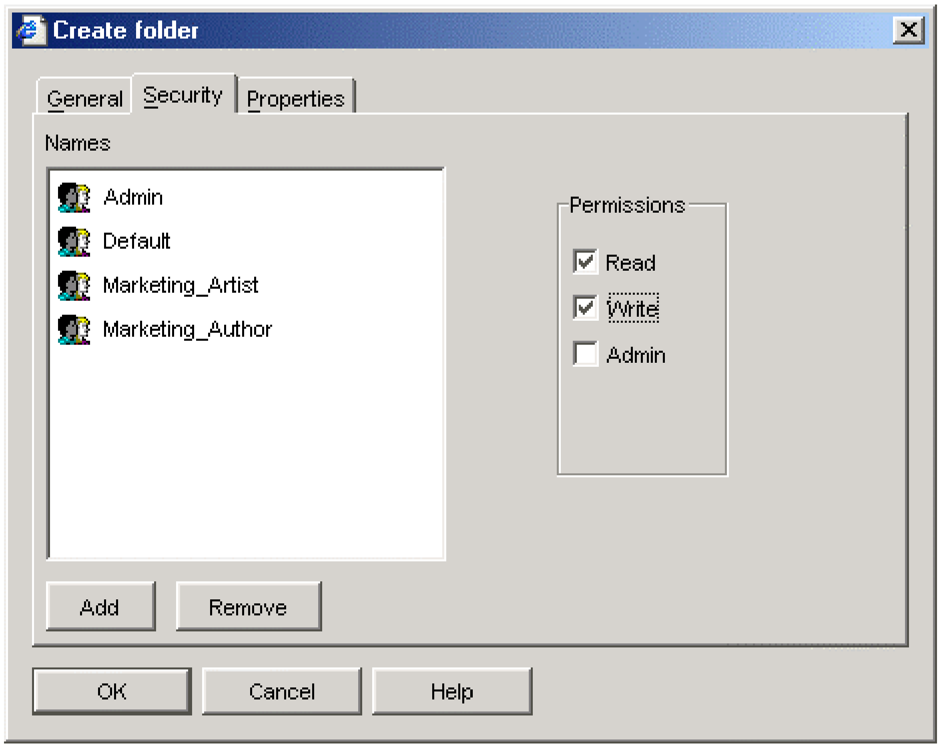


Figure 110: Create Folder Dialog, Security Tab

Then we create the USEditorial folder. We assign it to the Editorial Community and give the `Admin` Role Read, Write, and Admin permission and the `Editor` Role Read and Write permission.

Our completed folder structure appears as:

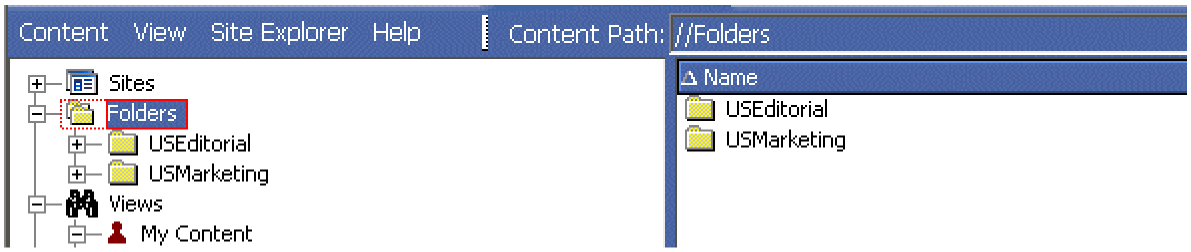


Figure 111: WebDAV-enabled Folders in Rhythmyx

2 We create a Web Server Security Provider. (Note: If you have a new installation of Rhythmyx, the Web Server Security Provider is already included, otherwise you must create it.) For details about implementing a Web Server Security Provider, see the topic “Web Server” in the `Rhythmyx Server Administrator` online Help. NOTE: If you are not using other security providers included in Rhythmyx, remove them to avoid possible errors.

3 We add the Roles that will have access to our Folders to the Rhythmyx servlet that is located in our WebDAV Web application. We back up `<Rhythmyxroot>\AppServer\webapps\rxwebdav\WEB-INF\web.xml.` In the current copy, we edit the <SecurityConstraint> element of the Rhythmyx servlet:

```
 <security-constraint>
   <web-resource-collection>
      <web-resource-name>Entire Application</web-resource-name>
         <url-pattern>/*</url-pattern>
   </web-resource-collection>
   <auth-constraint>
      <!-- Define a list of roles with rights to access the
         Servlet - Roles need to exist in both the App Server and
         Rhythmyx -->
      <!-- For Tomcat (App Server), you may need to modify
         tomcat-root/conf/tomcat-users.xml file, add userid,
         password and assign Rhythmyx roles to the userid -->
      <!-- Replace the value of role-name element with a character
         '*' if the roles are resolved by Rhythmyx, but not by
         Tomcat -->
         <role-name>Admin</role-name>
         <role-name>Default</role-name>
         <role-name>rxrole</role-name>
         <role-name>Web_Admin</role-name>
         <role-name>CI_Members</role-name>
         <role-name>Internet_Members</role-name>
         <role-name>CI_Admin_Members</role-name>
         <role-name>Internet_Admin_Members</role-name>
         <role-name>Marketing_Author</role-name>
         <role-name>Marketing_Artist</role-name>
   </auth-constraint>
   <user-data-constraint>
      <transport-guarantee>NONE</transport-guarantee>
   </user-data-constraint>
</security-constraint>
```

4 We also add the Roles to `<Rhythmyx root>\AppServer\conf\tomcat-users.xml` and associate them with the user and password we want to use to access WebDAV:

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
  <role rolename="CI_Admin_Members"/>
  <role rolename="Intranet_Admin_Members"/>
  <role rolename="CI_Members"/>
  <role rolename="tomcat"/>
  <role rolename="Internet_Members"/>
  <role rolename="Internet_Admin_Members"/>
  <role rolename="Marketing_Artist"/>
  <role rolename="Marketing_Author"/>
  <role rolename="manager"/>
  <role rolename="Default"/>
  <role rolename="Admin"/>
  <role rolename="admin"/>
  <role rolename="Web_Admin"/>
   <user username="admin1" password="demo"
      roles="Admin,Default,Web_Admin,Internet_Members,
      Internet_Admin_Members,Marketing_Author,Marketing_Artist"/>
   <user username="tomcat" password="tomcat"
      roles="admin,manager,tomcat"/>
   <user username="admin2" password="demo"
      roles="Admin,Default,Web_Admin,CI_Members,CI_Admin_Members,
      Marketing_Author,Marketing_Artist"/>
   <user username="rxuser" password="demo"
      roles="Admin,Default,Web_Admin,Internet_Members,
      Internet_Admin_Members,CI_Members,CI_Admin_Members"/>
</tomcat-users>
```

5 We modify the WebDAV deployment descriptor to define our two servlets:

We change the default servlet to have:

- USMarketing as a <servlet-name>;
- RxWebdavConfigMarketing.xml as a config file name in <init-param><param-value>.

We copy the <servlet> node and paste the copy under the original. We change the copied servlet to have:

- USEditorial as a <servlet-name>;
- RxWebdavConfigEditorial.xml as a config file name.

```
  <servlet>
    <servlet-name>USMarketing</servlet-name>
    <display-name>Rhythmyx WebDAV Router</display-name>
    <description>Rhythmyx WebDAV Router</description>
    <servlet-class>com.percussion.webdav.PSWebdavServlet</servlet-
class>
    <init-param>
      <param-name>RxWebDAVConfig</param-name>
      <param-value>/RxWebdavConfigMarketing.xml</param-value>
      <description>The webdav configuration file path, which is
         relative to the current web application</description>
    </init-param>
  </servlet>
  <servlet>
    <servlet-name>USEditorial</servlet-name>
    <display-name>Rhythmyx WebDAV Router</display-name>
    <description>Rhythmyx WebDAV Router</description>
    <servlet-class>com.percussion.webdav.PSWebdavServlet</servlet-
class>
    <init-param>
      <param-name>RxWebDAVConfig</param-name>
      <param-value>/RxWebdavConfigEditorial.xml</param-value>
      <description>The webdav configuration file path, which is
         relative to the current web application</description>
    </init-param>
  </servlet>
```

We modify the servlet mappings to specify these two servlets:

```
   <servlet-mapping>
      <servlet-name>USMarketing</servlet-name>
      <url-pattern>/USMarketing/*</url-pattern>
```

```
</servlet-mapping>
<servlet-mapping>
   <servlet-name>USEditorial</servlet-name>
   <url-pattern>/USEditorial/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
   <servlet-name>Rhythmyx</servlet-name>
   <url-pattern>/Rhythmyx/*</url-pattern>
</servlet-mapping>
```

We delete the servlets and servlet mappings that we are not using to prevent WebDAV from failing.

6 Next, we open each configuration file and specify the Rhythmyx Folder and Community that the WebDAV servlet makes available. In RxWebdavConfigMarketing, we make the following changes:

- we change root to //Folders/USMarketing;
- to make the Folder USMarketing available to members of the Default Community, we change the value of communityname to `Default;`
- we leave the default Content Type of File and modify the Image Content Type node to define the PDFfile Content Type;
- in the PDFfile definition:
  - we specify contentfield as `pdfupload,` the name of the PDFFile Content Type field that stores WebDAV content;
  - we change ownerfield to `lockuser,` the name of the PDFFile Content Type filed that stores the WebDAV user who has a lock on the item;
  - we set default to false, because the File Content Type is already the default.
- we change <Mimetype> to the appropriate value for PDFfile, `application/pdf;`
- in <PropertyMap>, we map the WebDAV field getcontenttype to the Rhythmyx Content Type field `pdfupload_type`. These are the fields in each application that store the MIME type. We also map the WebDAV field `getcontentlength` to the Rhythmyx Content Type field `pdfupload_size`. These are the fields in each application that store the Content Length.

```
<PSXWebdavConfigDef root="//Folders/USMarketing"
    communityname="Default" communityid="10" locale="en-
us" deleteas="remove">
<!--Defines one or more supported content type for
WebDAV
   "id" - the id of the content type in Rhythmyx
   "name" - the name of the content type in Rhythmyx,
      this is only used as areference to the id above.
   "contentfield" - the field name for the content
   "ownerfield" - the field name for the lock owner that
      is submitted by various WebDAV clients
   "default" - "true" or "false". There must be only one
      element with the value "true".
      If it is "false", must define one or more mime
      types (hence must define the "Mimetypes" element);
      If it is "true", the "Mimetypes" element is
      ignored.
   -->
   <PSXWebdavContentType id="318" name="PDFFile"
      contentfield="pdfupload" ownerfield="lockuser"
      default="false">
      <!-- Defines the mime type(s) that be accepted by
the current Rhythmyx content type.-->
      <MimeTypes>
         <MimeType>application/pdf</MimeType>
      </MimeTypes>
      <!--
        Defines the mapper between WebDAV properties and
         its corresponding field name of the current
         Rhythmyx content type
         Couple required mappings:
            "getcontenttype" - maps to a field for the
               mime-type
            "getcontentlength" - maps to a field for the
               content length
      -->
      <PropertyMap>
         <PSXPropertyFieldNameMapping name="displayname">
            <FieldName>displaytitle</FieldName>
         </PSXPropertyFieldNameMapping>
         <PSXPropertyFieldNameMapping
            name="getcontenttype">
            <FieldName>pdfupload_type</FieldName>
         </PSXPropertyFieldNameMapping>
         <PSXPropertyFieldNameMapping
            name="getcontentlength">
            <FieldName>pdfupload_size</FieldName>
         </PSXPropertyFieldNameMapping>
      </PropertyMap>
   </PSXWebdavContentType>
   <PSXWebdavContentType id="309" name="File"
      contentfield="item" ownerfield="webdavowner"
      default="true">
      <PropertyMap>
         <PSXPropertyFieldNameMapping name="displayname">
            <FieldName>displaytitle</FieldName>
         </PSXPropertyFieldNameMapping>
         <PSXPropertyFieldNameMapping
            name="getcontenttype">
            <FieldName>item_type</FieldName>
         </PSXPropertyFieldNameMapping>
         <PSXPropertyFieldNameMapping
            name="getcontentlength">
            <FieldName>item_size</FieldName>
         </PSXPropertyFieldNameMapping>
      </PropertyMap>
   </PSXWebdavContentType>
      <!--
      A list of excluded folder properties when creating
a folder in WebDAV.
```

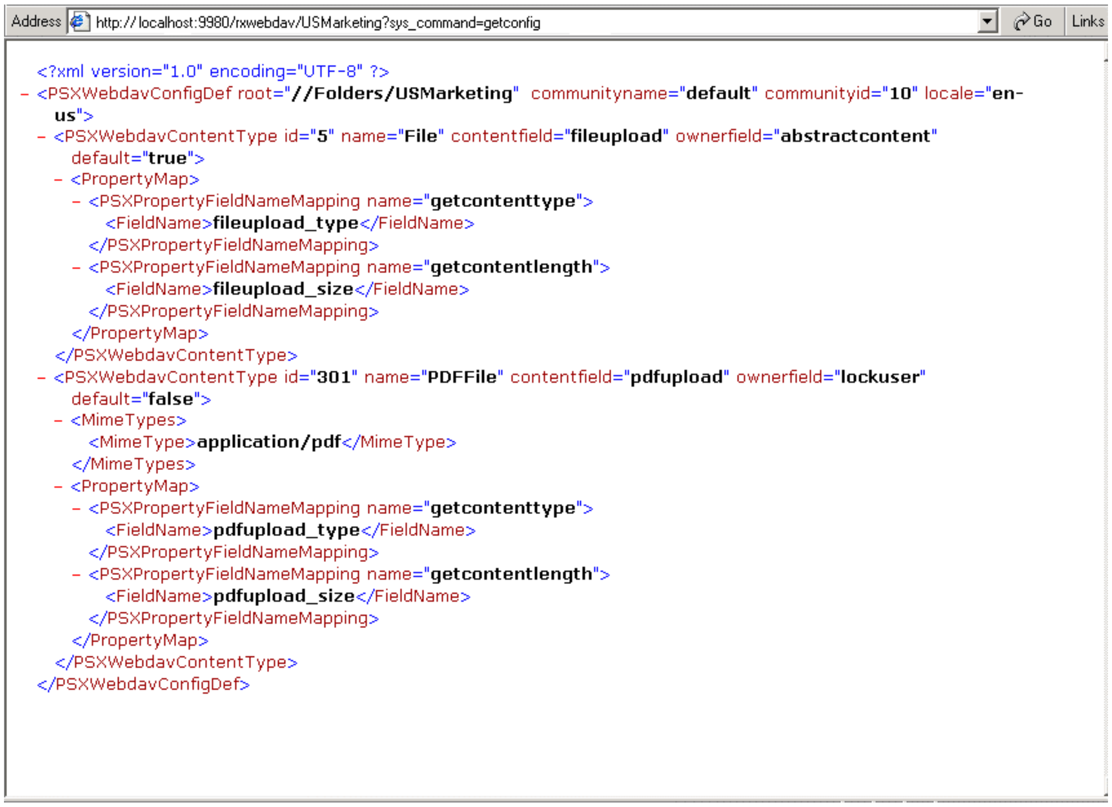
```
     A created folder will inherit all properties of its
parent folder except the list of excluded properties and
the folder name.
   -->
   <ExcludeFolderProperties>
      <PropertyName>sys_pubFileName</PropertyName>
   </ExcludeFolderProperties>
</PSXWebdavConfigDef>
```

- In RxWebdavConfigEditorial, we change root to //Folders/USEditorial, and we make the Folder Editorial available to members of the Editorial Community. We make the Image Content Type the default, and add a node for the Logo Content Type. The changes we make to the <PSXWebdavContentType> node for Logo are similar to those we made for the PDFFile Content Type, above.

```
<PSXWebdavConfigDef root="//Folders/USEditorial"
   communityname="Editorial" communityid="1001"
   locale="en-us" deleteas="remove"
<!--Defines one or more supported content type for
WebDAV
   "id" - the id of the content type in Rhythmyx
   "name" - the name of the content type in Rhythmyx,
      this is only used as areference to the id above.
   "contentfield" - the field name for the content
   "ownerfield" - the field name for the lock owner that
      is submitted by various WebDAV clients
   "default" - "true" or "false". There must be only one
      element with the value "true".
      If it is "false", must define one or more mime
      types (hence must define the "Mimetypes" element);
      If it is "true", the "Mimetypes" element is
      ignored.
   -->
   <PSXWebdavContentType id="307" name="Image"
      contentfield="img1" ownerfield="webdavowner"
      default="true">
      <!--
        Defines the mapper between WebDAV properties and
         its corresponding field name of the current
         Rhythmyx content type
         Couple required mappings:
            "getcontenttype" - maps to a field for the
               mime-type
            "getcontentlength" - maps to a field for the
               content length
      -->
   <PropertyMap>
      <PSXPropertyFieldNameMapping name="displayname">
         <FieldName>displaytitle</FieldName>
      </PSXPropertyFieldNameMapping>
      <PSXPropertyFieldNameMapping name="getcontenttype">
         <FieldName>img1_type</FieldName>
      </PSXPropertyFieldNameMapping>
      <PSXPropertyFieldNameMapping
         name="getcontentlength">
         <FieldName>img1_size</FieldName>
      </PSXPropertyFieldNameMapping>
   </PropertyMap>
</PSXWebdavContentType>
<PSXWebdavContentType id="12" name="Logo"
   contentfield="Logobody" ownerfield="lockuser"
   default="false">
   <!-- Defines the mime type(s) that be accepted by the
current Rhythmyx content type.-->
   <MimeTypes>
      <MimeType>image/gif</MimeType>
   </MimeTypes>
   <PropertyMap>
      <PSXPropertyFieldNameMapping name="displayname">
         <FieldName>displaytitle</FieldName>
      </PSXPropertyFieldNameMapping>
      <PSXPropertyFieldNameMapping name="getcontenttype">
         <FieldName>logobody_type</FieldName>
      </PSXPropertyFieldNameMapping>
      <PSXPropertyFieldNameMapping
name="getcontentlength">
         <FieldName>logobody_size</FieldName>
      </PSXPropertyFieldNameMapping>
      </PropertyMap>
   </PSXWebdavContentType>

   <!--
      A list of excluded folder properties when creating
a folder in WebDAV.
      A created folder will inherit all properties of its
parent folder except the list of excluded properties and
the folder name.
   -->
   <ExcludeFolderProperties>
      <PropertyName>sys_pubFileName</PropertyName>
   </ExcludeFolderProperties>
</PSXWebdavConfigDef>
```

7 After deployment is complete, we restart the Tomcat Server. In our Web browser, we enter: `http://hostname:9980/rxwebdav/USMarketing?sys_command=getConfig` If the WebDAV servlet has deployed correctly the browser displays the contents of the WebDAV configuration file.

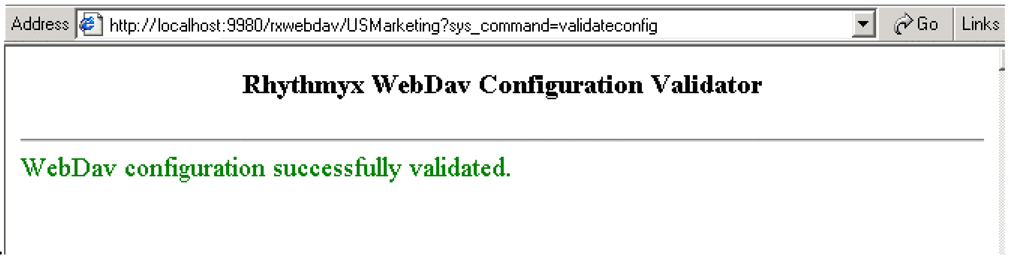


Address http://localhost:9980/rxwebdav/USMarketing?sys_command=getconfig

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <PSXWebdavConfigDef root="//Folders/USMarketing"  communityname="default" communityid="10" locale="en-
    us">
  - <PSXWebdavContentType id="5" name="File" contentfield="fileupload" ownerfield="abstractcontent"
      default="true">
    - <PropertyMap>
      - <PSXPropertyFieldNameMapping name="getcontenttype">
          <FieldName>fileupload_type</FieldName>
        </PSXPropertyFieldNameMapping>
      - <PSXPropertyFieldNameMapping name="getcontentlength">
          <FieldName>fileupload_size</FieldName>
        </PSXPropertyFieldNameMapping>
      </PropertyMap>
    </PSXWebdavContentType>
  - <PSXWebdavContentType id="301" name="PDFFile" contentfield="pdfupload" ownerfield="lockuser"
      default="false">
    - <MimeTypes>
        <MimeType>application/pdf</MimeType>
      </MimeTypes>
    - <PropertyMap>
      - <PSXPropertyFieldNameMapping name="getcontenttype">
          <FieldName>pdfupload_type</FieldName>
        </PSXPropertyFieldNameMapping>
      - <PSXPropertyFieldNameMapping name="getcontentlength">
          <FieldName>pdfupload_size</FieldName>
        </PSXPropertyFieldNameMapping>
      </PropertyMap>
    </PSXWebdavContentType>
  </PSXWebdavConfigDef>
```

We repeat using `http://hostname:9980/rxwebdav/USEditorial?sys_command=getConfig`

- Then, we enter: `http://hostname:9980/rxwebdav/USMarketing?sys_command=validateConfig` If the WebDAV servlet is configured correctly, the screen displays a confirmation message.



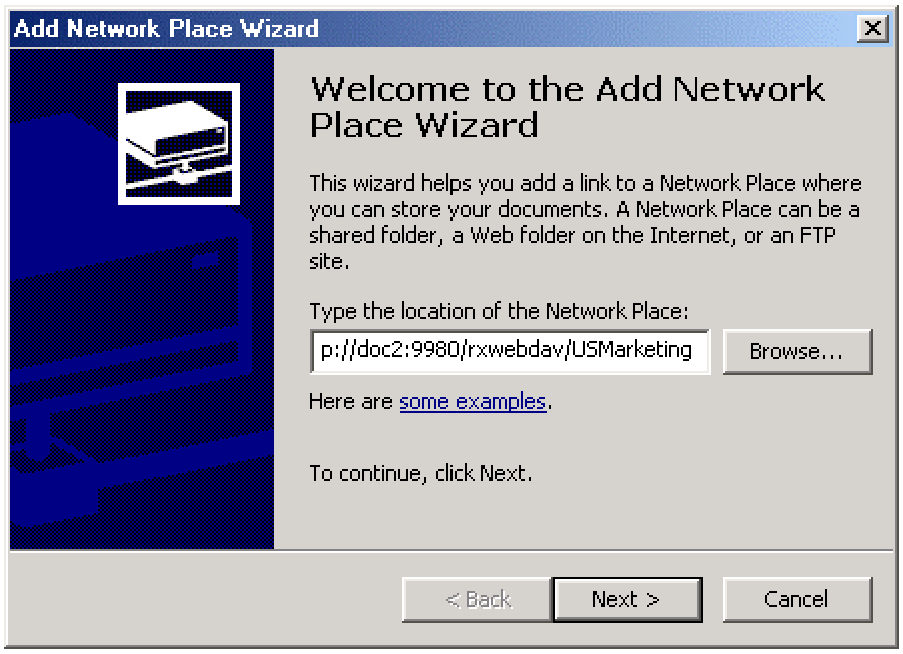
Address http://localhost:9980/rxwebdav/USMarketing?sys_command=validateconfig

Rhythmyx WebDav Configuration Validator

WebDav configuration successfully validated.

We repeat using `http://hostname:9980/rxwebdav/USEditorial?sys_command=validateConfig`.

8 Now, we can create Network Places in Windows Explorer that make these Rhythmyx folders available through Windows Explorer. We start Tomcat. In the Add Network Place Wizard, we enter the locations for our WebDAV servlets in the format http://localhost:<appserver port>/<WebDAV application>/<servlet URL pattern>. The servlet URL pattern is specified in the `root` parameter of the <PSXWebDAVConfigDef> element of the configuration file.



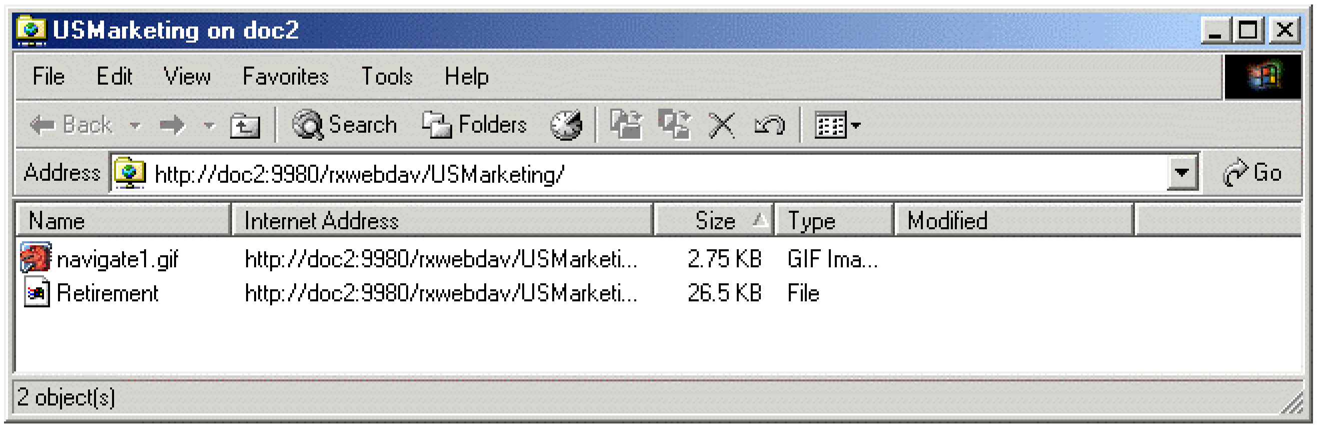
This wizard helps you add a link to a Network Place where you can store your documents. A Network Place can be a shared folder, a Web folder on the Internet, or an FTP site.

Type the location of the Network Place:

p://doc2:9980/rxwebdav/USMarketing

Figure 112: Add Network Place Wizard

Windows Explorer will ask us to log in when it connects to the WebDAV URL. The username and password must be defined in the `<Rhythmyx root>\AppServer\conf\tomcat-users.xml` file. When the login is authenticated, Windows Explorer displays the new Network Place:



http://doc2:9980/rxwebdav/USMarketing/

Name	Internet Address	Size	Type	Modified
navigate1.gif	http://doc2:9980/rxwebdav/USMarketi...	2.75 KB	GIF Ima...	
Retirement	http://doc2:9980/rxwebdav/USMarketi...	26.5 KB	File	

2 object(s)

Figure 113: WebDAV opened in Windows Explorer

9 We perform any steps that are required to connect a third-party application to the WebDAV enabled folders.

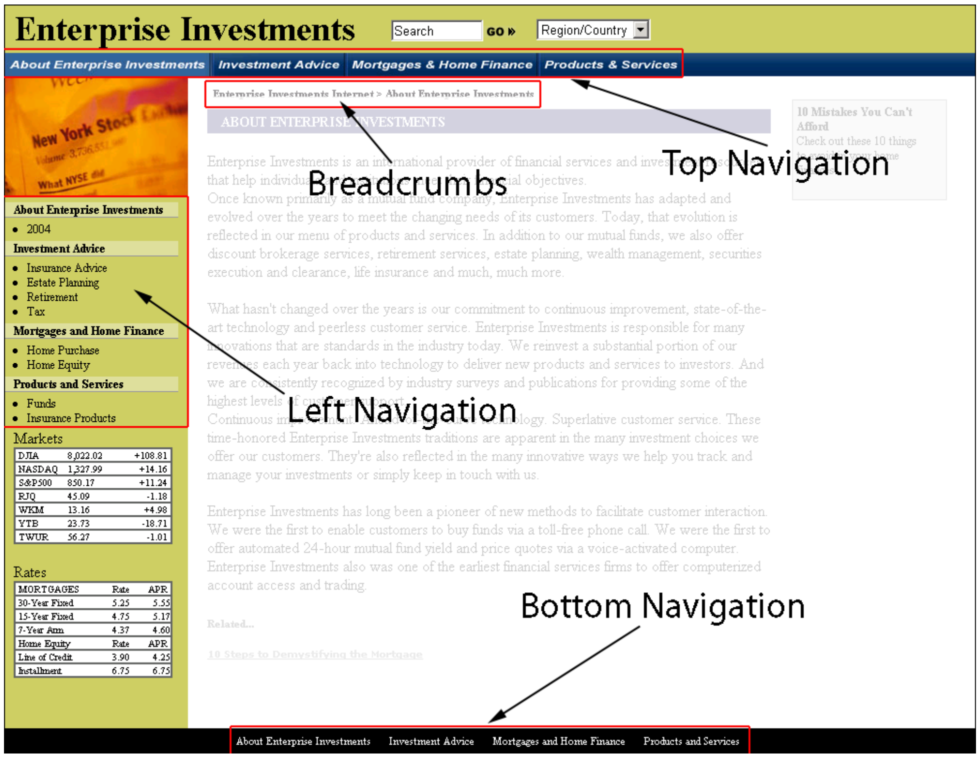
CHAPTER 7

Sites and Managed Navigation

Seamless and intuitive navigation through a web site assures its visitors a successful interaction. Site navigation is typically comprised of a combination of the following navigation elements:

- top navigation bar
- side navigation
- bottom navigation
- breadcrumbs
- a Site map

Effective employment of these elements adds to a site's ease of use.



Enterprise Investments Internet > About Enterprise Investments

ABOUT ENTERPRISE INVESTMENTS

Enterprise Investments is an international provider of financial services and inves[illegible] that help individua[illegible] [illegible]ial objectives.
Once known primarily as a mutual fund company, Enterprise Investments has adapted and evolved over the years to meet the changing needs of its customers. Today, that evolution is reflected in our menu of products and services. In addition to our mutual funds, we also offer discount brokerage services, retirement services, estate planning, wealth management, securities execution and clearance, life insurance and much, much more.

What hasn't changed over the years is our commitment to continuous improvement, state-of-the-art technology and peerless customer service. Enterprise Investments is responsible for many innovations that are standards in the industry today. We reinvest a substantial portion of our revenues each year back into technology to deliver new products and services to investors. And we are consistently recognized by industry surveys and publications for providing some of the highest levels of customer support.
Continuous im[illegible] [illegible]logy. Superlative customer service. These time-honored Enterprise Investments traditions are apparent in the many investment choices we offer our customers. They're also reflected in the many innovative ways we help you track and manage your investments or simply keep in touch with us.

Enterprise Investments has long been a pioneer of new methods to facilitate customer interaction. We were the first to enable customers to buy funds via a toll-free phone call. We were the first to offer automated 24-hour mutual fund yield and price quotes via a voice-activated computer. Enterprise Investments also was one of the earliest financial services firms to offer computerized account access and trading.

Related...

10 Steps to Demystifying the Mortgage

10 Mistakes You Can't Afford
Check out these 10 things [illegible]

About Enterprise Investments
- 2004

Investment Advice
- Insurance Advice
- Estate Planning
- Retirement
- Tax

Mortgages and Home Finance
- Home Purchase
- Home Equity

Products and Services
- Funds
- Insurance Products

Markets

DJIA	8,022.02	+108.81
NASDAQ	1,327.99	+14.16
S&P500	850.17	+11.24
RJQ	45.09	-1.18
WKM	13.16	+4.98
YTB	23.73	-18.71
TWUR	56.27	-1.01

Rates

MORTGAGES	Rate	APR
30-Year Fixed	5.25	5.55
15-Year Fixed	4.75	5.17
7-Year Arm	4.37	4.60
Home Equity	Rate	APR
Line of Credit	3.90	4.25
Installment	6.75	6.75

About Enterprise Investments   Investment Advice   Mortgages and Home Finance   Products and Services

Figure 114: Press Release with Side Navigation Menu and Breadcrumbs

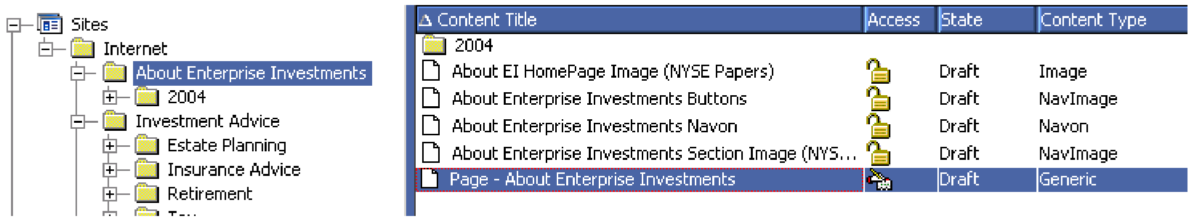
FastForward provides a Managed Navigation system that makes it easy to add navigation elements to a Web Page. This system is based on three navigation Content Types.

Navon - Items used to create navigation menus including breadcrumbs, bottom, side and top navigation, and site maps. Each Navon should be linked to a Content Item not used for Navigation (such as a Generic Page or a Category Content Item). A Navon may also be associated with a NavImage (see below).

NavTree - Similar to a Navon. NavTree Items reside at the root of a Site. These Items initiate the propagation of Navons to every sub folder in the Site Tree. The NavTree Item is generally linked to the Site's Home Page Item.

NavImage - Images used by Navons to replace text links for navigation elements. NavImages are also used by several Content Types to provide Image Links.

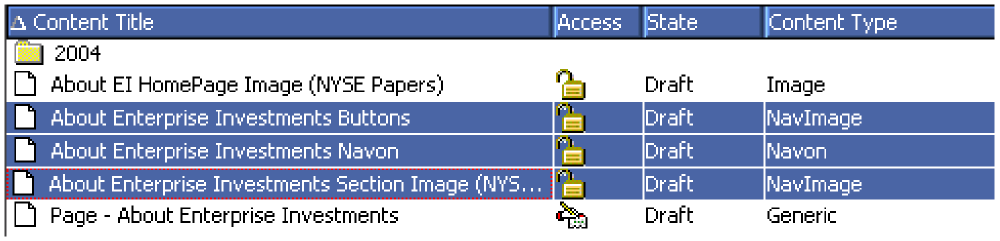
The About Corporate Investments Generic page Content Item, for example, contains a side navigation menu, breadcrumbs, and a bottom navigation menu (which is not visible in the diagram above). The Rhythmyx Managed Navigation system will automatically generate these elements based on the Site Folder path of the Item and its relationship to navigation Items such as a NavTree and Navon. A Site Folder is a Folder defined in the Content Explorer under the Sites node. This folder generally represents a folder within a web site's directory structure. The Item being previewed is in a Site Folder named About Corporate Investments.



Content Title	Access	State	Content Type
2004			
About EI HomePage Image (NYSE Papers)		Draft	Image
About Enterprise Investments Buttons		Draft	NavImage
About Enterprise Investments Navon		Draft	Navon
About Enterprise Investments Section Image (NYS…		Draft	NavImage
Page - About Enterprise Investments		Draft	Generic

Figure 115: Press Release Item in 2004 Site Folder

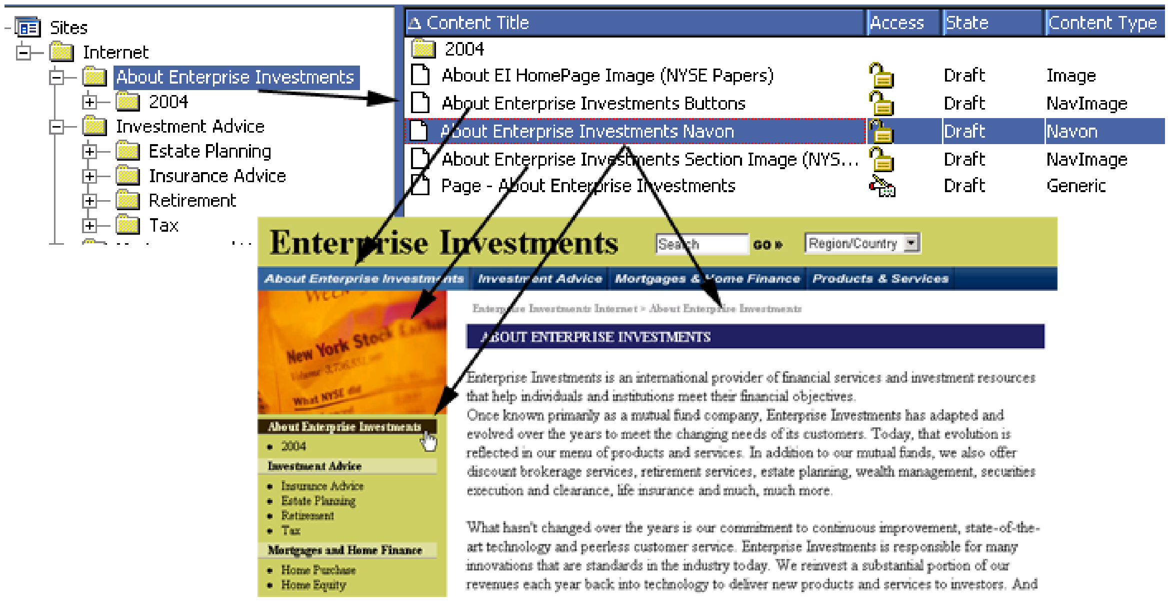
This Folder may or may not have additional Subfolders. Each of these Folders and Subfolders may contain navigation Content Items.



Content Title	Access	State	Content Type
2004			
About EI HomePage Image (NYSE Papers)		Draft	Image
About Enterprise Investments Buttons		Draft	NavImage
About Enterprise Investments Navon		Draft	Navon
About Enterprise Investments Section Image (NYS…		Draft	NavImage
Page - About Enterprise Investments		Draft	Generic

Figure 116: Navon for 2004 Press Releases

These elements control how (and whether) the Folder is represented in a navigation menu.



Content Title	Access	State	Content Type
2004			
About EI HomePage Image (NYSE Papers)		Draft	Image
About Enterprise Investments Buttons		Draft	NavImage
About Enterprise Investments Navon		Draft	Navon
About Enterprise Investments Section Image (NYS...		Draft	NavImage
Page - About Enterprise Investments		Draft	Generic

Enterprise Investments Internet > About Enterprise Investments

ABOUT ENTERPRISE INVESTMENTS

Enterprise Investments is an international provider of financial services and investment resources that help individuals and institutions meet their financial objectives.
Once known primarily as a mutual fund company, Enterprise Investments has adapted and evolved over the years to meet the changing needs of its customers. Today, that evolution is reflected in our menu of products and services. In addition to our mutual funds, we also offer discount brokerage services, retirement services, estate planning, wealth management, securities execution and clearance, life insurance and much, much more.

What hasn't changed over the years is our commitment to continuous improvement, state-of-the-art technology and peerless customer service. Enterprise Investments is responsible for many innovations that are standards in the industry today. We reinvest a substantial portion of our revenues each year back into technology to deliver new products and services to investors. And

Figure 117: Folder and Navon Result in Navigation Structure

Folders that contain a Navon or NavTree Content Item are generally rendered as a menu Item. If a NavImage is associated with the Navon or NavTree Item, this image will be represented in the top navigation.

Note: The out-of-the box solution for a top navigation Snippet requires a NavImage to render a visible representation of the Item. All other navigation Snippets use text values.

Finally, if either the Navon or NavTree Item has Content Items assigned to its landing page Slot, the menu link will target the specified page and provide links to the navigation section.

Rhythmyx provides out of the box solutions for common navigation menus. In addition, intuitive XML Variants allow you to develop customized navigation when the standard solutions do not apply. Though a Flash driven navigation menu is not provided by Percussion, the XML necessary to implement such an application is available out of the box.

As shipped, Managed Navigation allows implementers to create common navigation based on the Site's Folder structure without the need for complex XSL. A combination of specialized Communities, Content Types, and Effects provides implementers with the capability of implementing a variety of navigation structures while masking the complexities of Managed Navigation from Content Contributors. Implementers and Web Masters can use the Managed Navigation elements to create new sections of the Site, move pages from one section to another, change the navigational label or image on a section, reorder sections, and if necessary remove sections from the navigation menu completely.

Managed Navigation relies on Site Folder Publishing to deliver the Items organized with a Site's set of folders. Managed Navigation will not function if you have not used a Site Folder to structure your content.

Using Managed Navigation

Managed Navigation comes fully configured and implemented in FastForward, so it is fairly simple to use. The most common tasks performed when maintaining Managed Navigation are

- Adding new Items to a Site;
- Removing Items from a Site;
- Resetting Navigation;
- Adding new Sections to a Site;
- Removing Sections from a Site;
- Publishing a Site.

Additionally, you may occasionally have to maintain individual Managed Navigation Content Items while maintaining your Site. Such tasks include:

- Assigning a Landing Page to a Navigation Item;
- Creating a NavImage;
- Assigning a NavImage to a Navigation Item;
- Reordering Navigation Sub Menus.

When initially creating a Site, you will need to

- Create a Publishing Site;
- Create a Site structure, including a Site root and Subfolders.

Navigation Communities

The administrative Communities shipped with FastForward are intended to organize and provide access to administrative Rhythmyx applications used by Web Masters, producers, or other administrators responsible for maintaining the structure and navigation of one or more Sites.

- Enterprise Investments Admin
- Corporate Investments Admin

Within either of these Communities, the Rhythmyx Administrator can design, build, and control a site's structure by adding, removing and editing Site Folders and their properties.

Navigation Reset

It is often necessary to use the *Nav Reset* action in the Action Menu when maintaining Site Folders or Managed Navigation. This Action flushes the cache on the Navigation Content Items. When maintaining Managed Navigation or Site Folder structures, existing configurations remain stored in the cache, and you must refresh the cashed data to ensure that Previews work correctly. Published content is not affected by this cache.

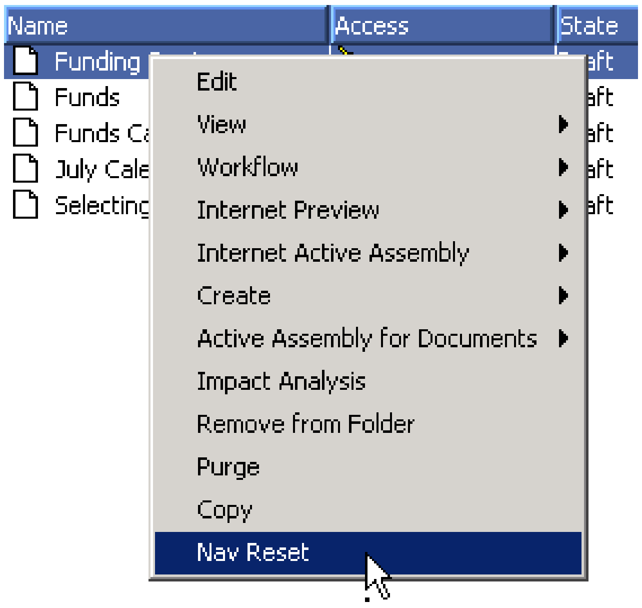


Figure 118: Resetting Navigation XML

The rule of thumb for determining whether to flush the Navigation cache is to flush it whenever you make a structural change to the navigation. Examples of such changes include:

- Editing navigation.properties;
- Dragging a sub folder to a new folder;
- Editing metadata in a Navon;
- Adding new Sections that appear in the navigation.

To flush the navigation cache:

1. Select any Item within any Site Folder in a Site where structural changes have been made to the navigation.
2. Right click the Item and select [Nav Reset].
3. Press [OK] in the reset pop up dialog.

Note: Clearing the navigation cache will cause a slow down in subsequent previews of Content Items until the navigation information is once again cached.

Setting up a Site

Setting up a Site entails two main tasks:

- Registering a Publishing Site
- Setting up Site Folders
    - Define a Site Root Folder
    - Build Site Structure

Creating Site Structures in Rhythmyx

The first step to configuring Managed Navigation is to create a Site Folder structure. A Site Folder structure consists of a Site root and any Subfolders added to it. Typically, the Folder structure of a Site in Rhythmyx mirrors the physical directory structure of the Web site. The Site Folders setup in the Content Explorer represent the virtual site as used and previewed by Content Contributors.

FastForward ships with two pre-defined Site Folder Structures, Enterprise and Corporate Investments.

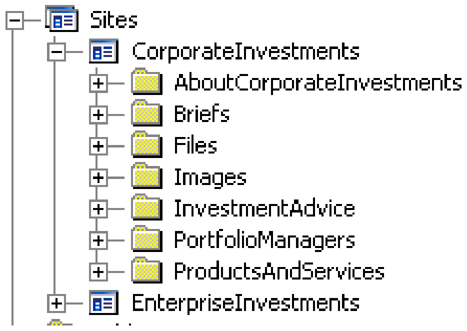


Figure 119: Default FastForward Sites

You can create additional Sites by adding new Site Folder root directories and populating them with content.

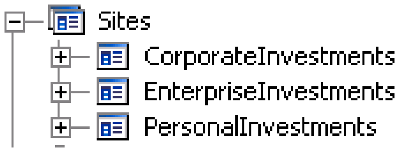


Figure 120: Adding a New Site

Copying a Site

Copying a Site often entails two tasks: Copying the Site structure and Content and Copying the Site registration and publishing details. We will, for this example assume that a new Site, Personal Investments, will be a copy of the existing Enterprise Investments Site. Several sub sections of the site will not be needed, and one section will need to occur in two different sections of the PI site (for navigation purposes). We will assume that a new Community set, Personal Investments and Personal Investments Admin will have been already created to map the new content into.

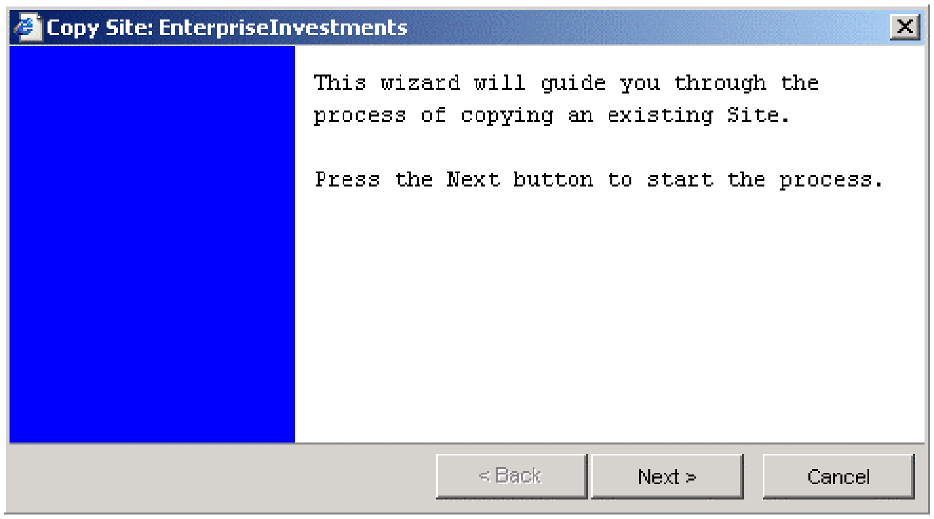
Copying the Site Structure and Content

1 Log into the Content Explorer.

2 Select the root Site Folder for Enterprise Investments, right click it and select [Copy].

3 Select the Sites root, right click it and select [Paste].

The Site Copy wizard will appear.



This wizard will guide you through the process of copying an existing Site.

Press the Next button to start the process.

Figure 121: Site Copy Wizard

4 Select [Next].

5 Give both the Site and Site folder a name. For convenience, it is best to have these names match. Note that unless configured to do otherwise, the Site Folder name will be used when publishing the physical Site. In addition, select the Site definition to copy when building this new Site. A Site Folder must have a matching Publishing Site registered to qualify the folder as a Site Folder.

6 Select [Next].

7 Decide whether to copy the Site Structure, Site Structure and Site Navigation, or Site Structure, Site Navigation, and Site Content. In our example, we will copy all three piece of the Site.

8 Select [Next].

9 Decide whether the Site Content should be copied as Links or New Copies. When copying as a link, a pointer is created to the original Item and any edits to the original cascades to the linked copy. When copying as a New Copy, a new copy of the Item is created and a unique Content Id is associated with the Item. During a New Copy, all associated relationships held by the original Item are maintained. Additionally, all New Copy Items will be placed in their respective Workflow's initial State (often named Draft). For our example we will create New Copies of all the Content Items.

10 Select [Next].

11 Map the soon to be copied Items (Folders and Content) to a Community. By default, the Items will be mapped to the current Community. If an alternate Community has been created, select the new target Community in the right hand column (or any other available Community for that matter). We will map the standard and administration communities to their matching Personal Investments and Personal Investments Admin peers.

12 Select [Next].

13 A summary dialog will be presented. If the list of pending events is confirmed, select [Finish] to complete the Copy. Press [Back] to modify and previous selections.

The Site Structure, Navigation, Content, and Publishing Site Registration are copied. An Edition defining this new Site is all that is need to publishing content from PersonalInvestment to a web application server. Refer to the product documentation on Publishing to create a New (of Copy) and Edition.

Copying a Site Sub-Section

A sub-section of a Site may be copied into another section within the originating Site or to a different Site. Additionally, an entire Site can be copied and pasted into the sub-folders. Our example will copy a sub-section from the Enterprise Investments to the Corporate Investments Site.

1 Log into the Content Explorer.

2 Open the Enterprise Investments Site and locate the Home Equity folder under the Mortgages and Home Finance section.

3 Right click the Home Equity folder and select [Copy].

4 Open the Corporate Investments Site and locate the Investment Advice folder.

5 Right click the Investment Advice folder and select Paste [As New Copy].

The Copy Wizard is presented.

6 Select [Next].

7 Give the new Site Folder a name.

8 Select [Next].

9 Copy the Folders, Navigation, and Content.

10 Select [Next].

11 Copy Content as Links.

12 Select [Next].

13 Map the Content from Enterprise Investments to the matching set of Communities registered for Personal Investments from the Site Copying section of this document.

14 Select [Next].

15 A summary of the copying tasks is presented. Press [Finish] to complete the copy. Press [Back] if it is necessary to make any changes.

Rhythmyx copies the selected items and creates the new sub-section.

Note: If the section is to be represented in the Navigation by any images, the appropriate NavImage Items will need to be swapped out with those from the originating Site.

Creating a Site Folder for a New Site

The first step in defining a Site Folder hierarchy is to create the root Site Folder. This Folder serves as the highest point in the navigation hierarchy of a Site. Though you can define multiple Site Root Folders, each is the root of a unique Site. Any individual Site can have only one root.

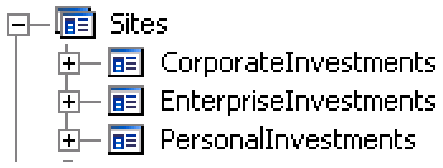


Figure 122: Adding a New Site

The root folder defines the anchor point for the Site's navigation scheme.

To create a Site Root Folder

1 Log into the Content Explorer.

2 Right click the Sites node in the Navigation Panel and select [New Folder...].

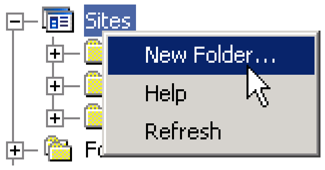


Figure 123: Creating a new Site Folder

Rhythmyx displays the Folder Properties dialog.

3 Fill in the fields for the new Folder accordingly.

4 Click the [**OK**] button to save the new Folder.

5 If you have not already created Site registration, do so now. The Site registration is required to publish the Site.

Creating a Site Folder in an Existing Site Tree

It is often necessary to add additional Site Folders to a Site. Users with administrative rights to a Site Root Folder can create Folders within the site.

Note: This activity assumes a Site and Navigation Items already exist.

To create a new Site Folder:

1 Expand the Site Tree until you find the Folder within which you want to create the new Site Folder.

2 Right-click the Folder and from the popup menu, choose *New Folder*.

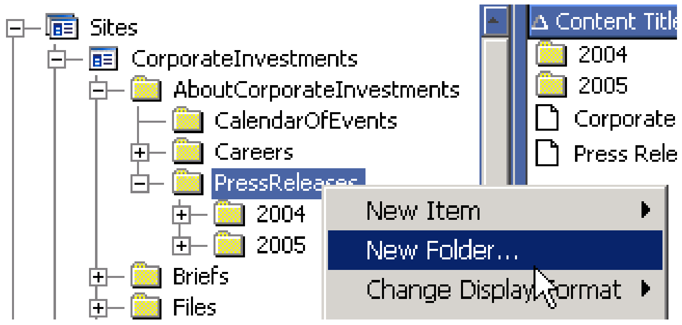
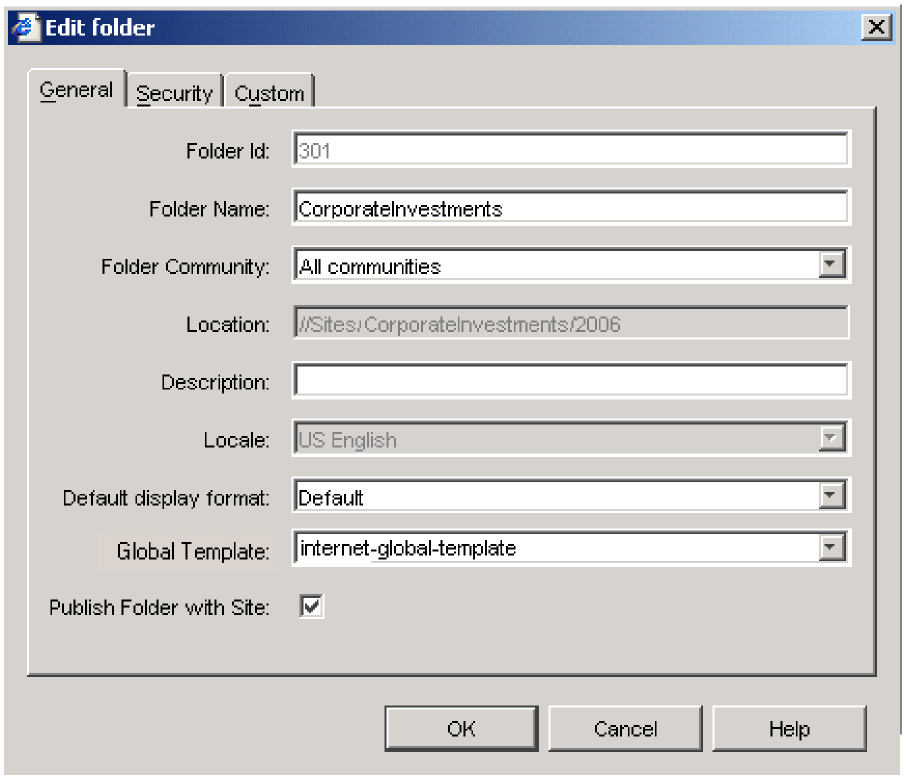


Figure 124: Creating a New Site Folder

3 Rhythmyx displays the Folder Properties dialog. Fill in each field.



Folder Id: 301
Folder Name: CorporateInvestments
Folder Community: All communities
Location: //Sites/CorporateInvestments/2006
Description:
Locale: US English
Default display format: Default
Global Template: internet-global-template
Publish Folder with Site: ✓

Figure 125: Create Folder General Tab

Folder Name - The name to be displayed in the Navigation Pane next to the Site Folder. This name will also be used to name the directory on the Publishing Site. If you want to use a different value for the name of the output directory, specify the custom property sys_pubFileName on the Properties tab. For the value of this property, specify the name you want for the output directory. If you change this value, you will need to initiate a Full Publish to reflect the results in the output directory. Incremental publishing will not pickup changes to this attribute.

Folder Community - Communities allowed to access the Folder. A Community not defined in this list will not be able to see the folder in the Content Explorer's Navigation Panel.

Description - A brief note describing the Folder.

Locale - The default Locale for the Folder.

Default Display Format - When a user selects the Folder, Items in the folder will be displayed using the specified Display Format.

4 Click on the Security Tab. Add or remove Users or Groups as necessary.

5 Click on the Properties Tab. This dialog allows the Folder Admin to assign Name Value pair properties to the Folder. Currently, sys_pubFileName is the only property available to users.

6 Click the [**OK**] button to create the new Site Folder.

7 Reset the Navigation.

If the new Folder is created within a defined Site Tree that contains a NavTree Item at its root, a Navon Item will be generated automatically in the new Folder. The new Navon will take on the Site Folder's name as its System Title.

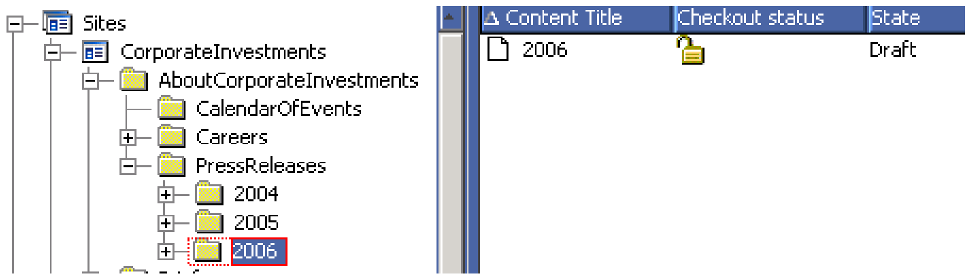


Figure 126: New Navon Created in a New Site Folder

The Navigation for the Site will now include a reference to the new Folder.

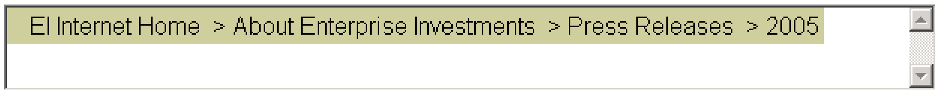

EI Internet Home > About Enterprise Investments > Press Releases > 2005

Figure 127: Breadcrumbs to New Navigation Section

Adding a new Folder within an Existing Site

In some cases, you may want to create a Folder that is not included in the Navigation hierarchy of the Site. For example, you may want to create a Folder specifically to store images for your Site. Such Folders are generally not included in navigation. To exclude a Folder from Navigation, purge the Navon created in the Folder when you create the Folder.

To create a Folder without navigation:

1. Create a new Folder as described in ***Creating a Site Folder in an Existing Site Tree*** (see "Adding a new Folder within an Existing Site" on page 168).

   If the Folder is created within a Site Tree that contains a NavTree Item at its root, a Navon Item will automatically be generated within the new Navigation Section.

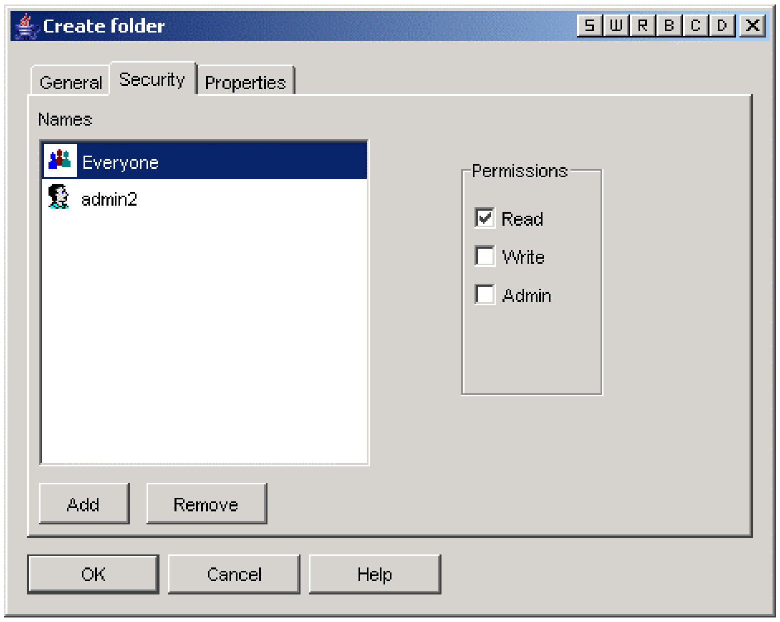
2. Open the new Folder, select the Navon Item, right click on it, and choose Delete from the popup menu. This action removes the Content Item from the Folder.

NOTE: It is strongly recommended that you purge this Navon Content Item. Stray Content Items contribute to clutter that can make your Suite structure difficult to navigate in Content Explorer. Also, if the Navon is somehow added back to the Folder, the Folder itself will be added back into the Site navigation.

Assigning Permissions to a Site Section

In addition to access control via assigned Community, you can also assign different permissions to users in the Folder Access Control List (ACL) itself. If a user can access a Folder, they can always see the Content Items in that Folder, but the user's rights to any specific Content Item are controlled by its current Workflow State.

To maintain permissions for a Folder, click the Security tab on the Folder Properties dialog.



Names
- Everyone
- admin2

Permissions
- [✓] Read
- [ ] Write
- [ ] Admin

Figure 128: Create Folder Security Tab

By default, the Everyone Group is issued Read permissions. This allows users not previously defined to have additional rights to see the contents of the new Folder. The user who created the Folder is issued Read, Write, and Admin rights to the Folder. This allows the user to see the contents of the Folder, add Items to the Folder, and change the Folder's properties. In addition, the Admin permission allows the user to delete the folder.

You can add permissions for additional users or Roles. Click the [Add] button and select the additional user or Role you want to add to the Folder ACL.

The following levels of security are available for Folders:

- *Read* - Permits users to view the folder and its contents. Does not allow users to move, copy, or add contents to the folder. Lets users copy but not move contents in the folder to another folder. Lets users view folder properties.
- *Write* - Permits users to view, copy, and move the folder, but not delete it, and to view, copy, move, or add contents. Lets users view folder properties.
- *Admin* - Gives users all Write permissions and enables them to delete the folder's sub-folders and to edit all folder properties.

If you have Write or Admin permission for a folder, when you right-click on the folder the option *New Folder* is available to you to create a Subfolder, and the option Properties lets you view or edit the selected folder's properties.

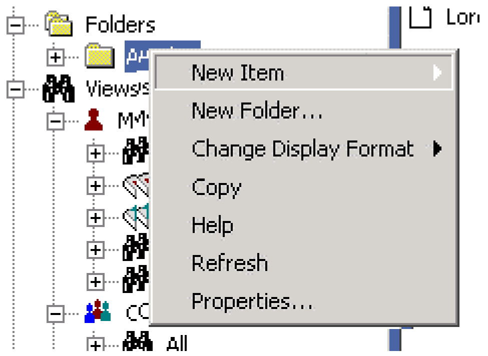


Figure 129: Folder actions menu

If you have read access to a folder, the *New Folder* option is not available to you and the option *Properties* lets you view the selected folder's properties.

The Community of the Folder also affects the user's access to it.

	Folder in User's Community	**Folder not in User's Community**
User not in Folder ACL	User cannot see Folder.	User cannot see Folder.

Reader Access	User can see Folder and Content Items in Folder.
	User cannot:
	create new sub-Folders in Folder;
	delete sub-Folders from Folder;
	create new Content Items in Folder;
	remove Content Items from Folder.
		User cannot see Folder.

Write Access	User can see Folder and Content Items in Folder. User can create new Content Items in Folder. User can remove Content Items from Folder. User can create new sub-Folders. User can delete sub-Folders.	User cannot see Folder.
Admin Access	User has all privileges of a user with Write access. In addition, the user can modify the Folder's ACL. NOTE: Any user with Admin access to the Rhythmyx server automatically has Admin access to all Folders, regardless of the Folder ACL.	User cannot see Folder.
Empty ACL	This is not a desirable setting. All users in the system will have Write access.	User cannot see Folder.

Content Items are always visible regardless of Community. Permissions for an individual Content Item (such as View, Modify, Purge) are based on the current Workflow State of the Content Item.

Editing a Site Section

Users with Admin rights to a Folder can modify its properties. All fields and properties can be modified except Locale, which is locked once the Folder has been created.

To edit a Folder's properties:

1 Log into the Content Explorer.

2 Right click on the Folder whose properties you want to modify and from the popup menu, choose *Properties*.

3 Modify any Folder Properties as needed.

Deleting a Site Folder Section

When you delete a Site Folder, the association between the Folder and its contents is broken. The Content Items are not purged, but remain in the system, and in any other Folder with which they are associated.

Note that any Navigation Items in the folder will need to be associated with a different Site Folder or deleted. Orphaned Navigation Items can cause problems if they are assigned to another Folder unintentionally.

To delete a Site Folder:

1 Log into the Content Explorer.

2 Find the Site Folder you want to delete.

3 Move a Content Items to another Folder, and remove or delete any Managed Navigation Content Items from the Folder.

4 Right click the Site Folder and on the popup menu, choose *Remove from folder*.

Rhythmyx deletes the Folder.

Adding Navigation to Rhythmyx Site Folders

Once you have created Site Root Folder and any Site Subfolders, you can add Managed Navigation Content Items to provide Navigation for your site. You can then assign landing pages to NavTree and Navon Content Items, and fine tune navigation submenus.

You can start adding Navigation by adding a NavTree Content Item to the Site Root Folder. You can add this Content Item either before or after creating Site Subfolders.

Creating a NavTree

NavTree Content Items reside in the root Site Folder for a given Site. The NavTree Content Item denotes the root of a Site and is responsible for propagating Navons through the sys_navFolder Effect. This effect generates Navon Items in the Site Folders below the root and establishes relationships between each of these Items. The result is a navigation hierarchy linking all levels of the Site together.

To create a NavTree:

1. Log into the Content Explorer.
2. Locate the Site Folder Root where you want to create the NavTree. Note that this Folder cannot already have a NavTree Content Item. If a Site Root Folder already contains a NavTree Content Item, Rhythmyx will generate an error when you try to add the new NavTree Content Item to the Repository.
3. Right click the Site Folder and from the popup menu choose *New Item* > *NavTree*.
4. Fill in the data for the Content Item.
5. Checking the "Propagate" check box to add Navons to any existing Site Subfolders. Rhythmyx will always add Navons to new Site Subfolders you add after adding a NavTree to a Site Root Folder.
6. Click the [**Insert**] button to add the NavTree to the Repository, then close the Content Editor.

Creating a Navon

Navon Content Items are generally created automatically but the sys_navFolder Effect. In some cases, however, you may need to create Navons manually. Once you create a Navon, the sys_navFolder Effect will create all necessary relationships between the new Navon and existing Items.

To create a Navon:

1. Log into the Content Explorer.
2. Open the Site Folder where you want to create the Navon.
3. Right click the Site Folder and from the popup menu, choose *New Item* > *Navon*.
4. Rhythmyx displays the Navon Content Editor.
5. Enter data in all of the Content Editor fields. There is no reason to check the **Propagate** checkbox when creating this Item.

6 Insert the Navon Content Item and close the Content Editor.

Assigning a Landing Page to a Navon

Navons represent the Folder in which they reside in any piece of navigation (breadcrumb, top navigation, site map, etc). When a site visitor selects a link in a piece of navigation, they are directed to a particular page, referred to as the "landing page". You must manually associate a landing page with each Navon.

To assign a landing page to a Navon:

1 Navigate to the Navon to which you want to add the landing page.

2 Right click the Item and from the popup menu choose *Active Assembly Table Editor*. (If the Navon is already Public, you may have to Transition it to the Quick Edit State before this option is available.)

Rhythmyx displays the Active Assembly Table Editor for the Navon.

3 Click on the nav_landing_page link.

Rhythmyx displays the Active Assembly Search dialog.

4 Find the Content Item you want to assign as the landing page for the Navon, check the box for that Content Item and click the [**Link to Slot**] button.

5 Close the Active Assembly Table Editor

6 If the Navon was edited in a Public State, Workflow the Item back to the Public State.

Removing a Landing Page from a Navon

A Navon should only have one Content Item in its landing page Slot. If you want to assign a different landing page for the Navon, you should remove the existing landing page before assigning a new one.

To remove the landing page from a Navon:

1 Locate the Navon from which you want to remove the landing page.

2 Right-click on the Navon and from the popup menu choose *Active Assembly Table Editor*.

Rhythmyx displays the Active Assembly Table Editor.

3 Click the delete button next to the Content Item that is currently assigned to the landing page Slot.

Rhythmyx processes your request and returns you to the Active Assembly Table Editor.

4 You can add the new landing page immediately if you link.

5 Close the Active Assembly Table Editor.

Assigning a NavImage to a Navon

Once you have created a NavImage, you can assign it to a Navon. NavImages are assigned to the nav_image Slot on the Navon. In the default FastForward installation, NavImages are used in the top_nav Variant to represent the Navon's Folder, and are a hotspot link to the Navon's landing page.

The top_nav Variant is the only Navigation Variant in the default FastForward implementation that supports images, but you can modify other Variants to support images as well. You could also implement new Variants to support combinations of images, text, and Flash.

Note that you can use essentially the same procedure to assign a NavImage to a NavTree.

To assign a NavImage to a Navon:

1 Log into the Content Explorer.

2 Locate the Site Folder containing the Navon to which you want to assign the NavImage.

3 Right-click the Navon and from the popup menu choose *Active Assembly Table Editor*. (If the Navon is already Public, you may have to Transition it to the Quick Edit State before this option is available.)

   Rhythmyx displays the Active Assembly Table Editor for the Navon.

4 Click on the nav_image link.

   Rhythmyx displays the Active Assembly Search dialog.

5 Find the NavImage Content Item you want to assign to the Navon.check the box for that NavImage, and click the [**Link to Slot**] button.

6 Close the Active Assembly Table Editor

7 If the Navon was edited in a Public State, Workflow the Item back to the Public State.

Removing a NavImage from a Navon

By default, removing a NavImage from a Navon eliminates representation of the Navon in the top navigation. The Navon will continue to render links in all other navigation elements.

Note that you can use essentially the same procedure to remove a NavImage from a NavTree.

To remove a NavImage from a Navon

1 Locate the Navon with the desired NavImage.

2 Right-click on the Navon and from the popup menu choose *Active Assembly Table Editor*.

   Rhythmyx displays the Active Assembly Table Editor.

3 In the nav_image Slot, click the delete button for the NavImage you want to delete.

Note: The nav_image Slot may contain multiple images. Be sure to delete only the desired NavImage from the Slot. Use the preview icon to the right of the NavImages to confirm selection of the image.

   Rhythmyx processes your request and returns you to the Active Assembly Table Editor.

4 Close the Active Assembly Table Editor.

In the default FastForward implementation, the nav_HomeImage Snippet is used only on Home Page Content Items. You can modify this Variant so you can apply it to other pages.

The nav_HomeImage Snippet is a Variant of Navon. This variant produces a title link to the page placed in the Navon's nav_landing_page Slot.

About Enterprise Investments

Figure 130: nav_HomeImage without Image

In addition, this Snippet has a Slot, titled Home Image, to which you can assign a NavImage. This image also becomes a hyperlink to the Landing Page. The combination of these two Snippets yields the nav_HomeImage Snippet.

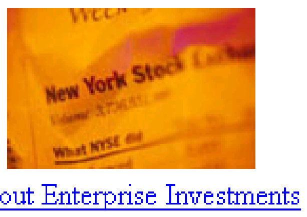


Figure 131: nav_HomeImage Snippet

Once assembled, this Snippet (along with a few others) is placed onto the Home Page and is used as a supplemental graphical navigation.

To assemble a nav_HomeImage Snippet:

1 Log in to Content Explorer and navigate to the About Enterprise Investments Folder.

2 Create the NavImage Content Item in this Folder, adding the image you want to use in the nav_HomeImage Snippet. In our example, we are using the NYSE Paper NavImage from the FastForward Sample Content.

3 Locate the Navon for the Site Section to which you want to link. In our example, we are using the About Enterprise Investments Navon.

4 Right click on the Navon and from the popup menu choose *Active Assembly Table Editor*.

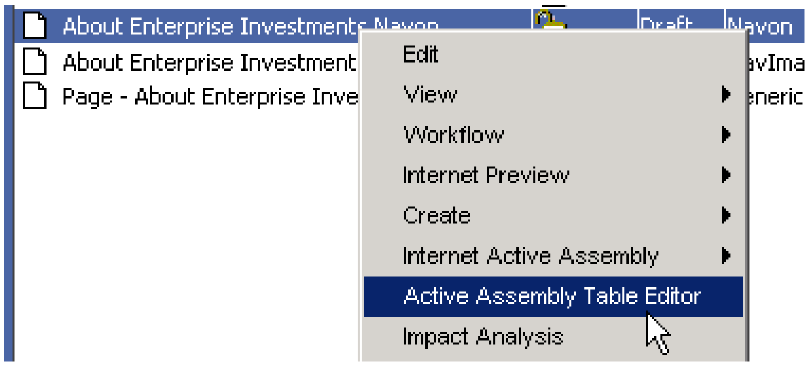

About Enterprise Investments Navon
About Enterprise Investment
Page - About Enterprise Inve
Draft
Navon
avIma
eneric
Edit
View
Workflow
Internet Preview
Create
Internet Active Assembly
Active Assembly Table Editor
Impact Analysis

Figure 132: Opening the Table Editor for a Content Item

5. Locate the Home Image Slot and add the newly created About Enterprise Investments Image to the Slot.
6. Close the Active Assembly Table Editor.

This produces a Snippet illustrated above. This Snippet will be added to the top_nav Slot and when clicked directs the user to the About Enterprise Investments Folder.

Creating a NavImage

NavImage Content Items support the use of images in navigation. If you want to represent a section of your site with an image, you must create a NavImage Content Item and include it as related content in the nav_image Slot on the NavTree or Navon Content Item. It is usually easier to find a NavImage if it resides in the same Site Folder as the Navon or NavTree that uses it, but it is not required to reside there.

A NavImage Content Item requires an Image file. You must assign this file to the NavImage.

If any given navigation element uses text links instead of images, you do not need to create a NavImage for it.

To create a NavImage

1. Log into the Content Explorer.
2. Locate the Site Folder where you want to create the NavImage.
3. Right click the Site Folder and from the popup menu, choose *New Item* > *NavImage*.

   Rhythmyx displays the NavImage Content Editor.
4. Fill in the fields for the new NavImage Content Item.
5. Insert the Item and close the Content Editor.

Maintaining a Site

Once you have defined your Site structure and added navigation, you can add content. You can either move existing Content Items into the Site Folders, or you can create new Content Items in the Folders. In fact, it is common for both tasks to occur when setting up a Site. Neither method has any advantages nor disadvantages over the other. The choice between the two tasks is merely a matter of convenience.

Adding Content to a Site Folder

When a Content Item does not yet exist, you should add it to a Site Folder when you create it. Creating Content Items directly in the Site Folders eliminates the need to drag them from Views or search results into the Folders.

To create a Content Item in a Site Folder:

1. Log into the Content Explorer.
2. Locate the Site Folder where you want to create the Content Item.
3. In the Navigation pane, right-click on the Folder where you want to create the Content Item and from the popup menu choose *New Item* > *<Content Type>* where "Content Type" is the name of the Content Type of the new Content Item you want to created.

   Rhythmyx displays a Content Editor for the specified Content Type.
4. Enter the data for the Content Type and click the [**Insert**] button to add the Content Item to the Repository.
5. Close the Content Editor.

   Your new Content Item will be added to the Folder you specified. It is not restricted to this Folder. You can move it or place additional links to it in other Folders as well.

Assigning a Content Item to Site section

If a Content Items already exists, you can move it or add it to other Site Folders, including Folders under different Site Roots. Publishing each Site creates a distinct page from the Content Item for that Site, formatted using the Global Template specified for that Site.

To move a Content Item from one Folder to another:

1. Log into the Content Explorer.
2. Locate the Content Item through any standard View or Search.
3. Expand the Site Folder for the target Site and locate the target Site Folder.

4 Left click and drag the Content Item into the desired Site Folder.

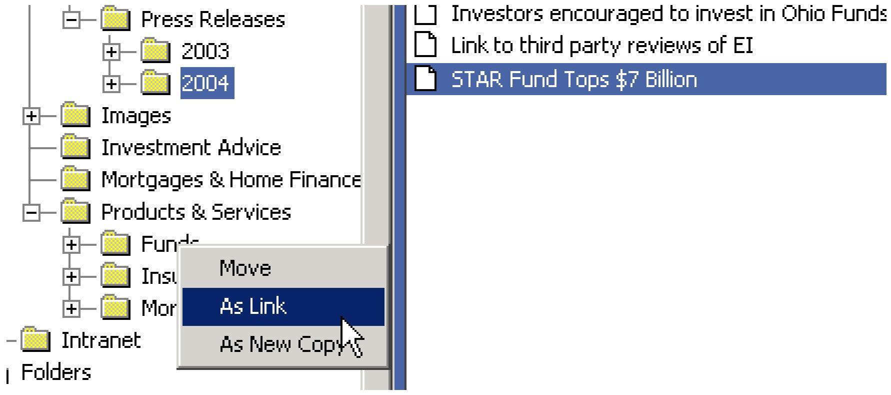


Figure 133: Dropping Item As Link in a Site Folder

5 When you release the mouse button, Rhythmyx displays a popup menu with three options:

- Move - This option moves the Content Item from its current Folder to the new location.
- As Link - This option creates a link to the Content Item in the specified location.
- As New Copy - This option creates a new Copy of the Content Item. This new Copy has its own Content ID and exists independent of the original Content Item. It also begins in the Draft State in the Workflow.

Note that, if you want to create links from Content Items under one Site node in Rhythmyx to Content Items under another Site node, you must include the following columns in the Display Format: sys_siteid and sys_folderid"

Locating an Item with Impact Analysis

You can determine which Folders currently contain a Content Item by specifying the Folder Content Relationship in Impact Analysis.

To view the Folder associations for a Content Item:

1 Log into the Content Explorer.

2 Locate whose locations you want to find. You can use a View, a search, or find it in a Folder if you know at least one current location.

3 Right click on the Item and from the popup menu choose select *Impact Analysis*.

4 Rhythmyx displays the Impact Analysis window for the Content Item. The default value in the **Relationship** field in Impact Analysis is *Active Assembly*.

5 In the **Relationship** drop list, choose *Folder Content*.

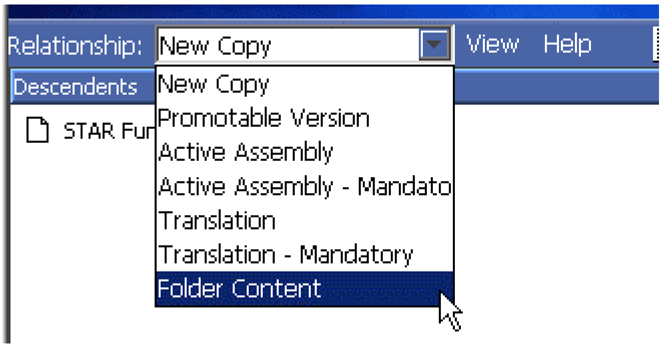
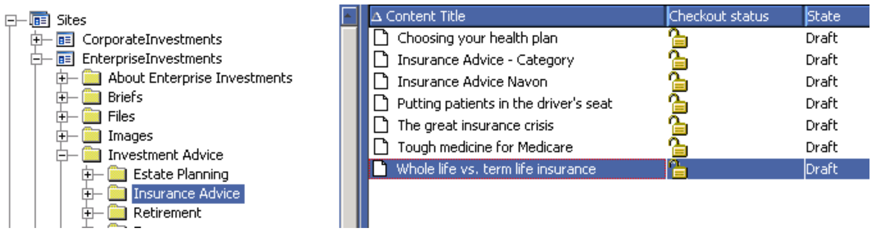


Figure 134: Viewing Folder Relationships with Impact Analysis

6 The Ancestors window displays all Folders containing the selected Item.

Locating an Item within a Site Tree

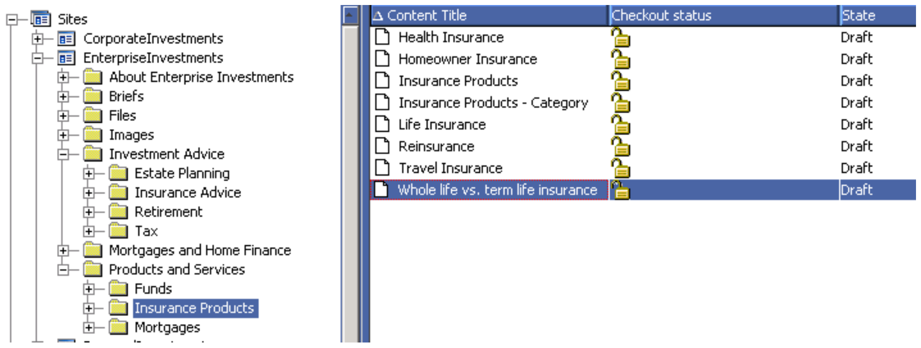
Content Items can be represented by multiple Item links within a Site Tree. Though each link may represent a different page in the published output, they all represent the same Content Item. Take the example of a single Item residing in both the Investment Advice > Insurance Advice folder



Content Title	Checkout status	State
Choosing your health plan	🔒	Draft
Insurance Advice - Category	🔒	Draft
Insurance Advice Navon	🔒	Draft
Putting patients in the driver's seat	🔒	Draft
The great insurance crisis	🔒	Draft
Tough medicine for Medicare	🔒	Draft
Whole life vs. term life insurance	🔒	Draft

Figure 135: Press Release in the 2004 Site Folder

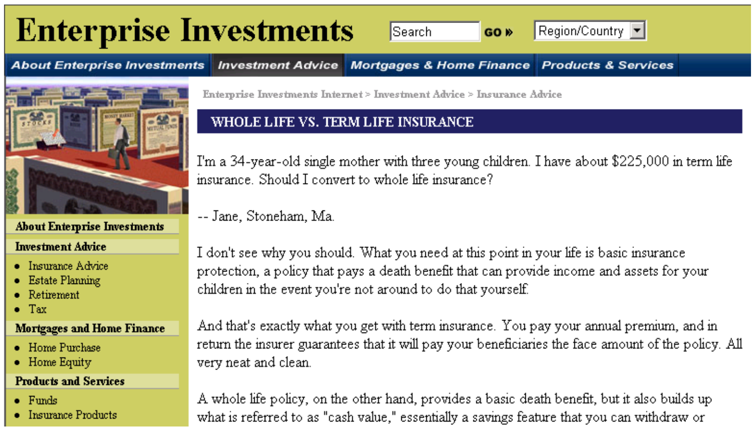
and the Products and Services > Insurance Products folder.



Content Title	Checkout status	State
Health Insurance		Draft
Homeowner Insurance	🔒	Draft
Insurance Products	🔒	Draft
Insurance Products - Category	🔒	Draft
Life Insurance	🔒	Draft
Reinsurance	🔒	Draft
Travel Insurance	🔒	Draft
Whole life vs. term life insurance	🔒	Draft

Figure 136: Press Release in Funds Site Folder

Both Folder Views target the same Content Item, but a preview of the Item in the Insurance Advice Site Folder



Enterprise Investments Internet > Investment Advice > Insurance Advice

WHOLE LIFE VS. TERM LIFE INSURANCE

I'm a 34-year-old single mother with three young children. I have about $225,000 in term life insurance. Should I convert to whole life insurance?

-- Jane, Stoneham, Ma.

I don't see why you should. What you need at this point in your life is basic insurance protection, a policy that pays a death benefit that can provide income and assets for your children in the event you're not around to do that yourself.

And that's exactly what you get with term insurance. You pay your annual premium, and in return the insurer guarantees that it will pay your beneficiaries the face amount of the policy. All very neat and clean.

A whole life policy, on the other hand, provides a basic death benefit, but it also builds up what is referred to as "cash value," essentially a savings feature that you can withdraw or

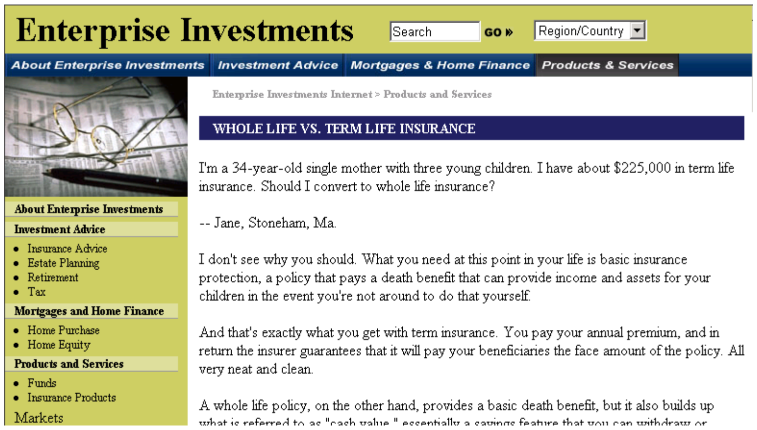
About Enterprise Investments
Investment Advice
- Insurance Advice
- Estate Planning
- Retirement
- Tax

Mortgages and Home Finance
- Home Purchase
- Home Equity

Products and Services
- Funds
- Insurance Products

Figure 137: Insurance Page in Insurance Advice Section

yields a different Item then the one previewed in the Insurance Products Site Folder.



Enterprise Investments Internet > Products and Services

WHOLE LIFE VS. TERM LIFE INSURANCE

I'm a 34-year-old single mother with three young children. I have about $225,000 in term life insurance. Should I convert to whole life insurance?

-- Jane, Stoneham, Ma.

I don't see why you should. What you need at this point in your life is basic insurance protection, a policy that pays a death benefit that can provide income and assets for your children in the event you're not around to do that yourself.

And that's exactly what you get with term insurance. You pay your annual premium, and in return the insurer guarantees that it will pay your beneficiaries the face amount of the policy. All very neat and clean.

A whole life policy, on the other hand, provides a basic death benefit, but it also builds up what is referred to as "cash value," essentially a savings feature that you can withdraw or

About Enterprise Investments
Investment Advice
- Insurance Advice
- Estate Planning
- Retirement
- Tax

Mortgages and Home Finance
- Home Purchase
- Home Equity

Products and Services
- Funds
- Insurance Products

Markets

Figure 138: Insurance Page in Insurance Products Section

Since these differences are significant, it is important to be able to find the correct instance of an Item when searching the system. Several methods are available to locate Content Items.

- You can manually traverse the Site Folder Tree to find each instance of a Content Item.
- You can use Impact Analysis to view the Folder Relationships of the Content Item,
- You can use the Duplicate Folder Paths View. This View returns a list of all Content Items that currently reside in more than one Site Folder.

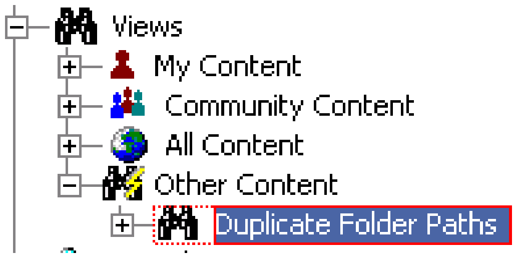


Figure 139: Duplicate Folder Path View

Removing a Content Item from a Site Section

Content Items that you remove from a Site Folder still exists in the system. You can thus temporarily remove a Content Item from a Web site.

Two options are available to remove a Content Item from a Site. You can either remove it from the Folder or you can move the Content Item into a new Workflow State in which it is no longer Public (Expire).

To remove a Content Item from a Folder, locate the Content Item you want to remove. Right-click on it and on the popup menu, choose *Remove from Folder*. This option only removes the Content Item from the specified Folder. If it exists in other Folders, it will still be there, and if Public, it will be published.

Users with the correct access to Content Items in the Public State can expire them, moving them out of the Public State into an Archive State. To expire a Content Item, right click on it and from the popup menu, choose *Workflow* > *Expire*. The Content Item will still exist in the Folder, but will be removed from the published Web site during the next publishing run.

Editing an Item in a Site Section

Items can be edited from any standard View or Folder. Items linked to Site Sections are edited like any other Content Item Managed by Rhythmyx.

Procedure

1 Locate the Item to be edited through any standard Views, using the Search Component, or manually locating it within a Site Folder.

2 Right click and select [Edit] ([Quick Edit] if the Item is currently in the Public State) or double click the Item.

3 Once edited, close the Item's Content Editor. If necessary, check the Item in or Workflow it back to the Public State if it was edited in a Public State originally.

Note: Since Item icons in Site Folders are simply links to the original Item, editing an Item in any location impacts all references to that Item.

Editing a Navigation Item

Navon, NavTree, and NavImage Items behave and can be edited like any other Content Item created in Rhythmyx.

1 Located the Navigation Item to be edited.

2 Double click the Item or select [Edit] or [Quick Edit] if the Item is in a Public State.

3 Edit the Item as needed.

4 [Update] and [Close] the Item.

5 If the Item was in a Public State prior to editing, Workflow the Item back to the Public State.

Editing a Public Navigational Element

Navigation Items behave the same as all other Item types when edited in a Public Workflow State. The following must be taken into consideration when editing any Item in such a State:

- Items edited in a Public State result in the creation of a new Revision of the Item;
- Items edited in a Public State must be manually transitioned back to the original Public State;
- Changes to an Item will not be reflected on the Published Site until the Edition containing the edited Item is run and the updated Item (or Items containing the Item) is re-published.

1 Log into the Content Explorer.

2 Locate the Item to be edited.

3 Double click or right click the Item and select [Edit] or [Quick Edit].

4 Edit the Item as needed.

5 [Update] and [Close] the Item.

6 If the Item was originally in a Public State ([Quick Edit] was selected), the Item will need to be manually transitioned back to this Public State.

Splitting Navigation Sections

As you add content to your Site, you may find that some Folders contain so many Content Items that they become unwieldy. In that situation, you may want to subdivide one Folder into several Subfolders. You may also want to subdivide a Folder to accommodate marketing needs, such as spinning off a new product line, or simply to make it easier for Content Contributors to manage the content assigned to them.

When you split a Folder, you can represent it as one or more subsections in the Site:

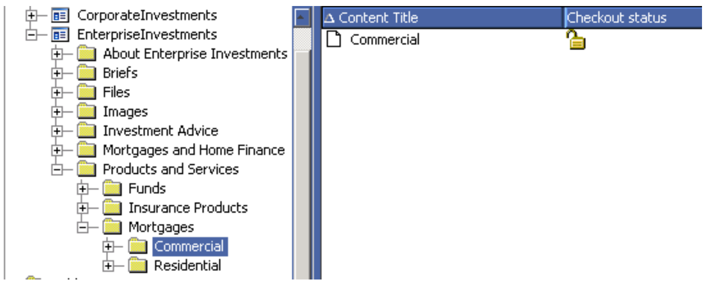
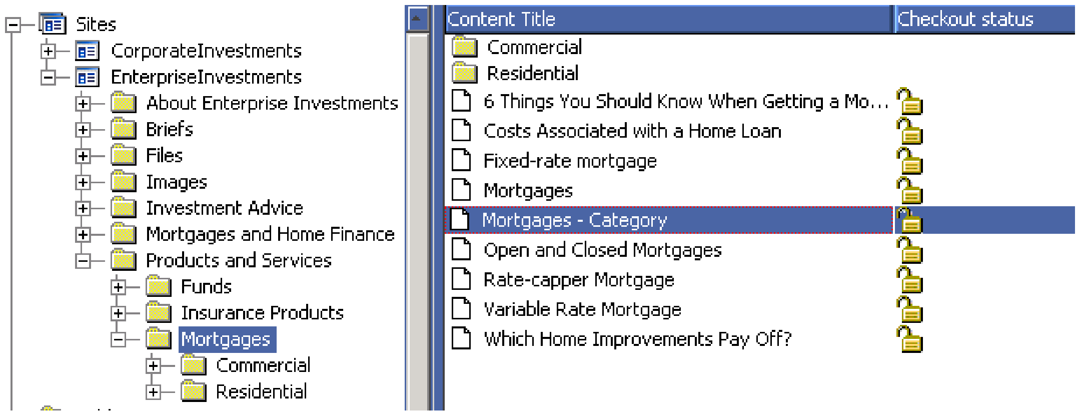


Figure 140: Splitting the Navigation Section

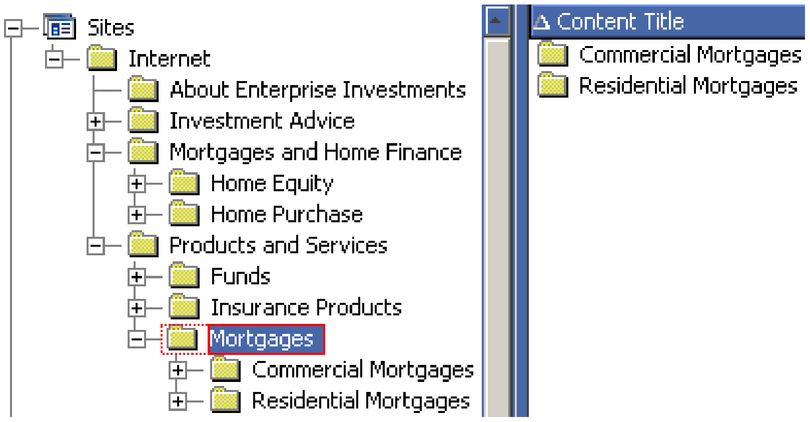
You can also choose to use a single Navigation Item to represent Content Items in multiple sub folders.



Content Title	Checkout status
Commercial	
Residential	
6 Things You Should Know When Getting a Mo…	
Costs Associated with a Home Loan	
Fixed-rate mortgage	
Mortgages	
Mortgages - Category	
Open and Closed Mortgages	
Rate-capper Mortgage	
Variable Rate Mortgage	
Which Home Improvements Pay Off?	

Figure 141: *Representing Two Folders with One Navigation Item*

Let us follow an example illustrating the first scenario. Assume we have decided to subdivide the current Mortgages section of our Web site into two new sections, Commercial Mortgages and Residential Mortgages. We will maintain the original Mortgages Site Folder and nest the two new sections within that folder.



Content Title
Commercial Mortgages
Residential Mortgages

Figure 142: Site Folder without a Navon

The Mortgages Folder, though, will not itself be represented in the navigation.

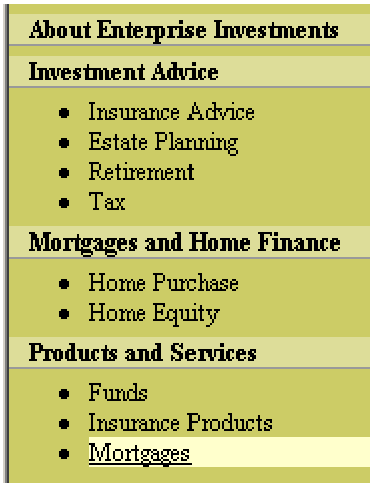


Figure 143: Navigation without Split Sections

To split a Site Folder:

1. Log into the Content Explorer find the Site section you want to Split (Mortgages in this case).
2. Create the necessary sub folders as descendants of the original Site Folder.

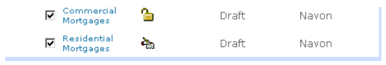
   In our example, we will create two new Folders, Commercial Mortgages and Residential Mortgages. Note that when we create this Folder, a Navon is added to them automatically.
3. Delete the Navon Item from the original Folder (from the Mortgages Folder in this case). Also delete or move any other Navigation Content Items, such as any NavImage Content Items in the Folder.
4. Drag and [Move] the Content Items from the Original Folder to the new descendant Folders.
5. Assign landing pages to the nav_landingpage Slot of each new Navon. You may need to modify the Content Item assigned as the landing page to reflect the new contents of the Folder.
6. Create any new NavImage Items for the new Navons and assign them to the appropriate nav_image Slot..

   At this point, you should have the original Site Folder (Mortgages) with no currently associated Content Items. The Mortgages Folder contains two Subfolders, Commercial Mortgages and Residential Mortgages. Each of these Folders contains a single Navon and several Content Items, and possibly one or more new NavImage Content Items.

   We want the navigation to skip the Mortgages Folder. We will have to add the new Navons to the nav_submenu of Navon in the Folder that contains the Mortgages Folder (which is Products and Services in our example).
7. Open the Products and Services Folder and locate its Navon.
8. Right-click on the Navon and from the popup menu choose *Active Assembly Table Editor*.

   Rhythmyx displays the Active Assembly Table Editor.
9. Click the nav_submenu link and search for Content Items with the word "mortgages" in the title.

10 Check the boxes for the Commercial Mortgages and Residential Mortgages Navons, then click the [**Link to Slot**] button.



✓	Commercial Mortgages		Draft	Navon
✓	Residential Mortgages		Draft	Navon

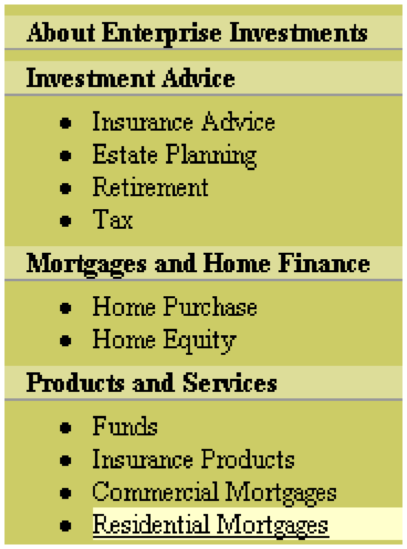
Figure 144: Adding Navon Items to the nav_submenu Slot

11 Close the Active Assembly Table Editor.

12 Transition the new Navons the Public State.

13 Reset the navigation.

14 Preview the landing pages in each new descendant section. The navigation should not show the old Mortgages Section, but instead display each of the new descendant sections.



About Enterprise Investments

Investment Advice

- Insurance Advice
- Estate Planning
- Retirement
- Tax

Mortgages and Home Finance

- Home Purchase
- Home Equity

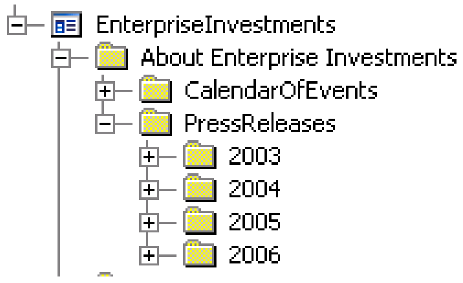
Products and Services

- Funds
- Insurance Products
- Commercial Mortgages
- Residential Mortgages

Figure 145: Navigation with New Split Sections

Merging Navigation Items

It is no less common to merge sections of a Web site than to split them. When you merge several Folders, you also need to merge their Navigation Content Items as well. For example, originally, we organized Press Releases by year:



```
EnterpriseInvestments
  About Enterprise Investments
    CalendarOfEvents
    PressReleases
      2003
      2004
      2005
      2006
```

Figure 146: Press Releases by Year

After gathering five years worth of Items, we decided that any Press Release two years or older would be managed in an Archives Site Folder.

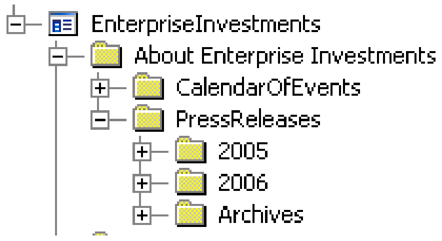


Figure 147: Press Releases by Year with Archive Folder

These Items would be represented by a single Navigation Item and sorted with an Auto Index by creation date. So we need to merge the originally separate yearly press releases Folders into the Archives Folder.

To merge Folders and navigation:

1. Log into the Content Explorer.
2. Create a new Folder to merge the existing Folders. In our example, we will create an Archives Folder. When we create the new Folder, a new Navon is created automatically. If desired, you can also create and associate a NavImage Content Item as well.
3. Move the necessary Content Items from the old Site Sections to the newly created Site Folder by selecting them, dragging them into the new Site Folder and choosing *Move* from the popup menu.
4. Specify a Landing page for the new Navon.
5. Delete the now stale Navigation Items from the old Site Folders.
6. Delete the now empty Site Folders.
7. Reset the Navigation.

Reordering a Submenu

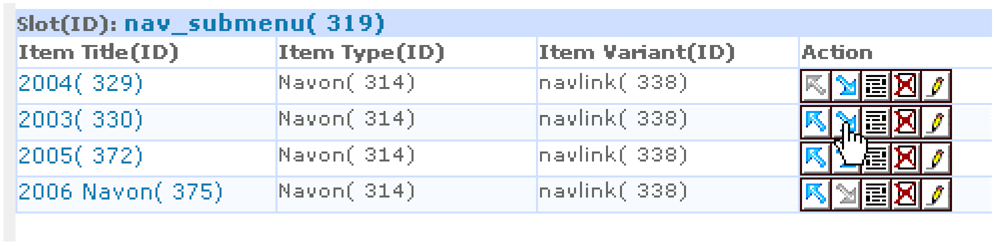
By default, when a Navon or NavTree Item is created, the nav_submenu slot is populated with links to the Navon Items in the directories immediately below the current one. These Relationships build a list of links to the Subfolders contained in a Folder. In the published output, when you choose certain navigation links (such as the left navigation), it expands to show links to the subsections. This list of sub menu Items is assembled in the order that the Folders appear in the Navigation pane of Content Explorer. However, you may want to modify this order.

To reorder a submenu:

1. Locate the Navon whose submenu you want to reorder.
2. Right-click on the Navon and from the popup menu choose *Active Assembly Table Editor*.

   Rhythmyx displays the Active Assembly Table Editor.

3 Use the up and down arrow icons to the right of the Navon Items displayed in the nav_submenu Slot to adjust the order of the Navons.



Slot(ID): nav_submenu(319)

Item Title(ID)	Item Type(ID)	Item Variant(ID)	Action
2004(329)	Navon(314)	navlink(338)	
2003(330)	Navon(314)	navlink(338)	
2005(372)	Navon(314)	navlink(338)	
2006 Navon(375)	Navon(314)	navlink(338)	

Figure 148: Reordering Navons in the nav_submenu Slot

4 Close the Active Assembly Table Editor.

Defining a Global Template for a Site Section

When creating a new Publishing Site, the default Global Template for the entire Site is defined. It is possible, though, to override the default Site Global Template with a Site Section Global Template. When defining a Global Template for a Site Section, all pages within the Section, including those in its sub folders, will inherit the override selection.

1 Log into the Content Explorer.

2 Right click the desired Site Folder where the override will begin.

3 Select [Properties].

4 Change the value associated with the Global Templates field to equal the name of the override Global Template.

5 Select [OK] to save changes.

Note: This activity assumes the override template has already been created.

Customizing Navigation Look and Feel

The look and feel of page navigation can be affected in two specific ways:

- Creation of a new Navigation Template;
- Manipulation of Navigation by CSS.

Both of these approaches also include their own avenue for selection during the generation of the Navigation. Themes allow the Web Administrator to have several entirely different Navigation template used for a single menu depending on its location within a Site (The Funds section can use the template defined by Theme A while the Mortgage section uses Theme B). This pick and choose approach can also be applied to the design of multiple CSS using Variable Selectors (The Funds section can use the CSS defined by Variable Selector A while the Mortgage section uses Variable Selector B). Additionally, these two approaches can be used in conjunction with one another to yield a myriad of output Navigation menus.

Customizing Navigation CSS

All installations will need to adapt the look and feel of the navigation to conform to the overall site design. The default navigation templates rely heavily on CSS for look and feel. FastForward ships with a CSS file named, rxs_styles.css. This CSS file contains sections for each of the navigation bars that ship with the system including nav_left, nav_top, and the like. This file is located in the \web_resources\rxs_nav\css directory.

These sections can be easily modified to produce the correct fonts, colors and indents.

The nav_left Slot that will ultimately hold the nav_left Snippet is wrapped with a <td> or similar container tag with the left_nav attribute defined.

```
    <td id="left_nav">
    <!-- start slot nav_left -->
    <!-- start snippet wrapper -->
    <span slotname="nav_left" psxeditslot="no"/>
    <!-- end snippet wrapper -->
    <!-- end slot nav_left -->
</td>
```

A sample from the nav_left.xsl template builds the contextual reference for the individual Item.

```
    <xsl:template match="navon[@relation='self']" priority="5"
mode="sectionImage">
       <!-- processes the image at the top of the Nav -->
       <xsl:apply-templates select="document(image-list/image-
link[@selector='section'])" mode="down"/>
    </xsl:template>
    <xsl:template match="navimageinfo" mode="down">
       <!-- processes the image at the top of the Nav -->
       <xsl:variable name="img_name" select="concat('img_',@contentid)"/>
       <img alt="{displaytitle}" class="sectionImage">
          <xsl:attribute name="src"><xsl:value-of
select="activeimage"/></xsl:attribute>
          <xsl:attribute name="id"><xsl:value-of
select="$img_name"/></xsl:attribute>
```

```
        </img>
    </xsl:template>
```

The resulting HTML file this template renders contains the appropriate relationship between the current Item and nearby Items:

```
...
        <navon absolute-level="1" name="About Corporate Investments Navon"
relation="self" relative-level="0" sys_contentid="329" sys_revision="1">
            <label>About Corporate Investments</label>
            <landing-page
pssessionid="5deaed6a4d87d443dc8c148d3cc4e28ef18fe7a5" sys_authtype="0"
sys_contentid="335" sys_context="0" sys_revision="1" sys_siteid="301"
sys_variantid="313">http://127.0.0.1:9999/Rhythmyx/rxs_Shared_cas/s_shar
ed.html?shared_variantid=4&amp;sys_revision=1&amp;sys_siteid=301&amp;sys
_authtype=0&amp;sys_contentid=335&amp;sys_variantid=313&amp;pssessionid=
5deaed6a4d87d443dc8c148d3cc4e28ef18fe7a5&amp;sys_context=0</landing-
page>
            <info-link sys_contentid="329" sys_revision="1"
sys_variantid="347">http://127.0.0.1:9999/Rhythmyx/rxs_Navon_cas/navon.x
ml?sys_revision=1&amp;sys_siteid=301&amp;sys_authtype=0&amp;sys_contenti
d=329&amp;sys_variantid=347&amp;pssessionid=5deaed6a4d87d443dc8c148d3cc4
e28ef18fe7a5&amp;sys_context=0</info-link>
            <image-list>
                <image-link sys_contentid="469" sys_revision="1"
sys_variantid="346">http://127.0.0.1:9999/Rhythmyx/rxs_NavImage_cas/NavI
mageInfo.xml?sys_revision=1&amp;sys_siteid=301&amp;sys_authtype=0&amp;sy
s_contentid=469&amp;sys_variantid=346&amp;pssessionid=5deaed6a4d87d443dc
8c148d3cc4e28ef18fe7a5&amp;sys_context=0</image-link>
                <image-link selector="section" sys_contentid="482"
sys_revision="1"
sys_variantid="346">http://127.0.0.1:9999/Rhythmyx/rxs_NavImage_cas/NavI
mageInfo.xml?sys_revision=1&amp;sys_siteid=301&amp;sys_authtype=0&amp;sy
s_contentid=482&amp;sys_variantid=346&amp;pssessionid=5deaed6a4d87d443dc
8c148d3cc4e28ef18fe7a5&amp;sys_context=0</image-link>
            </image-list>
            <navon absolute-level="2" name="2004" relation="descendent"
relative-level="1" sys_contentid="498" sys_revision="1">
                <label>2004</label>
                <landing-page
pssessionid="5deaed6a4d87d443dc8c148d3cc4e28ef18fe7a5" sys_authtype="0"
sys_contentid="502" sys_context="0" sys_revision="1" sys_siteid="301"
sys_variantid="313">http://127.0.0.1:9999/Rhythmyx/rxs_Shared_cas/s_shar
ed.html?shared_variantid=4&amp;sys_revision=1&amp;sys_siteid=301&amp;sys
_authtype=0&amp;sys_contentid=502&amp;sys_variantid=313&amp;pssessionid=
5deaed6a4d87d443dc8c148d3cc4e28ef18fe7a5&amp;sys_context=0</landing-
page>
                <info-link sys_contentid="498" sys_revision="1"
sys_variantid="347">http://127.0.0.1:9999/Rhythmyx/rxs_Navon_cas/navon.x
ml?sys_revision=1&amp;sys_siteid=301&amp;sys_authtype=0&amp;sys_contenti
d=498&amp;sys_variantid=347&amp;pssessionid=5deaed6a4d87d443dc8c148d3cc4
e28ef18fe7a5&amp;sys_context=0</info-link>
            </navon>
...
```

Of course, the implementer is free to create an entirely different convention for the CSS, as long as all page variants conform to the convention. As such, simple changes to the cascading style sheet allow for a great deal of control over the look and feel of the navigation in all its forms.

The class values defined in the previous code segment correspond to style classes found in the rxs_styles.css document.

```
#left_nav {
   top:0; left:0; width:180px;
   background:#cc6;}

#left_nav h3 {
   width:100%;
   display:block;
   padding:0;
   font-size:12px; font-weight:bold;
   line-height:15px; margin:0; margin-top:5px;}

#left_nav h3 a {
   width:100%;
   display:block;
   padding-left:10px;
    color:#000; background-color:#dd9; text-decoration:none;
   border-bottom:1px solid #999;}

#left_nav h3 a:hover {
   width:100%;
   display:block;
   color:#ffc;
   background-color:#320;}

#left_nav ul {
   width:100%; display:block;
   padding-left:24px; margin:4px 0;
   font-size:12px;}

#left_nav li { width:100%;}

#left_nav li a:link, #left_nav li a:visited {
   width:100%; display:block;
   text-decoration:none;
   color:#000;}

#left_nav li a:hover {
   text-decoration:underline;
   color:#000; background-color:#ffc;}
```

Using Variable Selectors

Context Variables are defined and applied to HTML templates to allow different values to be populated into a page depending on Context (preview, publish, etc.). These values are generally used to define the location of static, non-managed design resources such as CSS and images.

Variable Selectors allow Web Masters a greater amount of flexibility when working with Context Variables within a Site. Once the necessary Context Variables are defined, the Web Master has the ability to:

- Define a single variable to be used throughout an entire Site;
- Define different Variables to different Site Sections.

As an example, our Corporate Investments Internet Site can be using a single set of static images and CSS for all Site Sections except for the Products and Services section (and all sub-sections below it). This may be necessary as this section is maintained by a completely different Web Master than the other sections.

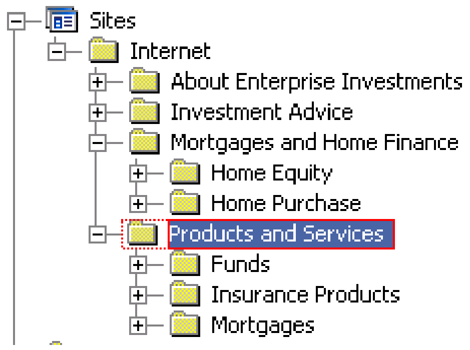
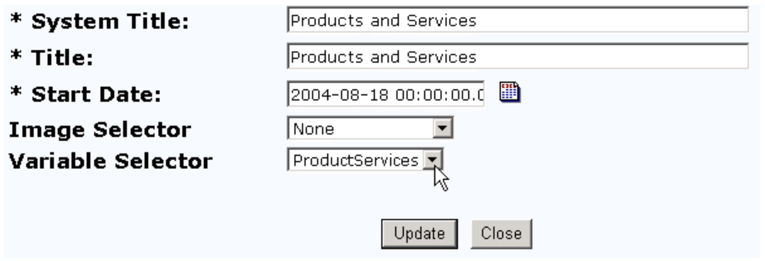


Figure 149: Products and Services Site Sub Section

By selecting a different Variable Selector for the Navon in the Products and Services section,



* System Title:	Products and Services
* Title:	Products and Services
* Start Date:	2004-08-18 00:00:00.0
Image Selector	None
Variable Selector	ProductServices

Figure 150: Defining an Alternate Variable for a Site Section

the Preview and Published Site can use the same Content Type but use a different CSS, Javascript, and set of design images and thus take on a completely different look and feel.

Note: The Variable rxs_navbase needs to be a registered Context Variable (even if it is never used). The value associated with rxs_navbase needs to be generic enough to be used whenever a Variable Selector is not defined in a Site. Our example will assume that the Products and Services section will have its own set of design images and CSS. The Javascript file used will be the same one used in all other pages.

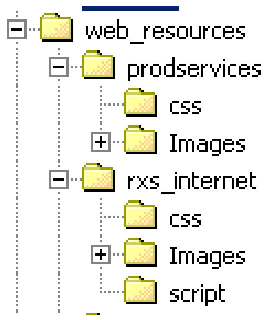
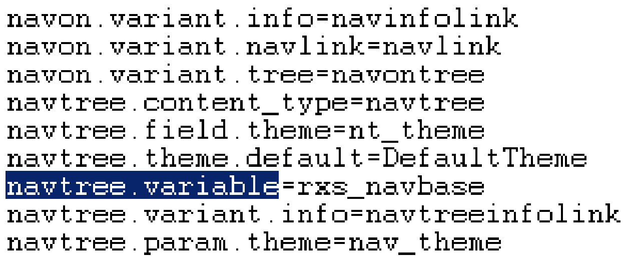


Figure 151: Multiple Locations for Design Elements

Procedure

1 Confirm the existence of a default rxs_navbase Context Variable in the Publishing tab.

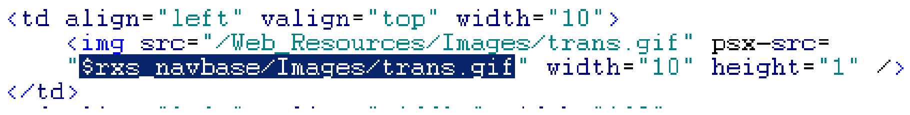
Note: The variable name, rxs_navbase can be changed if necessary. The new variable name must be replace the existing navtree.variable value in the navigation.properties file in the Rhythmyx/rxconfig/server directory. Once changes are made to this file, the Rhythmyx server must be restarted.



```
navon.variant.info=navinfolink
navon.variant.navlink=navlink
navon.variant.tree=navontree
navtree.content_type=navtree
navtree.field.theme=nt_theme
navtree.theme.default=DefaultTheme
navtree.variable=rxs_navbase
navtree.variant.info=navtreeinfolink
navtree.param.theme=nav_theme
```

Figure 152: Variable Selector Value as Defined in navigation.properties

2 Confirm the use of rxs_navbase for all Context Variables except src for <script> tags in the HTML templates for all Variants being used in the Products and Services Site Section.



```
<td align="left" valign="top" width="10">
    <img src="/Web_Resources/Images/trans.gif" psx-src=
    "$rxs_navbase/Images/trans.gif" width="10" height="1" />
</td>
```

Figure 153: Context Variables Used in an HTML Template

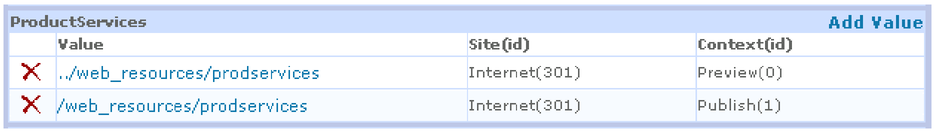
3 Use the Context Variable rxs_js for the src attributes in any <script> tags in the HTML template. Save and close the HTML templates. Replace existing split templates as necessary.

4 Confirm the existence of, or create as necessary, the new design elements (CSS and Images) and build the site structure to conform to the registered folder directories. A page cannot use a CSS file that doesn't exist...

5 Register new Context Variables pointing to the new design elements in the prodservices subdirectories. Be sure to provide a value for all necessary contexts (preview and publish).

Note: This Variable needs to be generic enough to use across CSS and Images. A value like ../web_resources/internet is more flexible than ../web_resources/css. Using the former will allow for the reuse of variables across multiple html tags. Remember, a single value is being used for elements in both a css and Images directory.

For our Products and Services example, lets use the variable ProductServices and register a preview and publish value for our Internet Site



ProductServices

	Value	Site(id)	Context(id)
✗	../web_resources/prodservices	Internet(301)	Preview(0)
✗	/web_resources/prodservices	Internet(301)	Publish(1)

Figure 154: Context Variables for the Products and Services Site Section

6 Navigate to the Content Explorer and locate the Products and Services Site Section in the Internet Site. In this section, select and [Edit] the Section's Navon.

7 Assign the new Variable ProductServices as the value associated with the Navon's Variable Selector field.

8 [Update] and [Close] the Item to save changes.

9 Reset the navigation and preview an Item in the Products and Section. These pages should now take on the look and feel associated with the prodservices design elements. The same templates are simply using different CSS and design images.

Creating Custom Navigation Variants

Navigation Variants use the XML generated by each navigation application. These Variants are responsible for identifying where in the navigation tree the current Item is located, and as a result how to represent the menu for the Item in relationship to nearby Items in the tree.

Note: It is important to note that even though it may be tempting, it is not recommended to add the sys_addAssemblerInfo Exit to the navon or navtree XML resources.

Navigation Schema

Though it is really a combination of several documents, we will always represent the navigation tree as a single XML document. This document consists of a single <navtree> root element and a series of <navon> elements.

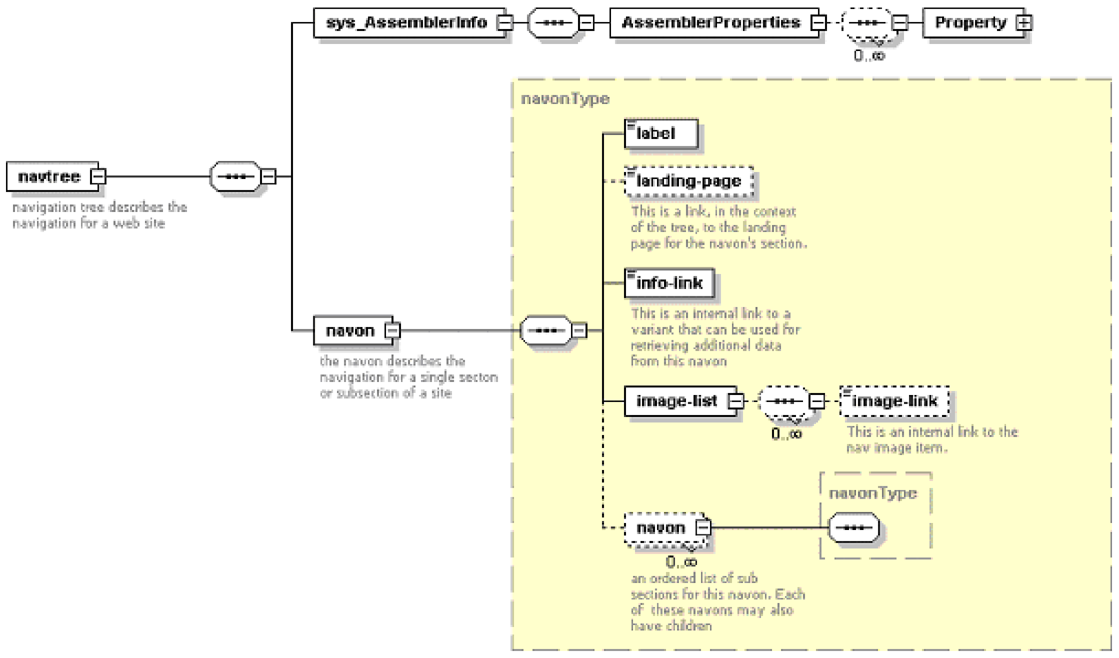

sys_AssemblerInfo
AssemblerProperties
0..∞
Property
navonType
label
landing-page
This is a link, in the context of the tree, to the landing page for the navon's section.
navtree
navigation tree describes the navigation for a web site
info-link
This is an internal link to a variant that can be used for retrieving additional data from this navon
navon
the navon describes the navigation for a single secton or subsection of a site
image-list
0..∞
image-link
This is an internal link to the nav image item.
navonType
navon
0..∞
an ordered list of sub sections for this navon. Each of these navons may also have children

Figure 155: NavTree Schema

The NavTree element has attributes for all of the standard Rhythmyx HTML parameters: sys_contentid, sys_revision, sys_context, sys_authtype, etc.

There is also a set of "Assembler Properties". These properties are in the same format as those added by the sys_casAddAssemblerInfo Exit used on all normal Content Assemblers. These properties are used to implement $Context variables.

The Navon element has 4 major sub-elements.

- The Label is used when rendering navigation Variants. The Label becomes the "clickable" text.

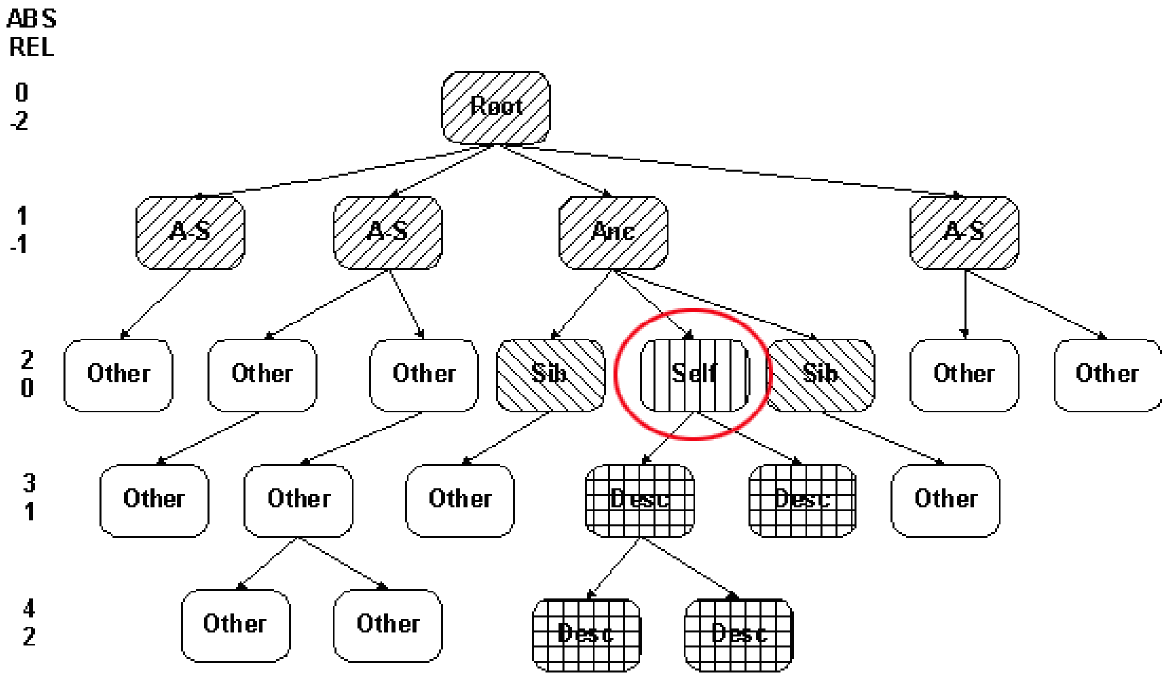
- The Landing Page is a link to the first content item in the Landing Page slot. It is rendered in the context specified by sys_context. The URI of this link is determined by the "href" attribute of the first <a> tag in the attached snippet. In other words, this link points to the Page variant of the landing page that is specified in the snippet that is included in the slot. For External items, the snippet should point directly at the external URL.
- The Info-Link is used for extended variants of the Navon. It is a pointer to a pre-defined XML variant that implementers can add extra fields and information to. This will be used for JavaScript information in the base implementation. Because this variant is intended for use in XSL stylesheets, the link is always an Internal link.
- The Image Link is used to render the menu images. It links to a specific XML variant contained within the Nav Image slot. Again, this link is always rendered as an Internal link for use in XSL stylesheets

Each Navon may also have any number of child Navons, which form the tree.

Every node is categorized by its relationship to the node where the assembly starts. These categorizations are:

- Root
- Ancestor
- Ancestor-Sibling
- Sibling
- Self
- Descendent
- Other

When diagramed, these categorizations take on a meaningful context.



The node where the assembly of the Item starts is the SELF node. All direct parents are ANCESTOR nodes. SIBLING and ANCESTOR-SIBLING nodes are the immediate relatives of the SELF and ANCESTOR nodes. The top node is always the ROOT. DESCENDENT nodes are all nodes underneath the SELF node. Typically, these are the nodes that render in a navigation bar. All OTHER nodes (more distant relations) are not typically used.

There is one special case worth mentioning. In the case where the SELF node is the same as the ROOT node (as in a Site Map variant, for example), here, there are only two categories used: SELF and DESCENDENT.

The Relative and Absolute levels are shown at the left of the diagram. The Absolute level starts at the ROOT (0) and increases as the tree grows downward. The Relative level starts at the SELF node. These levels are intended to be used by rendering style sheets to limit the number of levels displayed, and to control the styles used to render each part of the navigation bars.

Depending on the desired navigation menu, the developer can have each node represented in a unique manner. The node can be expanded, hidden, placed as a descendant of another node, etc. A copy of the navigation schema can be found in the appendix.

Navigation Variant XSL

Without being a lesson on XSLT, the intent of this section is to describe a generic template and its intended function. The style sheet nav_left.xsl is used by the rxs_navon_cas application. This style sheet is used to parse the navigation XML and render the left navigation. This style sheet can be broken down into several small sections. The first of the sections imports the nav_commons.xsl style sheet. In addition, a standard xsl:output statement is defined.

Template - xsl:template match="/"

Result - This first template builds the frame of the HTML document including the <head> and <body>. This template finds a particular node of the XML and begins processing.

```
<html>
   <head>
      <title>nav left snippet</title>
      <link rel="stylesheet" type="text/css"
         href="../web_resources/rxs_nav/css/rxs_nav.css"/>
   </head>
   <body>
      <table width="100%" border="0" cellpadding="0"
         cellspacing="0" class="rxs_navleft">
         <xsl:apply-templates
            select="/navtree/navon[@relation='root' or
            @relation='self']"/>
      </table>
   </body>
</html>
```

Template - xsl:template match="navon[@relation='root']"

Result - If the node being processed in the root, this template does nothing. We will not be displaying the root node in this navigation menu.

```
<xsl:apply-templates select="navon"/>
```

Template - xsl:template match="navon[@relation='ancestor']"

Result - If the node being processed is an ancestor, we will display the landing link indented with a bullet in front of it.

```
      <xsl:variable name="indentclass" select="concat('navlevel',
@absolute-level)"/>
      <tr>
         <td class="{$indentclass}"
            onmouseout="className='{$indentclass}'"
            onmouseover="className='{$indentclass}-on'">
            <div class="open">
               <xsl:if test="@absolute-level='2'">
                  &#xB7;&#160;
               </xsl:if>
               <xsl:call-template name="landinglink"/>
            </div>
         </td>
      </tr>
      <xsl:apply-templates select="navon"/>
```

The landinglink template is contained in the imported file, nav_commons.xsl.

Template - xsl:template match="navon[@relation='sibling' or @relation='ancestor-sibling' or (@relation='descendent' and @relative-level='1') or (@relation='other' and @absolute-level='2') or (@relation='descendent' and @absolute-level='2')]"

Result - This template matches on siblings, ancestor-siblings, descendants with a relative level equal to 1, others with an absolute level of 2, or descendants with an absolute level of 2. The processing on these nodes includes the display of the landinglink with their appropriate class values

```
      <tr>
         <td class="{$indentclass}"
            onmouseout="className='{$indentclass}'"
            onmouseover="className='{$indentclass}-on'">
            <div class="closed">
               <xsl:if test="@absolute-level='2'">
                  &#xB7;&#160;
               </xsl:if>
               <xsl:call-template name="landinglink"/>
            </div>
         </td>
      </tr>
      <xsl:apply-templates select="navon"/>
```

Template - xsl:template match="navon[@relation='self']"

Result - The last template match is if the node being processed is itself (self). If this is true, the landing link is generated and a bullet is place to the left of the indented text.

```
         <tr>
            <td class="{$indentclass}"
               onmouseout="className='{$indentclass}'"
               onmouseover="className='{$indentclass}-on'">
               <div class="self">
                  <xsl:if test="@absolute-level='2'">
```

```
                              &#xB7;&#160;
                           </xsl:if>
                           <xsl:call-template name="landinglink"/>
                        </div>
                     </td>
                  </tr>
               </xsl:if>
               <xsl:apply-templates select="navon"/>
```

All remaining nodes are left unprocessed. The basic concept is that the current node location is identified and processed according to the rules of the template and the matching CSS. Manipulating the output of each template and the CSS allows the developer to customize the navigation as needed.

Once the custom navigation Variant is written, it is added to the rxs_navon_cas assembly application on the navtree resource

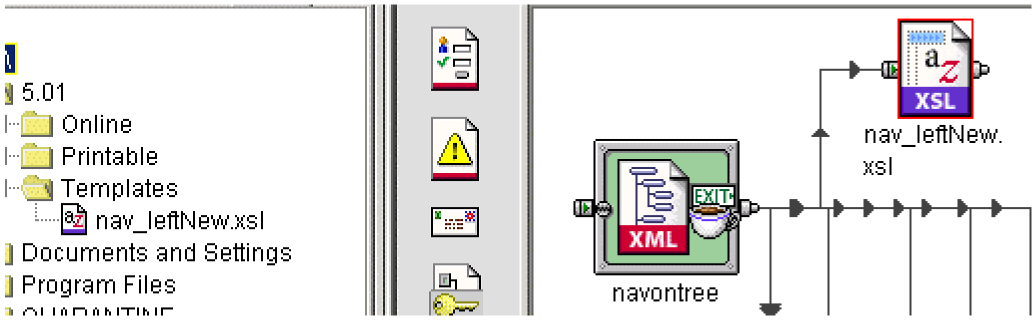
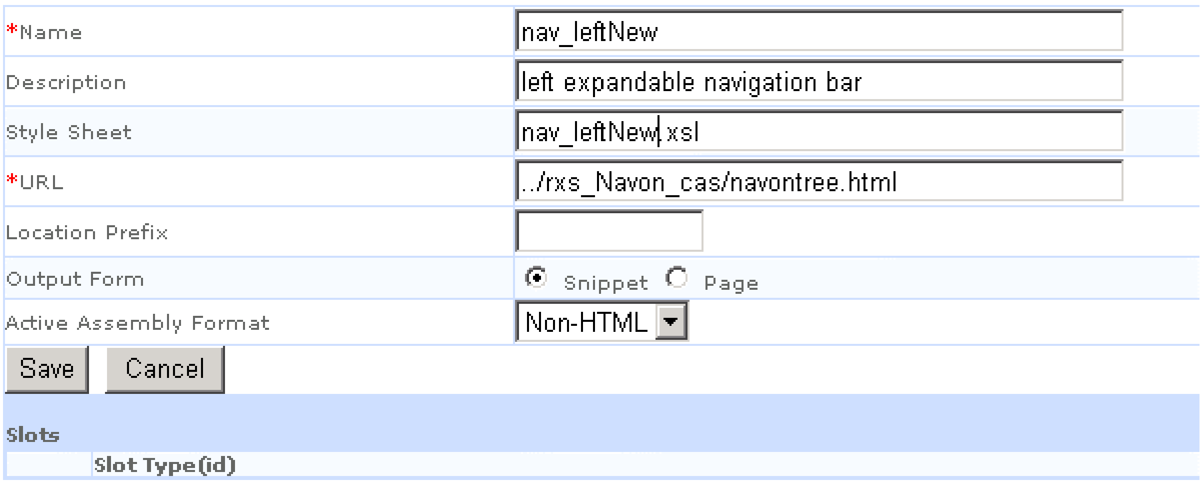


Figure 156: Adding a New Variant to navtree

and the new Variant is registered in the System Administrator like any other Snippet Variant.

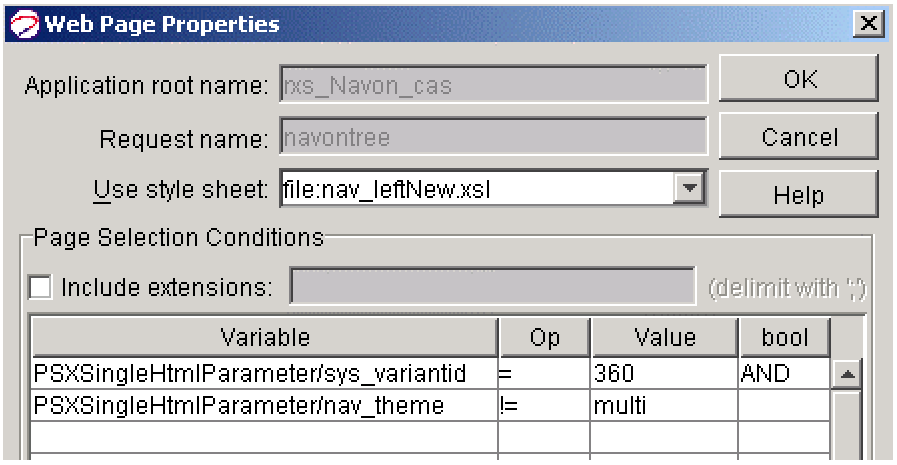


*Name	nav_leftNew
Description	left expandable navigation bar
Style Sheet	nav_leftNew.xsl
*URL	../rxs_Navon_cas/navontree.html
Location Prefix	
Output Form	[✓ Snippet] Page
Active Assembly Format	Non-HTML

Figure 157: Registering nav_LeftNew for Navon

In addition, it is necessary to assign the template its Variant ID and its associated Theme.

If Themes are not being used, simply select when the Theme is NOT equal to multi.



Application root name: rxs_Navon_cas

Request name: navontree

Use style sheet: file:nav_leftNew.xsl

Include extensions:

Variable	Op	Value	bool
PSXSingleHtmlParameter/sys_variantid	=	360	AND
PSXSingleHtmlParameter/nav_theme	!=	multi	

Figure 158: Adding Page Selection Conditions to nav_leftNew.xsl

More information on Themes can be found in the topic Navigation Themes. Save and close the application. Previewing the Variant on any given Item should result in the desired navigation.

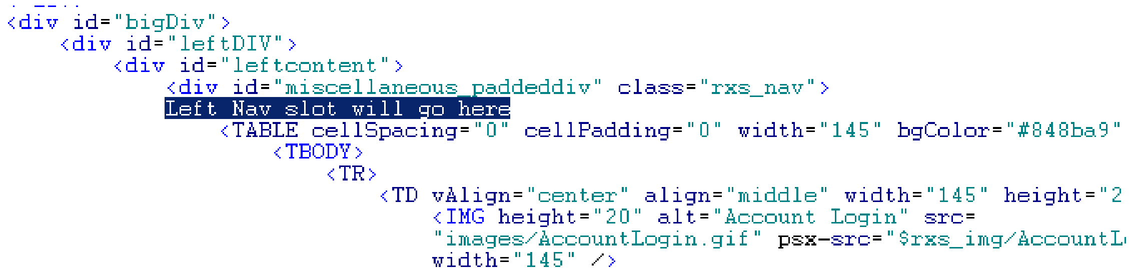
Note: At times, when creating new navigation variants it is tempting to add the addAssemblerInfo Exit to the navontree XML. This is not desirable as the Exit used to build the navon XML generates its own partial assembler info XML. Adding this Exit will cause undesired results.

At this point, this new navigation Snippet should be associated with the global template (or Variant responsible for rendering the navigation). See the topic titled, "Associating a custom navigation variant to a template" for details. Use the sample templates in the Rhythmyx/rxs_Navon_cas directory for assistance when developing new templates.

Adding Navigation Elements to Page Variants

For navigation to render on a Page, the Slot containing the navigation as allowed content must be defined in the Page template.

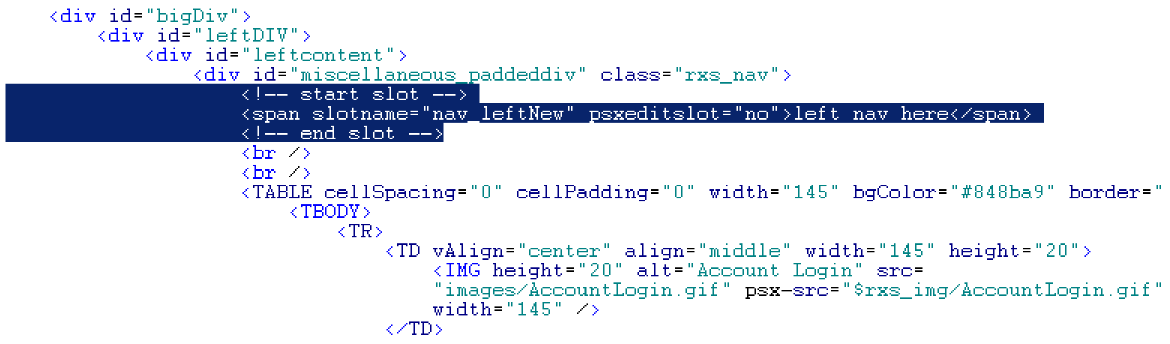
1. Open the Page template where the navigation Snippet will reside.
2. Locate the section of the template where the navigation will be rendered. The navigation Snippet is returned as a table. As such, the location of this Slot should be layed out accordingly.



```
<div id="bigDiv">
    <div id="leftDIV">
        <div id="leftcontent">
            <div id="miscellaneous_paddeddiv" class="rxs_nav">
            Left Nav slot will go here
                <TABLE cellSpacing="0" cellPadding="0" width="145" bgColor="#848ba9"
                    <TBODY>
                        <TR>
                            <TD vAlign="center" align="middle" width="145" height="2
                                <IMG height="20" alt="Account Login" src=
                                "images/AccountLogin.gif" psx-src="$rxs_img/AccountL
                                width="145" />
```

Figure 159: Left Navigation Slot Location in Page Template

3 Add the standard Slot markup to the Page template. Notice the Slot name used is the Slot created specifically to hold the new navigation Snippet. This Slot is set to not allow edits by Content Contributors as it will be automatically populated by the navigation Snippet.



```
<div id="bigDiv">
    <div id="leftDIV">
        <div id="leftcontent">
            <div id="miscellaneous_paddeddiv" class="rxs_nav">
                <!-- start slot -->
                <span slotname="nav_leftNew" psxeditslot="no">left nav here</span>
                <!-- end slot -->
                <br />
                <br />
                <TABLE cellSpacing="0" cellPadding="0" width="145" bgColor="#848ba9" border="
                    <TBODY>
                        <TR>
                            <TD vAlign="center" align="middle" width="145" height="20">
                                <IMG height="20" alt="Account Login" src=
                                "images/AccountLogin.gif" psx-src="$rxs_img/AccountLogin.gif"
                                width="145" />
                            </TD>
```

Figure 160: Left Navigation Slot with markup

Note the addition of the class, "rxs_nav." This class is defined around the navigation and provides standard look and feel. This <div> class should be replaced as necessary.

4 Save and re-split the Page according to the instructions provided in, "Creating a Global Template."

Registering a new Navigation Menu Slot

Once a new Navigation Variant is created and registered it must either replace an existing Snippet as allowed content in an existing Slot or a new Slot must be created, the new Variant assigned to it, and the Slot Template added to the Page Variant using it.

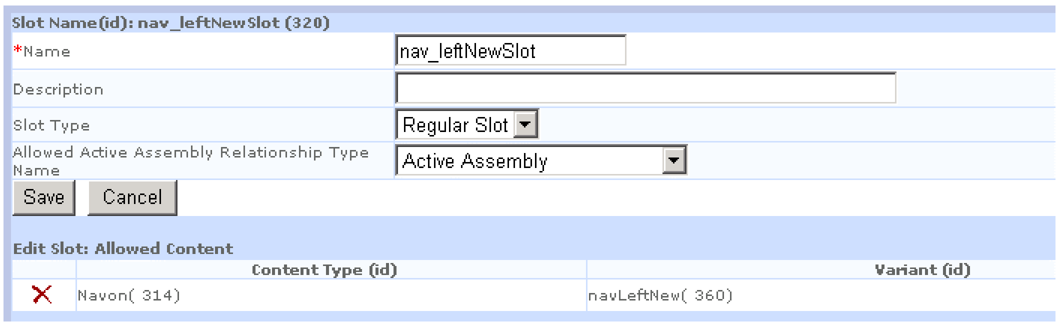
Procedure

1 Log into the System Administrator.

2 Select Slots [By Name] in the left panel.

3 Select [Add New Slot].

4 Register a new Regular Active Assembly Slot.



Slot Name(id): nav_leftNewSlot (320)

*Name	nav_leftNewSlot
Description	
Slot Type	Regular Slot
Allowed Active Assembly Relationship Type Name	Active Assembly

Edit Slot: Allowed Content

	Content Type (id)	Variant (id)
✗	Navon(314)	navLeftNew(360)

Figure 161: Assigning Allowed Content to a Slot

5 Add the new navigation Variant to the Slot as Allowed Content.

6 Save the new Slot registration.

7 On the Rhythmyx Server's file system, navigate to the <Rx Root>/rxconfig/Server directory and open the Navigation.properties file.

8 Locate the navon.slotnames attribute.

9 Add the name of the newly registered Slot to this list. Save and close the file.

Note: The Rhythmyx server must be restarted to pickup the changes to Navigation.properties. The addition of the Slot name to this attribute provides the rxs_NavAutoSlot Exit with a list of Slots to populate with <info-link> elements. This is accomplished by comparing the list of allowed Slots on a Variant to the navon.slotnames attribute. The intersection of these two lists is acted upon by the Exit.

At this point the nav_leftNewSlot can be added to a Page Variant. See the topic, "Adding Navigation Elements to Page Variants."

Using Navigation Themes

In larger, more complex installations, the preferred approach to controlling look and feel within navigation menus is to add a Theme. This allows the developer to create additional XSL style sheets without having to register new Variants or Slots. Using a Theme is less labor intensive than registering an entirely new Variant.

Themes are selected in the NavTree item at the root of the tree. The Theme name is available as an HTML Parameter named “nav_theme”, and also as an attribute of the tree root.

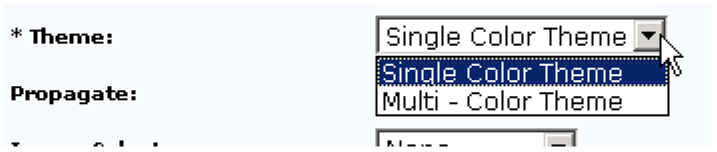
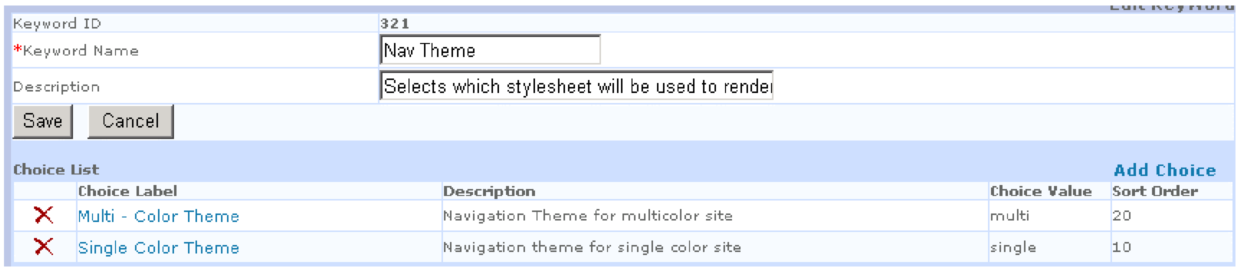


Figure 162: Selecting a Theme

The Themes are defined Key words with corresponding values.


Keyword ID	321
*Keyword Name	Nav Theme
Description	Selects which stylesheet will be used to rendel

	Choice Label	Description	Choice Value	Sort Order
✕	Multi - Color Theme	Navigation Theme for multicolor site	multi	20
✕	Single Color Theme	Navigation theme for single color site	single	10

Figure 163: Theme Keywords

The HTML parameter is then used when selecting style sheets in the workbench. This particular resource has two stylesheets, nav_left and nav_left.

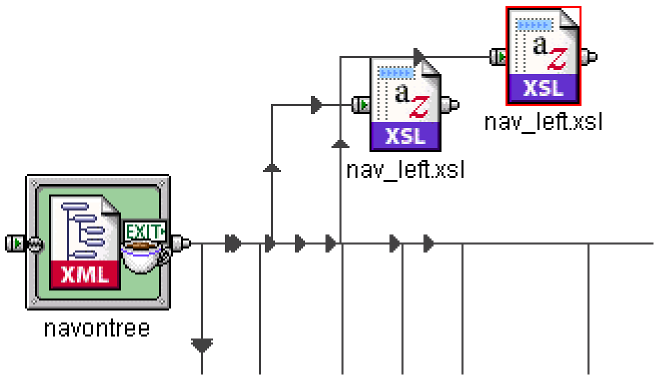
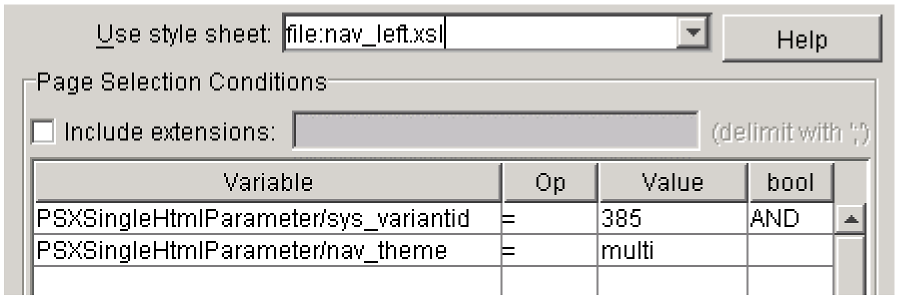


Figure 164: navontree with Two nav_left Style Sheets

The style sheets are differentiated by their page selection conditions.


Use style sheet: file:nav_left.xsl

Help

Page Selection Conditions

☐ Include extensions: (delimit with ';')

Variable	Op	Value	bool
PSXSingleHtmlParameter/sys_variantid	=	385	AND
PSXSingleHtmlParameter/nav_theme	=	multi	

Figure 165: Page Selection Conditions for one Style Sheet

This allows for a single registration of the nav_left Variant, and multiple output styles based on a selection at the root of a Site. For those writing Variant templates, the attribute is useful within an XSL stylesheet, where it can be referenced as: /*/@theme

Implementing Managed Navigation as Server-side Includes

If a Site contains a large number of pages, the publishing process can take considerable time. In these cases, consider implementing Managed Navigation as a set of server-side includes. Using this technique, the navigation bars are published as separate HTML files. In the displayed HTML files, the navigation bars are replaced by `<!--#include -->` statements, such as

```
<!-?#include virtual="leftnav.html" -->
```

When the page is rendered in the browser, this code is replaced by the contents of the included HTML file. Since each HTML file does not have to include the navigation code, the overall size of the HTML files is reduced, making publishing runs faster.

Using server-side includes is most effective when each directory on your Web server contains several HTML files. In Sites that contain only one or two HTML files per directory, implementing server-side includes will not save much time when publishing.

This technique requires the use of a Web server that supports server-side includes (all modern Web servers do). Server-side includes must be enabled on the web server; for details about enabling server-side includes, refer to the documentation for your Web server.

To implement server-side includes:

1 Register new page Variants to assemble the navigation HTML files.

2 Modify the navigation XSL stylesheets to generate the server-side include HTML file and the include tag on the content HTML files.

3 Add the modified stylesheets to the rxs_Navon_cas application.

4 Modify the file rx_Slots.xsl to process the server-side include code correctly.

5 Add new Location Schemes to the Contexts used to publish your Site.

To demonstrate the implementation of server-side includes, we will implement the left navigation bar in FastForward as a server-side include file. For the purposes of this exercise, we will use an Apache Web server. (The Tomcat server installed with Rhythmyx does not include support for server-side includes.)

Registering New Page Variants for the Server-side Include Files

Each navigation bar requires two new Page Variants, one for the Navon Content Type and one for the Navtree Content Type. Note that navigation Variants are typically Snippets because they are assembled into another Page Variant. Since we want these new Variants to generate HTML files, we must register them as Page Variants. We also want to configure these Variants to always be published. Note that you will NOT add these Variants to the navigation.properties file.

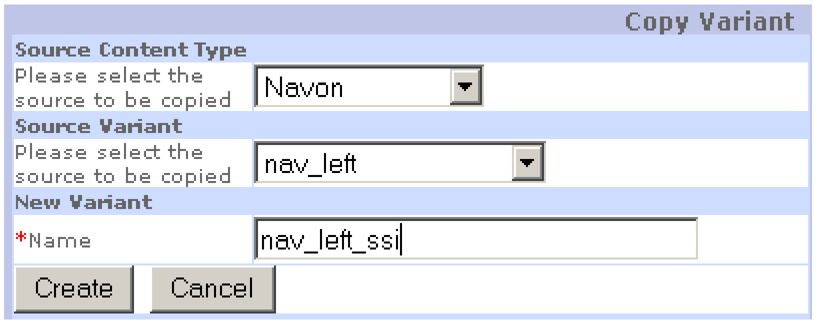
We will create a Variant called “nav_left_ssi”. We will base this Variant on the existing nav_left Variant.

To create this Variant:

1. Log in to Content Explorer.
2. Click the System tab.
3. In the left navigation, under Variants, click the Navon link.

   Rhythmyx displays the Variants Editor page with a list of Variants for the Navon Content Type.
4. Click the Copy Variant link.

   Rhythmyx displays the Copy Variant page.
5. In the Source Content Type field, select *Navon*.
6. In the Source Variant field, select *nav_left*.
7. In the New Variant field, enter *nav_left_ssi*.



Copy Variant

Source Content Type	
Please select the source to be copied	Navon
Source Variant	
Please select the source to be copied	nav_left
New Variant	
*Name	nav_left_ssi

Figure 166: Creating the nav_left_ssi Variant by copying the nav_left Variant

8. Click the [**Create**] button.

   Rhythmyx processes your request and returns to the Variant Editor.

   Other than the name, the new Variant inherits all values from the original Variant. We will need to make some changes.
9. Click on the nav_left_ssi link.

   Rhythmyx displays the Edit Variant page for the nav_left_ssi Variant.
10. Modify the **Description** field by adding the text *for server-side includes* to the end of the existing value.
11. Change the value in the **Stylesheet** field from *nav_left.xsl* to *nav_left_ssi.xsl*.
12. In **Output Form**, select the **Page** radio button.
13. In the **Publish When** field, choose *Always*.
14. Add this Variant to the Enterprise Investment Site

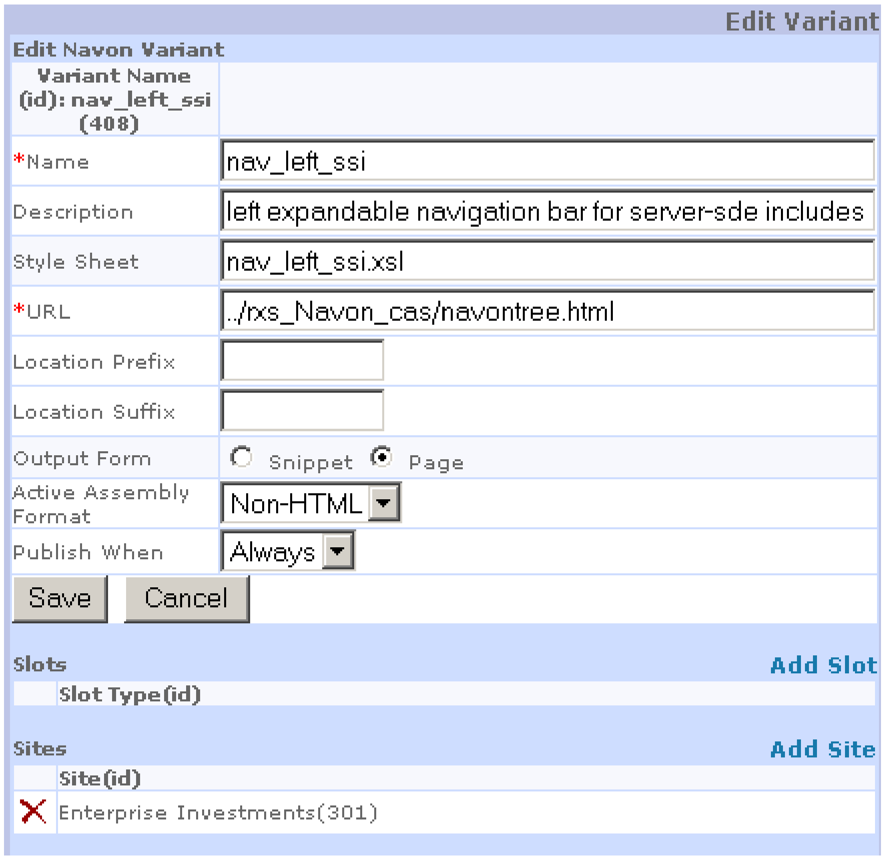
    a) Click the Add Site link.

    Rhythmyx displays the Add Site to Variant Page.

    b) Click the [**Add**] button for the Enterprise Investments Site.

Rhythmyx processes your request and returns you to the Edit Variant page for the nav_left_ssi Variant.

15 Click the [**Save**] button to save your changes.



Variant Name (id): nav_left_ssi (408)

*Name	nav_left_ssi
Description	left expandable navigation bar for server-sde includes
Style Sheet	nav_left_ssi.xsl
*URL	../rxs_Navon_cas/navontree.html
Location Prefix	
Location Suffix	
Output Form	Snippet [✓ Page]
Active Assembly Format	Non-HTML
Publish When	Always

Slots

Slot Type(id)

Sites

Site(id)
Enterprise Investments(301)

Figure 167: Completed nav_left_ssi Variant Registration

Rhythmyx saves your changes and returns you to the Variant Editor. Next you must copy the Variant to the Navtree Content Type:

1 Click on the Navtree link.

Rhythmyx displays the Variants Editor with a list of Variants for the Navtree Content Type.

2 Click the Copy Variant link.

Rhythmyx displays the Copy Variant page.

3 In the **Source Content Type** field, choose *Navon*.

4 In the **Source Variant** field, choose *nav_left_ssi*.

5 In the **Name** field, enter *nav_left_ssi_navtree*.

6 Click the [**Create**] button.

Rhythmyx processes your request and returns to the Variant Editor.

Note the IDs of your new Variants. You will need them later. The following examples assume that the ID of the SSI Variant for the Navon Content Type is 408, while the ID for the Variant for the NavTree Content Type is 409.

Modifying XSL Stylesheets for Server-side Includes

In FastForward, each navigation bar is produced by one stylesheet. The left navigation, for example, is produced by the stylesheet nav_left.xsl.

To implement server-side includes for Managed Navigation, three additional stylesheets are needed. To implement the left navigation using server-side includes, the required stylesheets are:

- nav_left.xsl

  This stylesheet is a modified version of the original stylesheet. It generates the navigation bar when a user previews a page.
- nav_left_ssi.xsl

  This stylesheet generates the left navigation HTML file that will be included on the rendered page by the Web server.
- nav_left_common.xsl

  This stylesheet contains common templates called by other stylesheets (such as nav_left.xsl and nav_left_ssi.xsl).
- nav_left_include.xsl

  This stylesheet generates the `#include` statement in the content HTML pages.

Creating the Common Stylesheet

The common stylesheet contains all of the templates required to the navigation bars that are defined in the original navigation bar stylesheet. The easiest way to create this stylesheet is to create a copy of the original stylesheet with a new name. For example, copy the nav_left.xsl stylseheet and rename it nav_left_common.xsl.

The main template of the common stylesheet should have a name that makes it easy to identify. A simple practice is to add the suffix "_common" to the name of the navigation bar template. For example, since we are implementing the left navigation as a server-side include, and basing our implementation on the nav)left.xsl stylesheet, we would name the main template of our common stylesheet "nav_left_common"

This template should be enclosed in a single container tag, such as a <table> or a <div> tag. For example:

```
<xsl:template name="nav_left_common" >
      <table class="rxs_navleft">
         <xsl:apply-templates select="/navtree/navon [@relation='root'
or @relations='self' ]"  mode="ProcessNavon" />
      </table>
</xsl:template>
```

Add this template as the first template in the stylesheet.

Remove any templates that are not required to generate the navigation bars. You can either comment out these templates or delete them outright. For example, in the nav_left_common.xsl we created by copying nav_left.xsl, we remove the following template:

```
<xsl:template match ="/">
```

(NOTE: Comment out the entire template, not just the initial template statement. For thr sake of space, the entire template will not be reproduced here.)

This should leave the following templates in the nav_left_common.xsl stylesheet (NOTE: only the template statements are included below, not the complete templates):

- `<xsl:template match="/navtree/navon[@relation='root' or @absolute-level='0']" mode="ProcessNavon" priority="11">`
- `<xsl:template match="/navtree/navon[@relation='self']" mode="ProcessNavon" priority="10">· <xsl:template match="navon[@absolute-level='1']">`
- `<xsl:template match="/navtree/navon/navon/navon[@relation='descendant']" mode="ProcessNavon" priority="10">`
- `<xsl:template match="navon[@absolute-level='2']">`
- `<xsl:template match="navon[@relation='self']" priority="5" mode="sectionImage">`

Modifying the Main Navigation Bar Stylesheet

In the main navigation bar stylesheet (nav_left.xsl in our case), remove any templates that are still active in the common stylesheet. In our example, remove all of the templates listed at the end of the topic "Creating the Common Stylesheet" from nav_left.xsl..

In addition, you must include the common stylesheet in the main navigation bar stylesheet. The include tag must preced any output tags in the main stylesheet.

For example, to include the nave_left_common.xsl stylesheet in the nav_left.xsl stylesheet, add the following statement before the <xsl:output> tag in nav_left.xsl:

```
<xsl:include href=″../rxs_Navon_cas/nav_left_common.xsl />
```

Note that this statement assumes we are adding the stylesheets to the Rhythmyx application rxs_Navon_cas.

Finally, you must include a call to the common template in the <body> tag of the man stylesheet. You can either call the common template itself or call one of the component templates directly. For example, the FastForward nav_left.xsl stylesheet includes the call

```
<xsl:apply-templates select="/navtree/navon" mode="ProcessNavon"/>
```

Leaving this code will work, but you can also use an <xsl:call-template> call:

```
<xsl:call-template name=″nav_left_common″ />
```

Creating the SSI Stylehseet

The SSI stylesheet generates the content of the left navigation HTML file. This stylesheet is very simple, consisting of an include statement including the common stylesheet and a call to the main template of the common stylesheet. For example, the nav_left_ssi.xsl file includes the following statements:

```
<xsl:include href="../rxs_Navon_cas/nav_left_common.xsl" />
<xsl:call-template name="nav_left_common" />
```

The complete stylesheet is very short:

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <xsl:import href="../rxs_Navon_cas/nav_commons.xsl"/>
   <xsl:include href="../rxs_Navon_cas/nav_left_common.xsl" />
   <xsl:output method="xml" encoding="UTF-8" omit-xml-
declaration="yes"/>
   <!--<xsl:variable name="rxs_navbase"
select="/*/sys_AssemblerInfo/AssemblerProperties/Property[@name=&apos;rx
s_navbase&apos;]/Value/@current"/>  -->
      <xsl:template match="/">
         <xsl:comment>Nav Left SSI Template</xsl:comment>
         <xsl:call-template name="nav_left_common" />
   </xsl:template>
</xsl:stylesheet>
```

Creating the Include Stylesheet

The include stylesheet generates the `#include` statement for the content HTML files. The `#include` statement is generated by an `<xsl:comment>` such as the following for our implementation:

```
<xsl:comment>#include virtual="leftnav.html"</xsl:comment>
```

Note that with some Web servers, you can substitute “file” for “virtual”. On those Web servers, “file” result in improved performance.

Again, the complete stylesheet is very short:

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:import href="../rxs_Navon_cas/nav_commons.xsl"/>
   <xsl:output method="xml" encoding="UTF-8" omit-xml-
      declaration="yes"/>
   <xsl:variable name="rxs_navbase"
      select="/*/sys_AssemblerInfo/AssemblerProperties/
      Property[@name=&apos;rxs_navbase&apos;]/Value/@current"/>
   <xsl:template match="/">
   <xsl:comment>Nav Left Include Template</xsl:comment>
         <html>
         <head>
            <title>nav left snippet ssi</title>
            <link rel="stylesheet" type="text/css">
               <xsl:if test="$rxs_navbase">
                  <xsl:attribute name="href"><xsl:value-of
```

```
                        select="concat($rxs_navbase,
                        &apos;/css/rxs_nav.css&apos;)"/></xsl:attribute>
                  </xsl:if>
               </link>
               <!-- link rel="stylesheet" type="text/css"
                  href="../web_resources/rxs_nav/css/rxs_nav.css"/  -->
            </head>
            <body>
                   <xsl:comment>#include
                     virtual="leftnav.html"</xsl:comment>
              </body>
           </html>
      </xsl:template>
   </xsl:stylesheet>
```

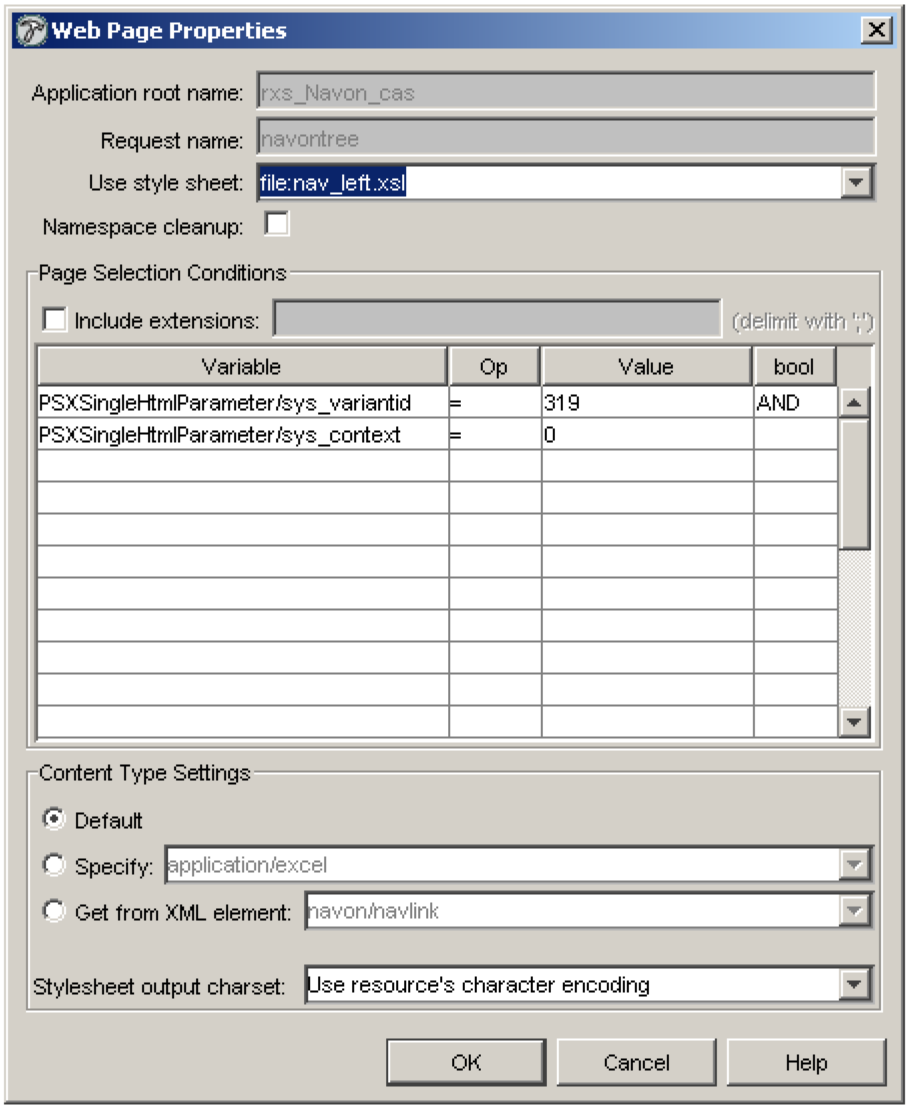
Adding Modified Stylesheets to the Navon Assembly Application

Managed navigation is assembled by the application rxs_Navon_cas. Start the Workbench and open this application. The stylesheets must be attached to the navontree resource in this application. (Note that the nav_left.xsl is already in this application; drop the new version of the stylesheet and choose Update from the popup menu.)

You must also add conditions to each stylesheet in the application to ensure that they are processed correctly. Note that you will need to stop and restart the application after making all changes. The modification described in "***Modifying the Rhythmyx Slots Stylesheet to Process Server-side Includes*** (on page 214)" requires that you stop and restart the Rhythmyx server, so you can also wait until you complete that procedure if you prefer.

Modifying the Conditions on the Existing Navigation Stylesheet

The existing stylesheet should only be called when previewing Page Variants of Content Items. You will need to add a statement to the Web Page Properties that select the stylesheet only when the value of sys_context=0 (the Preview Context). You do not need to change the value of the sys_variantid parameter. The following graphic illustrates an example:



Application root name: rxs_Navon_cas

Request name: navontree

Use style sheet: file:nav_left.xsl

Namespace cleanup: ☐

Page Selection Conditions

☐ Include extensions: (delimit with ';')

Variable	Op	Value	bool
PSXSingleHtmlParameter/sys_variantid	=	319	AND
PSXSingleHtmlParameter/sys_context	=	0	

Content Type Settings

[✓ Default]

Specify: application/excel

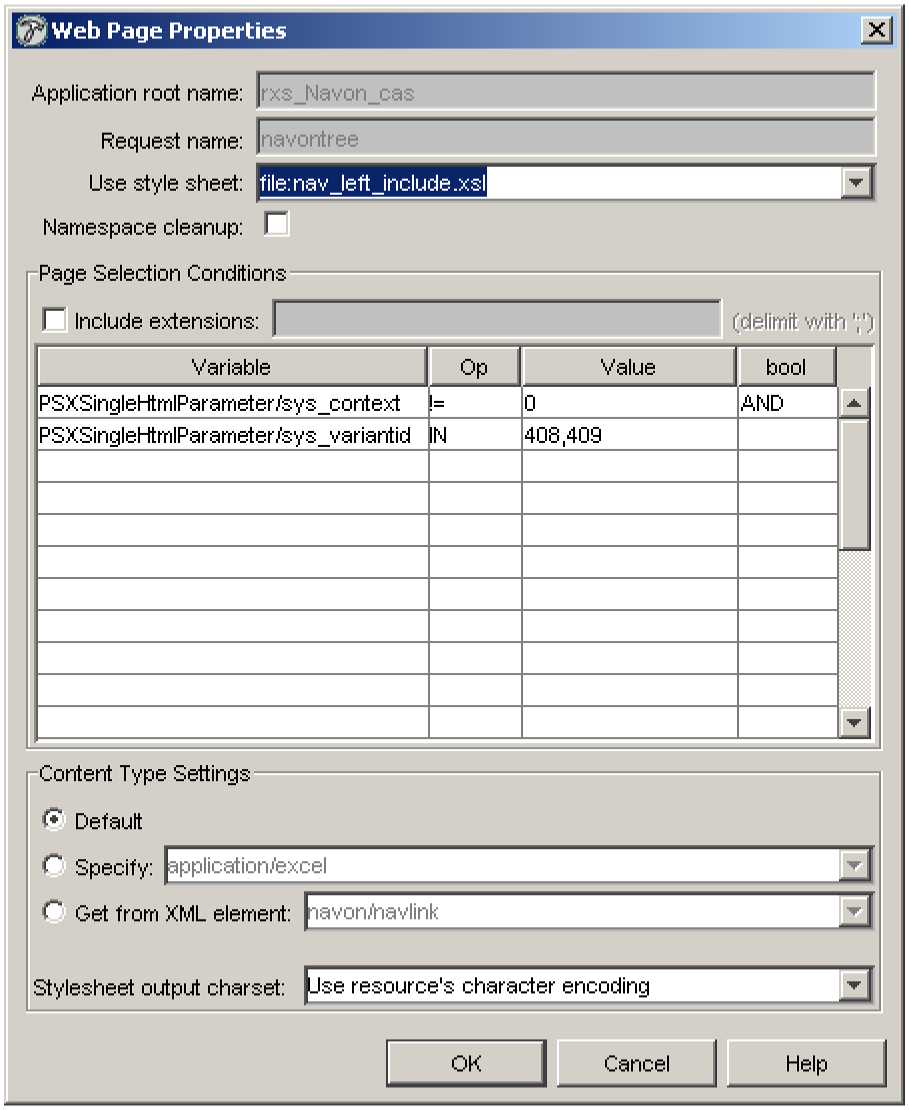
Get from XML element: navon/navlink

Stylesheet output charset: Use resource's character encoding

Figure 168: Web Page Properties defining the conditions to call the nav_left stylesheet

Defining Conditions for the Include Stylesheet

The Include stylesheet is selected when the navigation is assembled into an HTML page during a publishing run. The conditions include the Variant ID (408 or 409) and sys_context does not equal 0 (sys_context !=0). The following graphic illustrates an example:



Application root name: rxs_Navon_cas

Request name: navontree

Use style sheet: file:nav_left_include.xsl

Namespace cleanup: ☐

Page Selection Conditions

☐ Include extensions: (delimit with ';')

Variable	Op	Value	bool
PSXSingleHtmlParameter/sys_context	!=	0	AND
PSXSingleHtmlParameter/sys_variantid	IN	408,409	

Content Type Settings

[✓ Default]

Specify: application/excel

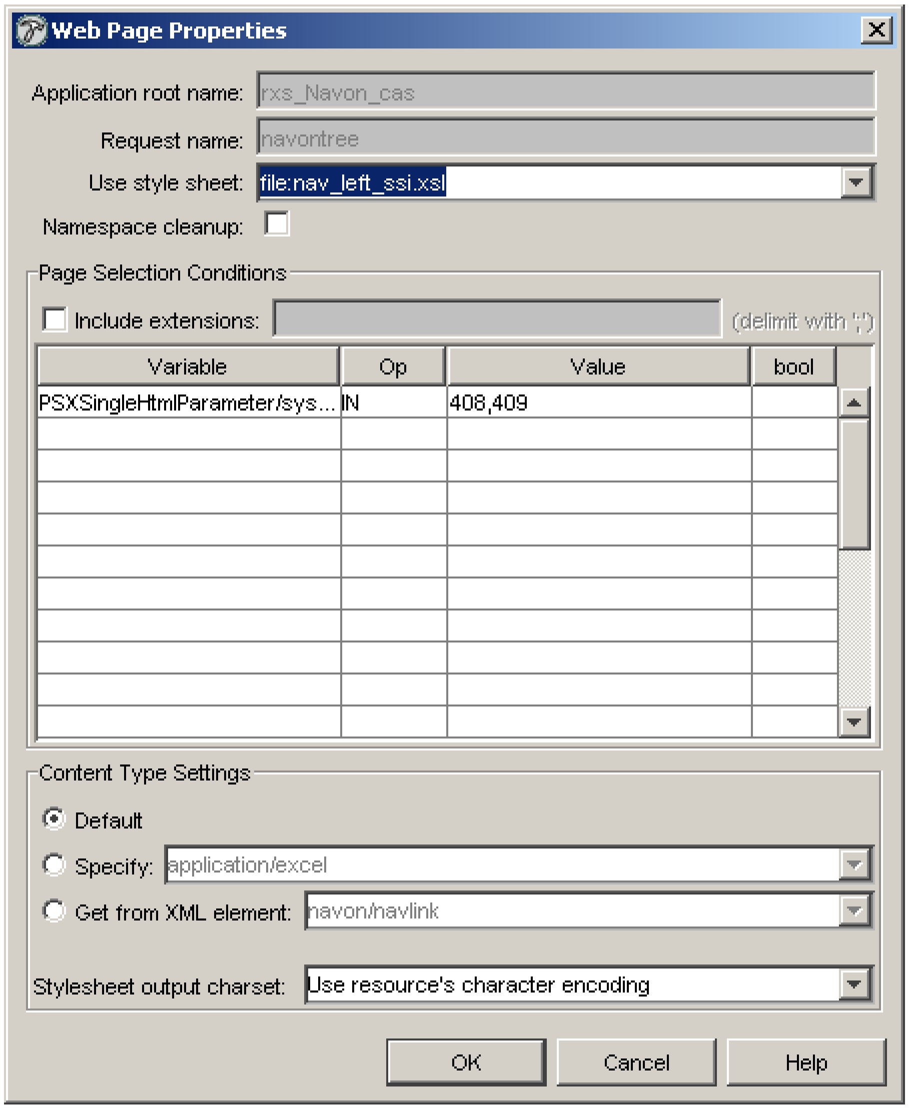
Get from XML element: navon/navlink

Stylesheet output charset: Use resource's character encoding

Figure 169: Web Page Properties defining the conditions to call the nav_left_include stylesheet

Defning Conditions for the Server-side Include Stylesheet

The SSI stylesheet generates the SSI HTML files during a publishing run. The conditions on this stylesheet call it when one of the new Page Variants defined earlier is assembled. Use the IN operator and enter a list of possible Variant IDs into the Value (for example, sys_variantid IN 408,409. The following graphic shows an example:



Application root name: rxs_Navon_cas

Request name: navontree

Use style sheet: file:nav_left_ssi.xsl

Namespace cleanup: ☐

Page Selection Conditions

☐ Include extensions: (delimit with ';')

Variable	Op	Value	bool
PSXSingleHtmlParameter/sys...	IN	408,409	

Content Type Settings

● Default

○ Specify: application/excel

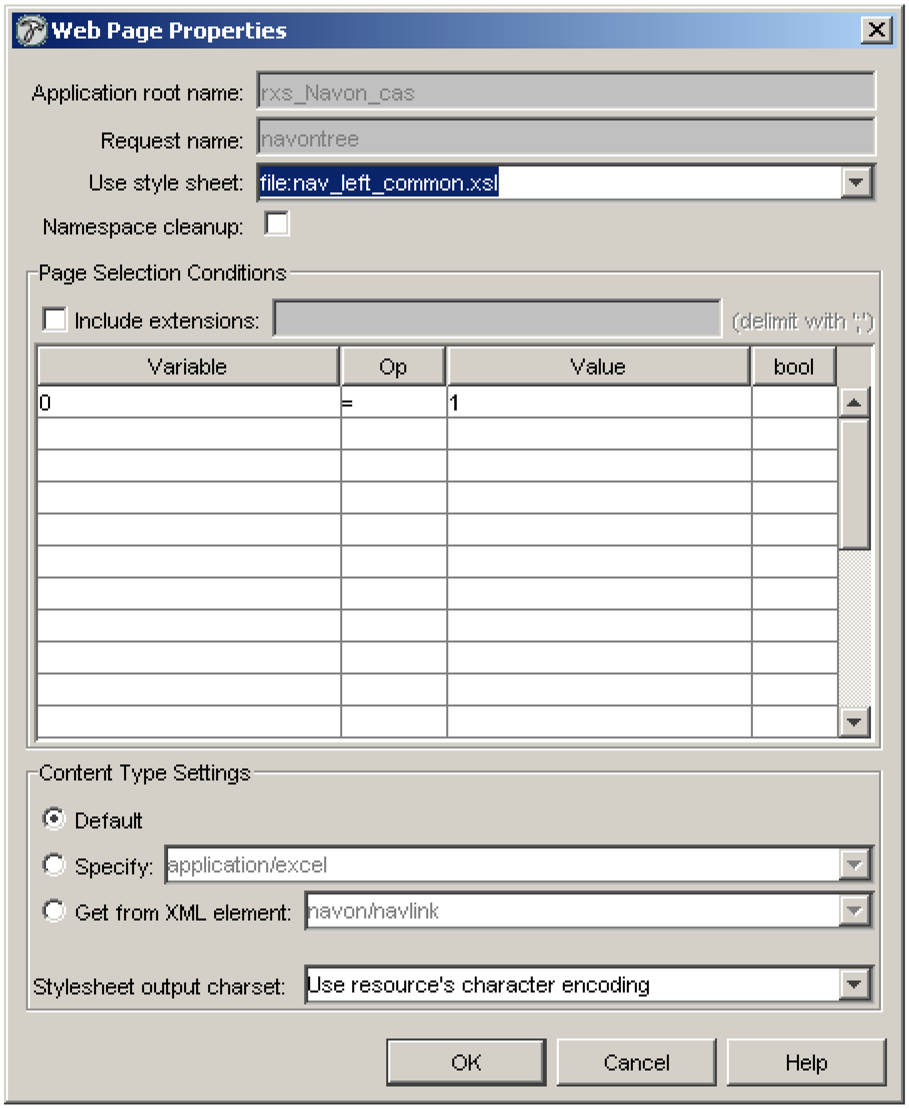
○ Get from XML element: navon/navlink

Stylesheet output charset: Use resource's character encoding

Figure 170: Web Page Properties showing conditions to call the server-side include stylesheet

Defining Conditions for the Common Stylesheet

You must add the common stylesheet to the rxs_Navon_cas application to ensure that it can be called correctly and to ensure that it will be included when Multi-Server Manager packages the application for deployment. The stylesheet will never be called directly, however, so we need to add a condition to ensure that it can never be called, such as 0=1.



Application root name: rxs_Navon_cas
Request name: navontree
Use style sheet: file:nav_left_common.xsl
Namespace cleanup: ☐

Page Selection Conditions

☐ Include extensions: (delimit with ';')

Variable	Op	Value	bool
0	=	1	

Content Type Settings

[✓ Default]
Specify: application/excel
Get from XML element: navon/navlink

Stylesheet output charset: Use resource's character encoding

Figure 171: Web Page Properties defining conditions to call the common stylesheet. Note that the condition can never occur, ensuring that the stylesheet can never be called directly.

Modifying the Rhythmyx Slots Stylesheet to Process Server-side Includes

To ensure that the `#include` statement is processed correctly and included in the output HTML pages, you must make a minor change to the stylesheet <Rhythmyxroot>/rx_resources/stylsheets/assemblers/rx_Slots.xsl. Find the following template:

```
<xsl:template match="linkurl" mode="rxs_nav_Inner">
      <xsl:copy-of select="document(Value/@current)/*/body/*"/>
</xsl:template>
```

Change this template by replacing the last "`*`" with "`node()`":

```
<xsl:template match="linkurl" mode="rxs_nav_Inner">
      <xsl:copy-of select="document(Value/@current)/*/body/node()"/>
</xsl:template>
```

(This change is required because in XSL, "`*`" means "all elements" while "`node()`" means "all nodes". The statement adding the `#include` is an XSL comment, not an element, so using "`*`" will not copy the `#include` statement to the output file. Changing this code to copy all nodes by changing "`*`" to "`node()`" corrects this problem.)

You must shut down and restart your Rhythmyx server before this change will take effect.

Adding New Location Schemes to Publish Contexts

Each of the new Variants will require a Location Scheme for the Publishing Context (sys_context=1). The Location Scheme uses a Rhythmyx resource to generate the output location for the HTML file. The resource is specified in the Location Scheme Parameter.

Creating the Server-side Include Output Location Resource

The SSI output location resource will be called by the Location Schemes to generate the location where the navigation HTML files will be published. This resource uses the Site Folder Assembly infrastructure to determine the correct location. To create the resource:

1 Start the Rhythmyx Workbench.

2 Create a new application.

3 Choose the Files tab, and navigate to the DTD directory. Drop the location.dtd into the new application. On the popup menu, choose *Query*.

4 Double-click on the new resource to open the Resource Editor.

5 Click on the Data tab for your backend and add the table RXDUAL as the datatank for the query. (A resource is not valid without a table in the datatank. The RXDUAL table is a dummy table used to facilitate the creation of resources when the resource does not require data from the backend.)

6 Double-click on the mapper icon to open the mapper.

7 Add the sys_casGetSiteBaseUrl UDF to the mapper.

a) On the Backend side of the mapper, select *User Defined Functions*.

b) Expand the Global node, then expand the contentassembler node.

c) Drag the sys_casGetBaseUrl to the first line of the mapper.

Rhythmyx displays the Function Properties dialog.

d) Double-click in the Value column of the SiteId parameter, then click the browse button.

Rhythmyx displays the Value Selector.

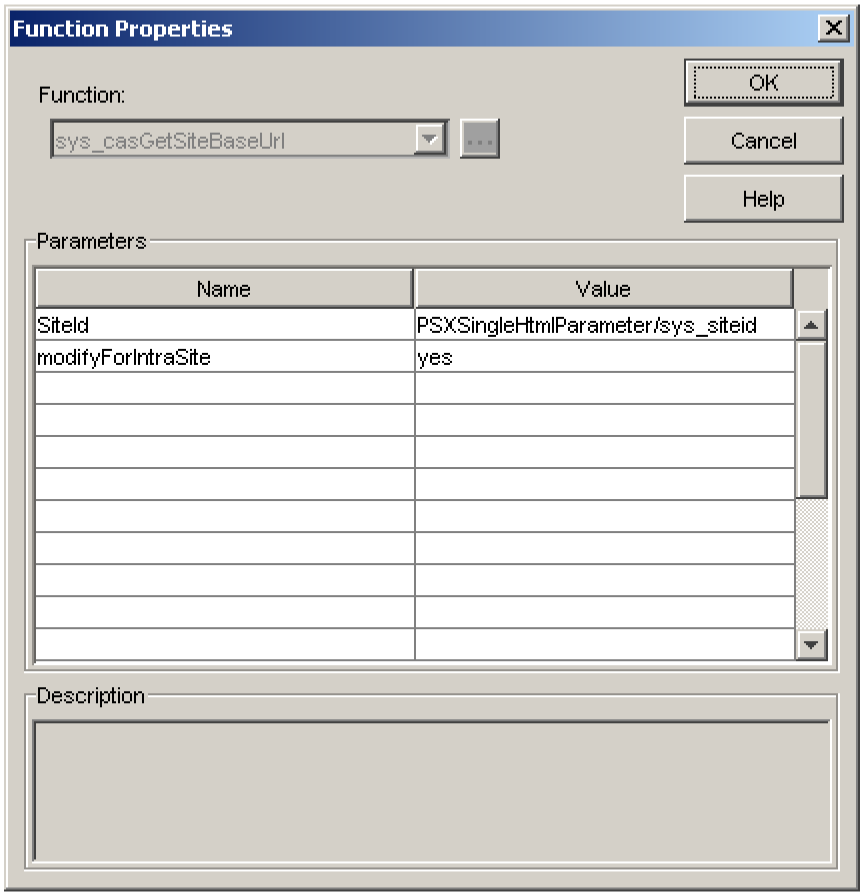
e) In the **Type** field, choose *Single HTML Parameter*.

f) In the **Choices** field, select *sys_siteid*. Rhythmyx adds this value to the Value field.

g) Click the [**OK**] button to save your choice.

Repeat steps e through i for the modifyForIntraSite parameter. The Value should be the Literal Value *yes*.

h) Click the [**OK**] button to save the properties for this function.



Function: sys_casGetSiteBaseUrl

Name	Value
SiteId	PSXSingleHtmlParameter/sys_siteid
modifyForIntraSite	yes

Figure 172: Parameters of the sys_casGetSiteBaseURL UDF to define the output location

8 From the DTD side of the mapper, drag the *baseurl* attribute to the XML side of the row where you added the sys_casGetBaseUrl UDF.

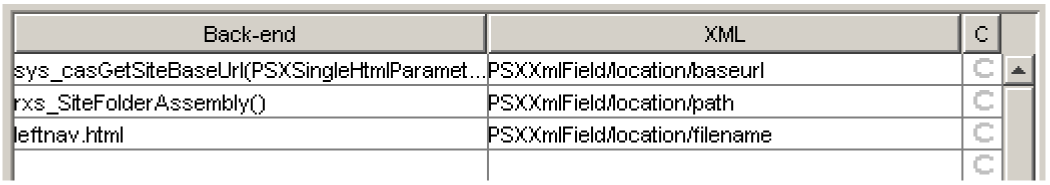
9 Under User Defined Functions, expand the sitefolderpublishing node. Drag the rxs_SiteFolderAssembly UDF to the next blank row. This UDF does not require any parameters.

10 From the DTD side of the mapper, drag the *path* attribute to the XML side of the row where you added the rxs_SiteFolderAssembly UDF.

11 Double-click in the next blank row and enter *leftnav.html*.

12 Drag the *filename* element to the XML side of the row where you entered *leftnav.html*.

13 Click the [**OK**] button to save the mappings.



Back-end	XML	C
sys_casGetSiteBaseUrl(PSXSingleHtmlParamet...	PSXXmlField/location/baseurl	C
rxs_SiteFolderAssembly()	PSXXmlField/location/path	C
leftnav.html	PSXXmlField/location/filename	C
		C

14 Close the Resource Editor.

15 Rename the resource.

a) Right-click on the resource and from the popup menu choose *Request Properties*.

Rhythmyx displays the Request Properties dialog.

b) Change the **Request Name** to *Nav_SSILocation*.

c) Click the [**OK**] button to save your edits.

16 Save the application. Name the application <companyname>_NavSSI, where <companyname> is the name of your company or other prefix you use to designate Rhythmyx applications you develop in your implementation.

17 Start the application.

If you implement multiple navigation bars as server-side includes, you will need to map the filename element of the location DTD to multiple file names with conditional processing for each row to apply the correct file name based on the Variant being processed. The condition will be

```
sys_variantid=<Variant ID of the  SSI Variant>
```

You might also use the IN operator to choose the row for multiple Variants.

For example, if we wanted to add a condition to add the name leftnav.html only for the Variants we created earlier (408 and 409), the conditional statement would be:

```
sys_variantid IN 408,409
```

Creating the Location Scheme

The following procedure illustrates how to create the Location Scheme for the nav_left_ssi Variant of the Navon Content Type. Use the same procedure, with appropriate changes, to create the Location Scheme for the Navtree Variant.

To create the Location Scheme:

1 Start Content Explorer and click the Publishing tab.

2 In the left navigation, under Contexts, click the By Name link.

Rhythmyx displays the Contexts Editor.

3 Click the Publish link.

Rhythmyx displays the Edit Context page for the Publish Context.

4 Click the New Location Scheme link.

Rhythmyx displays the Edit Location Scheme page.

5 In the **Name** field, enter *Nav Left SSI*. Enter the same value in the **Description** field.

6 In the **Generator** drop list, choose *sys_casGenericAssemblyLocation*.

7 In the **Content Type** drop list, choose *Navon*.

8 In the **Variant Type** drop list, choose *nav_left_ssi*.

9 Click the [**Save**] button.

Rhythmyx processes your request and returns you to the Edit Context Page.

10 Now you must add a parameter to retrieve data from the SSI output location resource.

a) Click on the Nav Left SSI link.

Rhythmyx displays the Edit Location Scheme page for the Nav Left SSI Location Scheme.

b) Click on the New Location Scheme Parameter link.

Rhythmyx displays the Edit Location Scheme Parameter page.

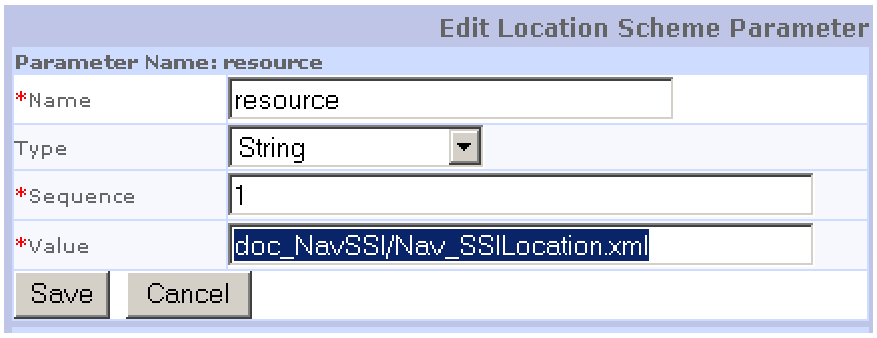
c) In the **Name** field, enter *resource*.

d) In the **Type** drop list, choose *String*.

e) In the **Sequence** field, enter *1*.

f) In the Value field, enter *<companyname>_NavSSI/Nav_SSILocation.xml*, where <companyname> is the name of your company or other prefix you use to designate Rhythmyx applications you develop in your implementation. This value designates the application and resource you defined to generate the publishing location for the navigation HTML files.

g) Click the [**Save**] button to save the parameter.



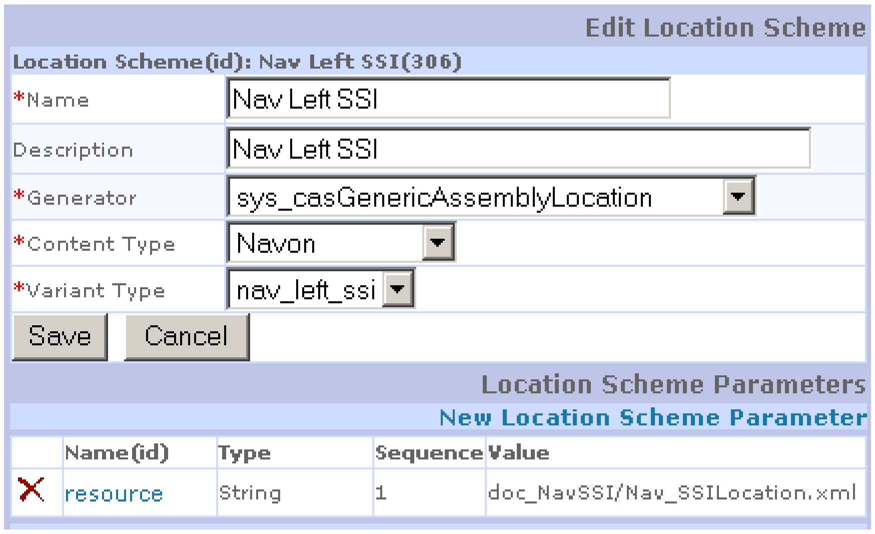
Edit Location Scheme Parameter

Parameter Name: resource

*Name	resource
Type	String
*Sequence	1
*Value	doc_NavSSI/Nav_SSILocation.xml

Figure 173: Location Scheme Parameter specifying the resoruce for server-side includes

On the Edit Location Scheme page, click the [**Save**] button to save the updates to the Location Scheme.



Edit Location Scheme

Location Scheme(id): Nav Left SSI(306)

*Name	Nav Left SSI
Description	Nav Left SSI
*Generator	sys_casGenericAssemblyLocation
*Content Type	Navon
*Variant Type	nav_left_ssi

Location Scheme Parameters

New Location Scheme Parameter

	Name(id)	Type	Sequence	Value
✗	resource	String	1	doc_NavSSI/Nav_SSILocation.xml

Figure 174: Nav Left SSI Location Sheme

Validating the Manged Navigation Using Server-side Includes Implementation

To verify that you have correctly implemented server-side includes for Managed Navigation, perform the following tests:

1. Preview the original Variant. You should see the same preview as before implementing server-side includes. In the URL of the preview window, change the value of the sys_context attribute to 1 (the publishing context), then refresh the preview. Rhythmyx should return a blank window. View the source. You should see the code similar to the following for the body:

   ```
   <body><!--#include virtual="leftnav.html" --></body>
   ```

   (The code will include “file” instead of “virtual” if you defined your includes statement to use that option.)

   You should see an instance of the include statement for each navigation bar you have converted to use server-side includes.

2. Preview the SSI Variant. This preview should contain the rendered navigation bar. You should see the same output if you change the context to 1.

3. Preview a landing page. You should see a normal preview of the page. In the URL of the preview window, change the value of the sys_context attribute to 1 (the publishing context), then refresh the preview. The preview should be rendered without the navigation bars you have converted to use server-side includes. View the source for the page, and you should an include statement (similar to the example in Test 1) for each navigation bar you have converted to use server-side includes.

4. On the Publishing tab of Content Explorer, preview the Full Publish Edition Site Folder Content List for the Site. View the Source for the preview. Search for the Variant IDs of your Managed Navigation Variants. You should find at least one instance of each Variant ID. The delivery location for the output page for each Variant should include the complete path, followed by the filename for the published page (in our example, leftnav.html).
5. Publish the Full Edition. Each output director should contain an HTML file for each navigation bar you have converted to use server-side includes.
6. Start a new browser session and access the site from the Web server. Browse the site. All navigation bars you have converted to use server-side includes should be rendered correctly on each page.

If any XSL code is incorrect, the previews will return errors attempting to preview the pages.

If the navigation is not included, or other unexpected results occur, check the configuration of this implementation. In particular, confirm that you have specified the correct Variant IDs for the XSL files in the Web Page Properties dialog for each file.

CHAPTER 8

About Workflows in Rhythmyx

A workflow is a business process that defines a sequence of events. Assign a workflow to a Rhythmyx application to define and control the sequence of events that occur in that application.

In Rhythmyx, a workflow consists of ***States*** (on page 243) (steps in the workflow processing of a content item) and ***Roles*** (see "Workflow Roles" on page 229) (groups of individuals who can act on content). Assign Roles to determine which users can take action on content in each State. Define Transitions to move content from one State to another.

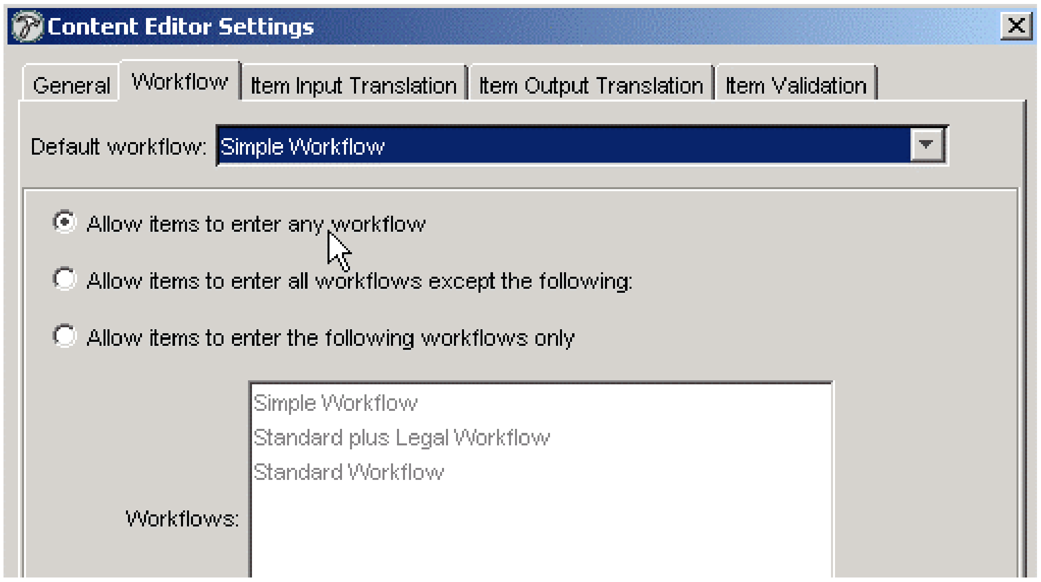
When you create content in a Rhythmyx application that is attached to a workflow, the content is automatically managed by the workflow. The content starts in an *Initial State*, which is the first State the content enters in the workflow. Most commonly, the Initial State is some form of draft state in which the creator actively edits the content. When the draft is complete, the creator initiates a *Transition* to a State, typically some form of approval State, in which a member of another Role reviews the content. Further Transitions may involve additional drafts and approvals. The exact pattern is defined by the combination of States, Roles, and Transitions in the workflow. Eventually, the content reaches a State where it is Publishable, meaning it is eligible to be processed by the Rhythmyx Publisher application, which delivers the final output of the content.

Copying a Workflow

As described, FastForward ships with two Workflows, Simple and Standard. During the creation of additional Sites, it is sometimes necessary to create modified Workflows (usually a new copy of each the Simple and Standard). Rhythmyx provides a mechanism for easily duplicating an entire Workflow when needed.

1 Log into the Content Explorer.
2 Navigate to the Workflow tab.
3 Select the [Copy Workflow] link.
4 Select the Workflow to be copied in the drop down menu.
5 Give the soon to be created Workflow a unique name.
6 Press the [Copy] button.
7 Modify the new Workflow as necessary to meet the new business needs.

Once this new Workflow is complete, it is necessary to associate it with the Community it is being used with (by default the Workflow is assigned as an allowed Workflow in the Community it was copied in). If a Community has multiple allowed Workflow, it is necessary to associate Content Types with a specific default Workflow.



Default workflow: Simple Workflow

(●) Allow items to enter any workflow
( ) Allow items to enter all workflows except the following:
( ) Allow items to enter the following workflows only

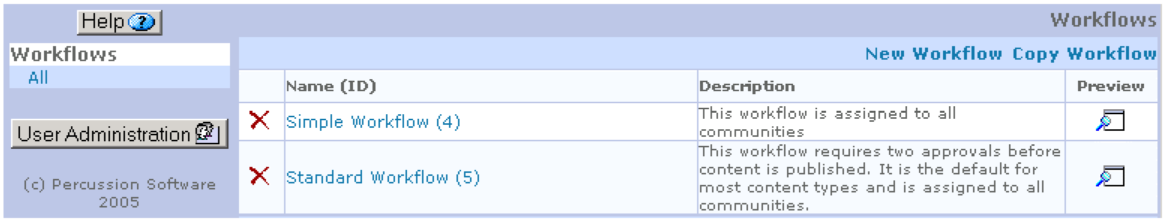
Workflows:
Simple Workflow
Standard plus Legal Workflow
Standard Workflow

Figure 175: Content Editor Workflow Tab

Often, a single community has associated with it a single Workflow. As such, it is not necessary to assign a given Content Type a specific Workflow. If set to use any Workflow, when the item is created, Rhythmyx selects the available Workflow for a Community (if the default is not available) and assigns the Items to that Workflow.

Maintaining Workflows

When you start Rhythmyx Workflow Editor, Rhythmyx displays the Workflow Editor page.



	Name (ID)	Description	Preview
✗	Simple Workflow (4)	This workflow is assigned to all communities	
✗	Standard Workflow (5)	This workflow requires two approvals before content is published. It is the default for most content types and is assigned to all communities.	

Figure 176: Workflow Editor

Each workflow item listing includes two maintenance buttons. Clicking on a button prompts Rhythmyx to perform a specific function.

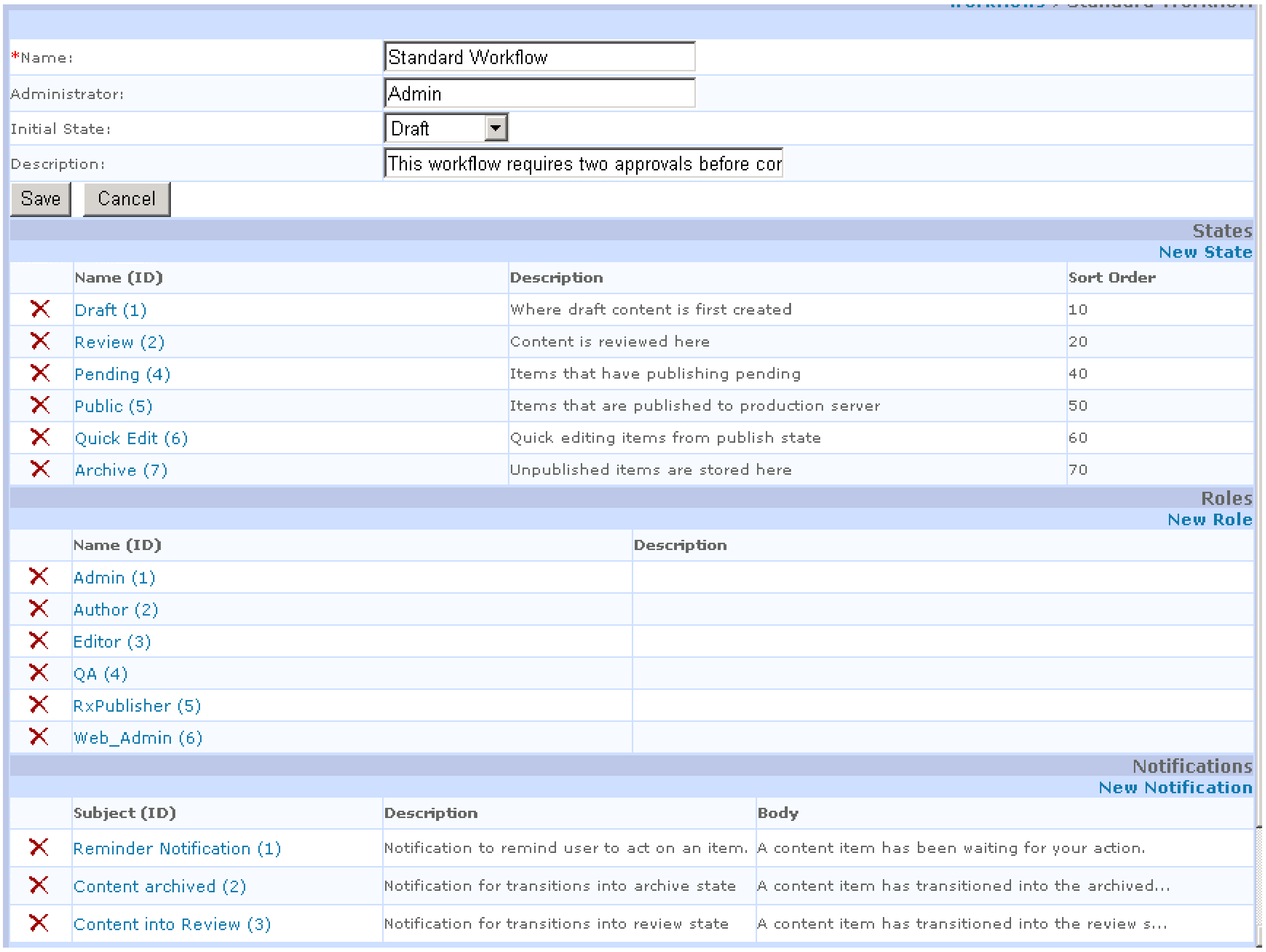
Click this button…	To…
✗	delete the workflow item.
🔍	preview the workflow item.

To edit a Workflow, click on its name.

Edit Workflow Page

Use the Edit Workflow page to maintain workflows.

A new workflow does not include ***States*** (on page 243), ***Roles*** (see "Workflow Roles" on page 229), or ***Notifications*** (on page 257). Therefore, when Rhythmyx displays this page as the New Workflow page, the lists of States, Roles and Notifications are not included.



*Name:	Standard Workflow
Administrator:	Admin
Initial State:	Draft
Description:	This workflow requires two approvals before cor

	Name (ID)	Description	Sort Order
✗	Draft (1)	Where draft content is first created	10
✗	Review (2)	Content is reviewed here	20
✗	Pending (4)	Items that have publishing pending	40
✗	Public (5)	Items that are published to production server	50
✗	Quick Edit (6)	Quick editing items from publish state	60
✗	Archive (7)	Unpublished items are stored here	70

	Name (ID)	Description
✗	Admin (1)	
✗	Author (2)	
✗	Editor (3)	
✗	QA (4)	
✗	RxPublisher (5)	
✗	Web_Admin (6)	

	Subject (ID)	Description	Body
✗	Reminder Notification (1)	Notification to remind user to act on an item.	A content item has been waiting for your action.
✗	Content archived (2)	Notification for transitions into archive state	A content item has transitioned into the archived...
✗	Content into Review (3)	Notification for transitions into review state	A content item has transitioned into the review s...

Figure 177: Edit Workflow Page

You can access the Edit Workflow page in the following ways:

- On the Workflow Welcome page, click the Name of the workflow you want to edit;
- On the Workflow Welcome page, click Workflow in the New box.

Edit Workflow Fields

Name Mandatory. The name of the workflow.

Administrator Optional. The ***Workflow Role*** (see "Workflow Roles" on page 229) defined as the Administrator of this workflow. For more about the Workflow Administrator, see About Workflow Administrator.

Initial State Read only. The numeric identifier of the first ***State*** (see "States" on page 243) that an item enters in the workflow. Defaults to *1*.

Description Optional. Free-form description of the workflow.

Editing a Workflow

A workflow consists of ***States*** (on page 243) and ***Roles*** (see "Workflow Roles" on page 229). You can add States and Roles to a workflow or delete them from the workflow.

To edit a workflow:

1. Access the Workflow Editor.
2. Click the Name of the Workflow you want to edit.
3. Rhythmyx displays the Edit Workflow page.

From the Edit Workflow page you can

- ***add a State*** (see "Creating a New State" on page 244);
- ***edit a State*** (see "Editing a State" on page 245);
- ***delete a State*** (see "Deleting a State" on page 245);
- ***add a Role*** (see "Creating a New Role in Rhythmyx Workflow" on page 230);
- ***edit a Role;*** (see "Editing a Role in Workflow" on page 230)
- ***delete a Role*** (see "Deleting a Role in Workflow" on page 230)
- ***add a Notification*** (see "Creating a Notification" on page 258);
- ***edit a Notification*** (see "Editing a Notification" on page 258);
- ***delete a Notification*** (see "Deleting a Notification" on page 258).

Deleting a Workflow

To delete a workflow:

1. Access the Workflow Editor.
2. Click the Delete button ✕ in the row of the workflow you want to delete.

   Rhythmyx displays a confirmation page.
3. To confirm the delete action, click [**OK**]. To abort the delete action, click [**Cancel**].

Viewing a Workflow

Rhythmyx Workflow Editor can display a graphic representation of each workflow. To view the workflow graphically, click the Preview button in the row of the workflow you want to view.

Vertical lines represent ***States*** (on page 243). The name of the State at the bottom of each line is a hyperlink to the Edit State page for that State. To add a new state to the workflow, click the [**Add State**] button.

Horizontal arrows represent Transitions. The name of each Transition is a hyperlink to the Edit Transition page for that Transition. To add a new Transition to a State, click the New Transition button under that State. To add a new Aging Transition to a State, click the new Aging Transition button under that State.

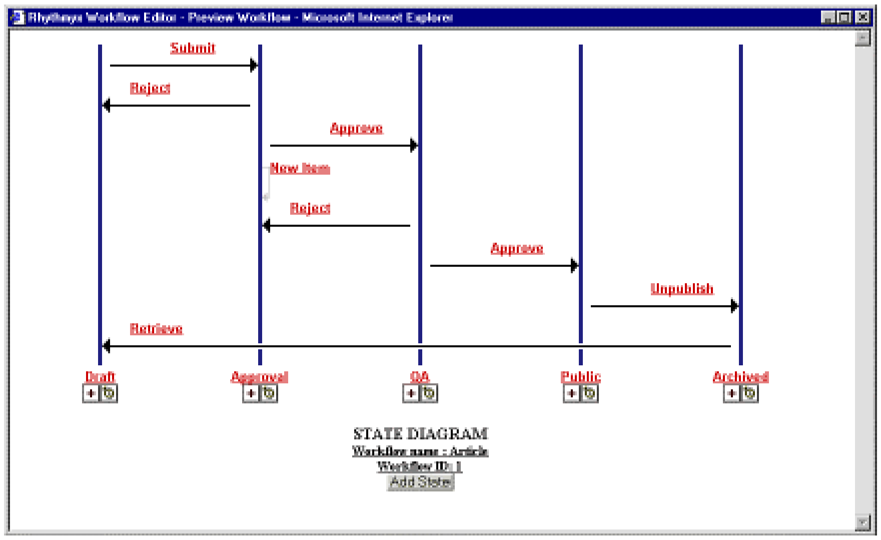


Figure 178: Workflow Preview

About the Workflow Administrator

The Workflow Administrator can edit and transition content in any state and has the ability to override the workflow process. The Workflow Administrator role gives assigned users these rights, and is otherwise like any other role.

Normally only the user who has checked out a content item may edit or check in those content items. If that user is unavailable, there is potential for a content item to get stuck in the workflow. Without administration, you would have to modify the content record in the database itself to return the content item to an accessible condition. The Workflow Administrator role assigns designated end users the power to check in content items without manually changing the database.

Any Role assigned as Workflow Administrator must include the internal Member `rxserver`. This is a special Member used in the Aging process. If the Role assigned as Workflow Administrator does not include the Member `rxserver`, Aging Transitions will not occur because Workflow authentication will fail.

Mandatory Relationships and Workflows

If you choose to implement a mandatory Relationship, you must pay special attention to your Workflows.

Since mandatory Relationships require that the Owner and Dependent in the Relationship both go Public together, a poorly designed Workflow can allow content to become trapped, unable to progress to Public. Best Practice when designing Workflows that might be used by a mandatory Relationship is to include "pending" State immediately prior to the Public State. This State acts as a marshalling area for Content Items for which all work is effectively complete and which are ready to go Public, but which must wait for Dependent Content Items to reach the same State before they can make the final Transition to Public.

Another issue to consider is whether you want to force Transitions on Content Items. You might want to force Transitions in two cases:

- Several Content Items are waiting in a "pending" State. When you Transition one of them, you want to Transition all of them.
- A Content Item is in a "pending" State, but it's Dependent is not there yet. You want both to go Public regardless of the current State of the Dependent Content Item.

To facilitate forced Transitions, you need to specify one of the Transitions from a State as the Default Transition. To make a Transition the Default Transition, choose Y from the Default Transition drop list on the when defining the Transition on the ***Edit Transition page*** (on page 247) in Content Explorer. (Note: If you specify more than one Transition as the Default Transition from a State, Rhythmyx uses the first Transition from the State in alphabetical order among those specified as a Default Transition.)

You will need an Effect to implement your forced Transition. For example of an Effect that forces a Transition, see the sys_PublishMandatory Effect.

Workflow Roles

A Role defines a group of individuals that can act on content in a State. The Roles used in a workflow may not correspond to job titles in your company. For example, if you create a marketing workflow, you might create Roles with the names Copywriter, Editor, and Approver. The individuals assigned to the Copywriter Role may carry the job title Copywriter, and the individuals assigned the Editor Role may carry the job title Editor, but the individuals assigned to the Approver Role probably don't have the job title Approver. It is more likely that they carry a job title such as Manager or Director or Vice-President.

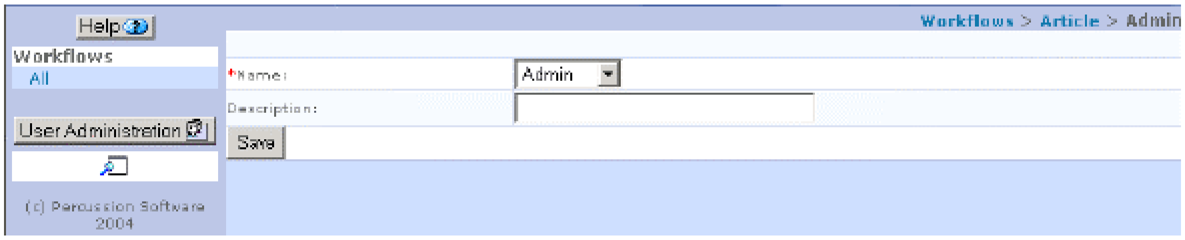
Roles allow you define groups that correspond to the needs of the workflow, rather than constraining the workflow to the structure of your organization.

You must define Roles before you can assign them to a State.

You should create separate Roles for Communities and Workflows. You should not assign the Roles that you associate with Communities to Workflows, and you should not associate the Roles that you assign with Workflows to Communities. This allows you to create parallel Workflows, or Workflows that that apply to Content in more than one Community. To transition a Content Item, a user must be in a Role assigned to the Content Item's Community and in a Role assigned to the transition.

Edit Role Page in Workflow

Use the Edit Role page to maintain Roles.



*Name: Admin

Description:

Figure 179: Edit Workflow Role

To access the edit Role page:

- On the Edit Workflow page, click **Role** in the New box (displays the page blank as the New Role page);
- On the Edit Workflow page, click the name the Role you want to edit;

Edit Role Fields

Name Mandatory. The name of the Role you want to maintain. Roles must be defined in the Rhythmyx Server Administrator as well as in Workflow in order to function in the workflow. To access Role Maintenance in the Server Administrator, click the User Administration button.

Description Optional. A free-form description of the Role.

Creating a New Role in Rhythmyx Workflow

NOTE: To function properly, any Role defined in the Workflow Editor must also be defined in the Server Administrator. Defining the Role in the Server Administrator associates individual users with the Role so they can access content items. To access Role Maintenance in the Server Administrator, click the User Administration button. For more details, see ***"Maintaining Server Role**s"*

To create a Role in Rhythmyx Workflow:

1. On the Edit Workflow page click **Role** in the New box.

   Rhythmyx displays the New Role for Workflow page, which is a blank version of the Edit Role page.
2. Enter the **Name** of the Role and optional **Description**.
3. Click [**Save**].

Rhythmyx adds the Role to the workflow.

NOTE: To allow a Workflow to apply to more than one Community, do not assign Roles associated with Communities to Workflows. The user assigned to a Workflow Role may have a separate Community Role.

Editing a Role in Workflow

NOTE: To function properly, any Role defined in the Workflow Editor must also be defined in the Server Administrator. Defining the Role in the Server Administrator associated individual users with the Role so they can access content items. To access Role Maintenance in the Server Administrator, click the User Administration button.

To edit a Role:

1. Access the Edit Workflow page of the workflow whose Role you want to edit.
2. Click the name of the Role you want to edit.

   Rhythmyx displays the Edit Role page.
3. Make your changes. You can change the **Name** or **Description** of the Role.
4. Click [**Save**].

Rhythmyx saves your changes.

Deleting a Role in Workflow

Deleting a Role also deletes any assignments of the Role to any States in the workflow.

To delete a Role:

1. Access the Edit Workflow page.
2. Click the Delete button ✕ in the row of the Role you want to delete.

Rhythmyx displays a confirmation message.

3 To confirm the delete action, click [**OK**]. To abort the delete action, click [**Cancel**].

Maintaining Server Roles

You must define Roles on the Rhythmyx server as well as in your Workflows in order for them to function. If you do not define the Roles on the server, Rhythmyx cannot recognize them and Workflow functionality, such as Transitions, will not be available.

To maintain Roles on the Rhythmyx server, click the User Administration button. Rhythmyx pops up a browser displaying the login to the Rhythmyx Administration Console. Enter your **User ID** and **Password**, then click [**Login**]. Rhythmyx displays the Server Administrator.

Roles

A Role is a collection of users or groups of users. In Rhythmyx, Roles are used to manage access to specific applications and to manage access to content items in the Workflow.

Access to applications is granted based on the Roles to which a user belongs. Organizing users into Roles helps you manage users that have the same permissions. Instead of managing the permissions for each user, you define a Role and the permissions for it, then assign users to it. Users assigned to that Role have the permissions specified for that Role. When you assign a Role to the Access Control List of an application, the users in that Role have access to that application.

In Workflow, Roles determine which users have access to a content item and can act on it when it is in a particular State. Roles also determine which users are notified when a content item makes a Transition into or out of a State.

Each Role, and each Member in a Role, has a set of properties. Properties are defined by the administrator to provide additional options for customization. For example, a Role might have the property sys_community. The value of this property specifies the Community to which the members of the Role belong. Rhythmyx uses this property to implement the Communities feature, filtering the content to which the user has access, the tabs on the Content Explorer that the user can see, and the Content Types and Variants available to the user.

To maintain Roles, use the Roles tab of the Security tab in the System Administrator. Two views are available. You can view a list of Roles and the Members associated with each Role (Members by Role) or you can view a list of Members and the Roles associated with each Member (Roles by Member).

Select the option you want from the **View** field.

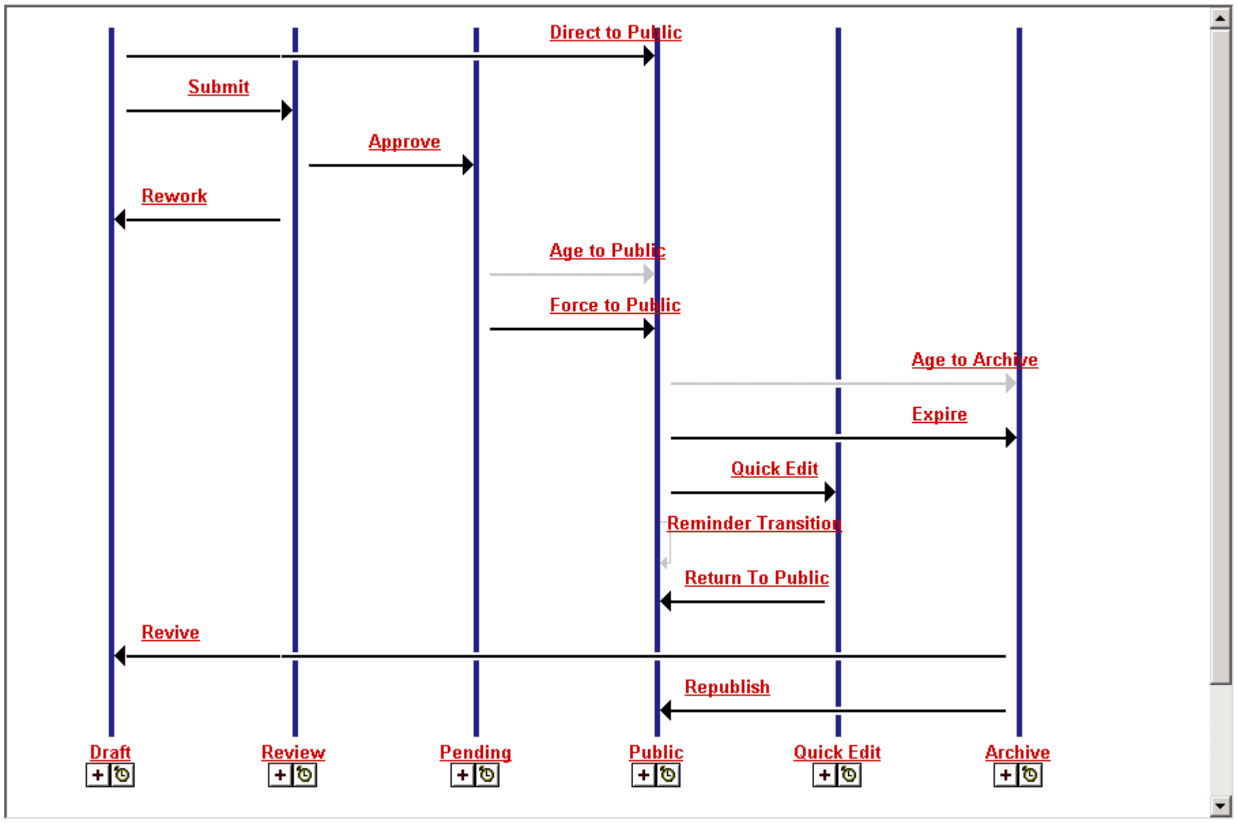


Figure 180: Edit Roles Tab in the Rhythmyx Server Administrator

Add/Edit Role Dialog

The New Role dialog and Edit Role dialog are identical, but the Name field is unavailable when the dialog is displayed as the Edit Role dialog. Users can edit the properties associated with the Role but not the name of the Role. To change the name of the Role, you must ***delete the Role*** (see "Deleting a Role" on page 235) and ***add a new Role*** with the new name.

To access the New Role dialog, click the Security tab of the Rhythmyx Server Administrator, then click Role tab. On the Role tab, click the [**Add Role**] button.

To access the Edit Role dialog, click the Security tab of the Rhythmyx Server Administrator, then click the Role tab. Select the Role whose properties you want to edit and click the [**Edit**] button.

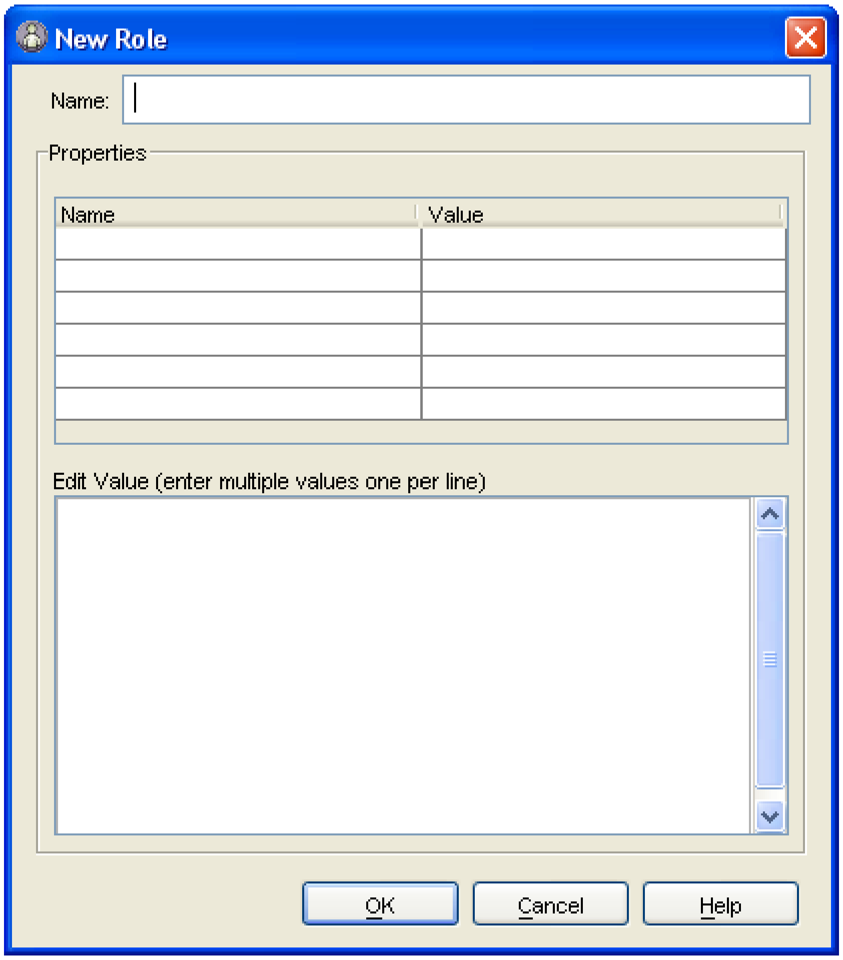


Figure 181: New Role Dialog

Field Descriptions

Name - Mandatory on New Role dialog; read-only on Edit Role dialog. The name of the Role.

Properties - Name Optional. The name of the property.

Properties - Value Read-only. The value of the property.

Edit Value - Free-form field to enter values for the selected property. Enter each separate value on a different line.

Adding a New Role

To add a New Role:

1. In the Rhythmyx Server Administrator, click the Security tab along the top, then the Roles tab along the bottom.
2. Click [**Add role**].

   Rhythmyx displays the ***Add Role dialog*** (see "Add/Edit Role Dialog" on page 232).

3 Enter the Name of the new Role.

4 Add any Properties to the new Role. To add a property:

a) Click in an empty row under Name.

b) Rhythmyx displays a drop list of available properties.

c) Select the property you want to assign to the Role.

d) To add a value to the property, click in the same row under Value.

- If values are pre-defined (see Maintaining Role and Member Property Names and Values), Rhythmyx will display the available properties in a drop list. Select the value you want to assign to the property.
- If values are free-form, Rhythmyx will move your cursor to the Edit Value field, where you can enter the values. Enter each unique value on a separate line.

5 Click [**OK**] to save the new Role.

Editing a Role

You can edit a Role to change the properties associated with that Role.

To edit a Role:

1 In the Rhythmyx Server Administrator, click the Security tab along the top, then the Roles tab along the bottom.

2 Select the Role whose properties you want to edit.

3 Click the [**Edit**] button.

Rhythmyx displays the ***Edit Role dialog*** (see "Add/Edit Role Dialog" on page 232).

4 To add a property:

a) Click in an empty row under Name.

b) Rhythmyx displays a drop list of available properties.

c) Select the property you want to assign to the Role.

d) To add a value to the property, click in the same row under Value.

- If values are pre-defined (see Maintaining Role and Member Property Names and Values), Rhythmyx will display the available properties in a drop list. Select the value you want to assign to the property.
- If values are free-form, Rhythmyx will move your cursor to the Edit Value field, where you can enter the values. Enter each unique value on a separate line.

5 To delete a property, right click on the property and select *Clear* from the popup menu.

6 To change the value of an existing property, click in the same row under Value.

- If values are pre-defined, Rhythmyx will display the available properties in a drop list. Select the value you want to assign to the property.
- If properties are free-form, Rhythmyx will display the current value of the property in the **Edit Value** field, with the cursor at the end of the last value. You can add a new value on a new line or modify or delete an existing value.

7 Click [**OK**] to save your changes.

Deleting a Role

When you delete a Role, you delete all Member associations with that Role.

To delete a Role:

1 In the Rhythmyx Server Administrator, click the Security tab along the top, then the Roles tab along the bottom.

2 In the **View** field, select Members by Role.

3 Select the Role you want to delete. You can select multiple Roles.

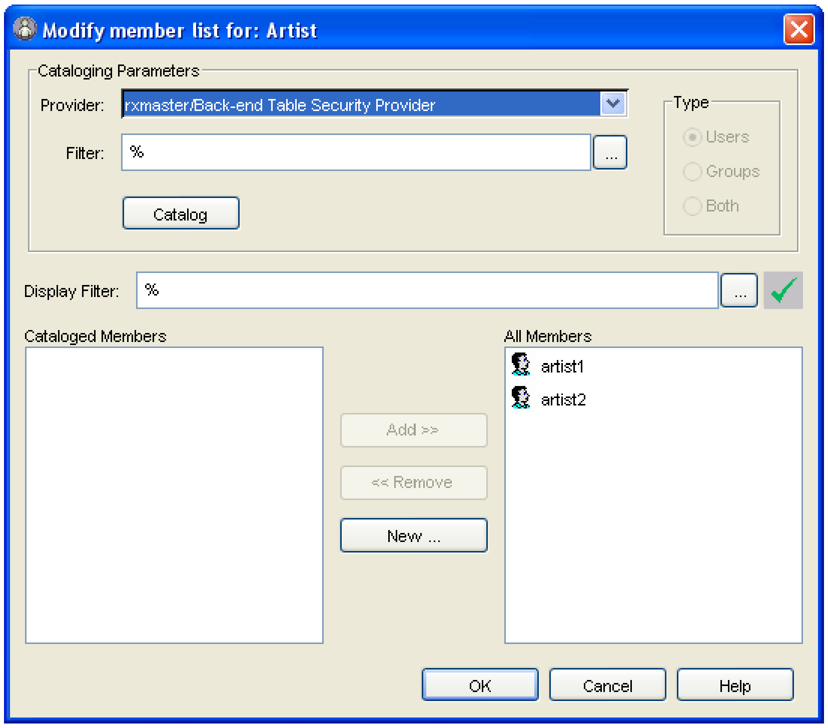
4 Click the [**Delete**] button.

5 Rhythmyx displays a confirmation message. Click [**Yes**] to confirm the delete action or [**No**] to abort the delete action.

Modify Member List for "Role" Dialog

Use the Modify Member List for <Role> dialog to ***add new members to a Role***.

To access the Modify Member List for <Role> dialog, on the Roles tab of the Rhythmyx Server Administrator, select the Role whose Member list you want to modify and click [**Add Members**].


Modify member list for: Artist

Cataloging Parameters

Provider: rxmaster/Back-end Table Security Provider

Filter: %

Type: [✓ Users] Groups Both

Display Filter: %

Cataloged Members

All Members

artist1

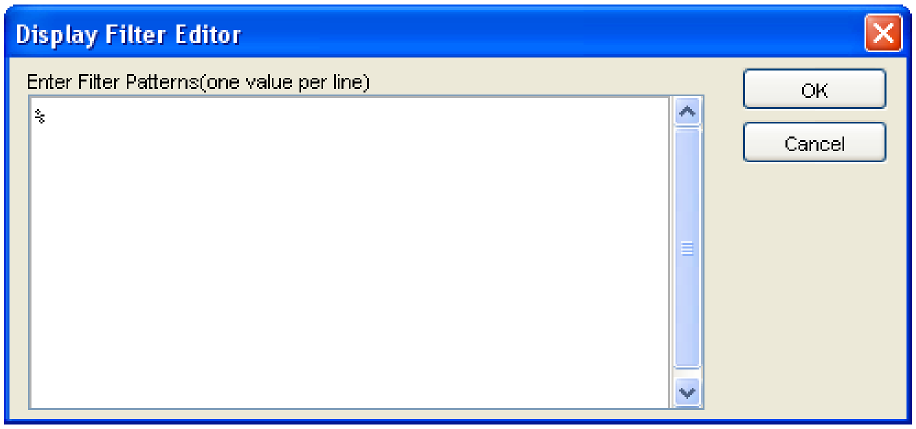
artist2

Figure 182: Modify Member List for <Role> Dialog

Field Descriptions

Provider - Drop list of registered security providers.

Filter - Optional. Use the ... button to display the Catalog Filter Editor dialog:



where you can enter one or more characters to use to filter the return from the security provider.

Display Filter - Optional. This field functions in the same fashion a the Filter field, but filters only the cataloged Members displays.

Cataloged Members - List of Members returned from the security provider.

All Members - List of Members assigned to the Role.

Adding Existing Members to a Role

You can add members that already exist in a security provider to a Role. The Server Administrator includes a cataloging function that retrieves a list of users associated with a specified security provider.

To add Existing Members to a Role:

1 In the Rhythmyx Server Administrator, click the Security tab along the top, then the Roles tab along the bottom.

2 Select the Role to which you want to add Members.

3 Click [**Add Member(s)**].

Rhythmyx displays the Modify Member List for <Role> dialog.

4 Select the security **Provider** from which you want to select Members.

5 Choose the Type of Member you want to add. Options are *Users* (add individual Members), *Groups* (add groups of Members) or *Both* (add both individual Members and groups of Members). NOTE: Not all security providers support Groups. If the security provider you select does not support Groups, no Groups will be retrieved.

6 Enter an optional Filter. Use SQL syntax to define the values. Two wildcard characters are available: % matches 0 or more characters, while "_" matches any single character. (For example, to catalog the name admin1, you would enter *admin1*. This entry would return only admin1. If you wanted to catalog all names that begin admin, you would enter *admin%*. This entry would return *admin1*, *admin2*, etc. If you wanted to return all names that begin with "ad", you would enter *ad%*. This entry would return Adams, Adkins, admin1, admin2, etc. Similarly, if you entered *ols_n*, Rhythmyx would return Olsen, Olson, and Olsun.

   Separate multiple search values with semicolons. You can also use the Catalog Filter Editor dialog to enter a filter:

   a) Click the ... button.

      Rhythmyx displays the Catalog Filter Editor dialog.

   b) Enter values. Each value should be entered on a separate line.

   c) Click [**OK**] on the Catalog Filter Editor dialog.

7 Click the [**Catalog**] button.

8 Rhythmyx displays a list of Members that matches the parameters you defined. You can define a filter for this list in the Display Filter field. The same options are available for this field as for the catalog filter. Click the ✓ button to activate the display filter.

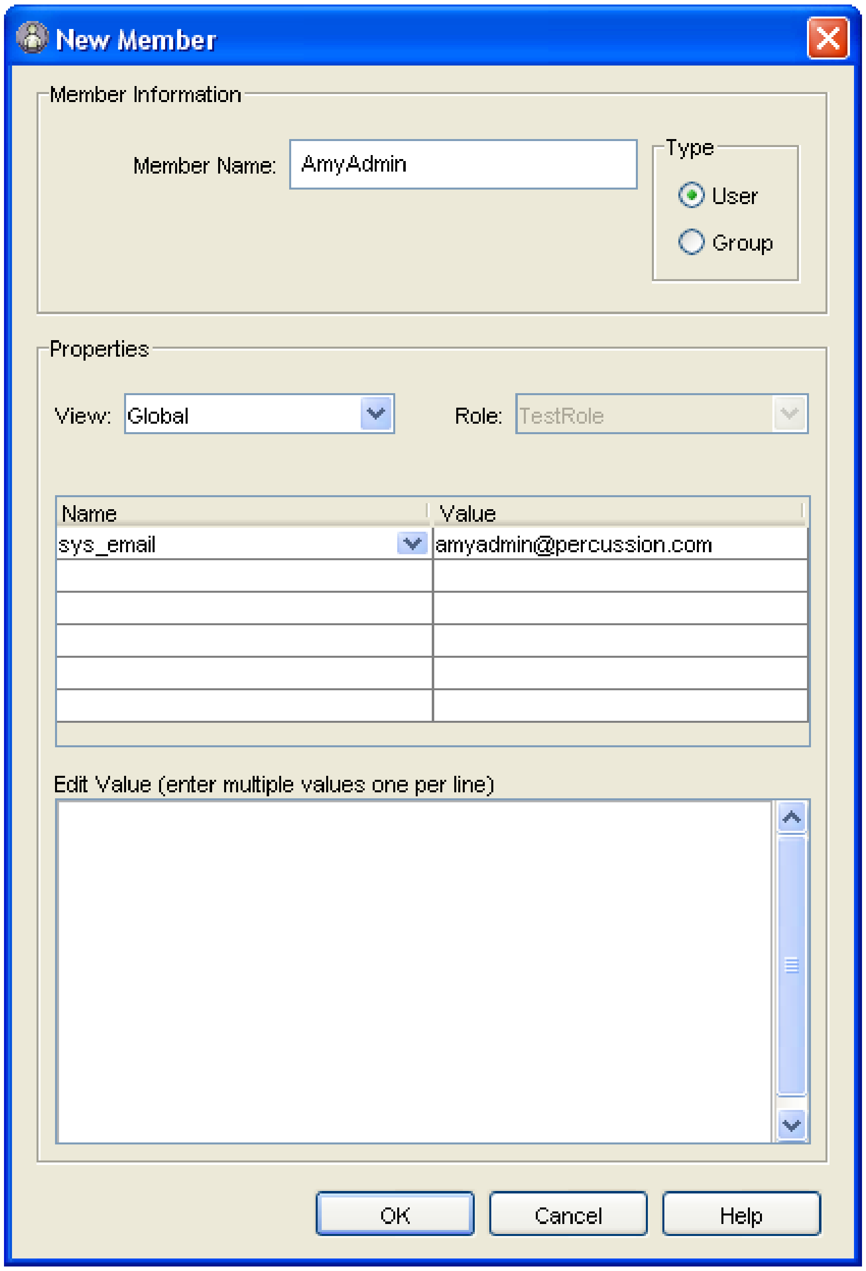
9 Select the Members you want to add and click the [**Add>>**] button.

10 Click [**OK**] to save your edits.

New Member Dialog

Use the New Member dialog to ***add a new Member to a Role without cataloging the Member from a security provider***. The Edit Member Properties dialog is a variation of this dialog.

To access the New Member dialog, click [**New**] on the ***Modify Member List for <Role> dialog***.



Member Name: AmyAdmin

Type: [✓ User] Group

View: Global

Role: TestRole

Name	Value
sys_email	amyadmin@percussion.com

Figure 183: New Member Dialog

Field Descriptions

Member Name - Mandatory. The name of the Member you are adding to the Role.

Provider - Drop list. The security provider for the Member.

Type - Radio buttons. Indicates whether the new Member is a Group or an individual User.

View - Drop list. Specifies which properties you are viewing and editing. Options are Global (edit all Member properties) and Role (edit only properties associated with a specific Role).

Role - Drop list. Only available if the value of View is Role. The Role for which you are viewing and editing properties.

Name - The name of the property.

Value - The value of the property.

Edit Value - Free-form field to enter values for the selected property. Enter each separate value on a different line.

Adding New Members to a Role

To add a new Member to a Role without Cataloging:

1. In the Rhythmyx Server Administrator, click the Security tab along the top, then the Roles tab along the bottom.
2. Select the Role to which you want to add Members.
3. Click [**Add member(s)**].

   Rhythmyx displays the Modify Member List for <Role> dialog.
4. On the Modify Member List for Role dialog, click the [**New**] button.

   Rhythmyx displays the Member Properties dialog.
5. Enter the **Name** of the new Member.
6. Select a security **Provider** for the new Member.
7. Choose the Type of new Member. Options are **Users** (individual Member) or **Group** (group of Members).
8. Select the Properties you want to define. Options are *Global* (properties not associated with a specific Role) and *Role* (properties for a specific Role). If you select *Role*, you must select the Role for which you want to define properties. Options for the Role field are all Roles associated with the Member. Note that any Global property will override a property of the same name from the security provider.
9. To select a property, click in an empty row under Name and select a property from the drop list.
10. To add a value to a property, click in the same row under Value.
    - If values are pre-defined, Rhythmyx will display the available properties in a drop list. Select the value you want to assign to the property.
    - If values are free-form, Rhythmyx will move your cursor to the Edit Value field, where you can enter the values. Enter each unique value on a separate line.
11. Click [**OK**] to save the Member record.

Role and Member Properties

Role and Member properties provide a generic mechanism for storing information about users that Rhythmyx can use for processing. Properties may be associated with a Role globally (Role Properties), with individual users (Member Properties), or with individual users only within a specific Role (Role Member Properties).

In general, Role and Member properties are used either to customize interfaces or to facilitate interactions between the end-user and the Rhythmyx server. The Communities feature is an example of Role Properties used for interface customization. One of the Role Properties is the Community to which that Role is assigned. The content types, content items, workflows, and other interface features available to the users in a Role depend on the value of the Role Property Community. An example of a Member Property used to facility interaction between the server and the end-user is the Notification feature of Workflow. The user's e-mail address is defined as a Member Property, and Rhythmyx sends Notifications to this address when a content item makes a Transition into a State to which the user's Role is assigned.

Editing a Member's Properties

To edit a Member's properties:

1. In the Rhythmyx Server Administrator, click the Security tab along the top, then the Roles tab along the bottom.
2. Select the Member whose record you want to edit.
3. Click the [**Edit**] button.

   Rhythmyx displays the Edit Member Properties dialog (same as New Member dialog).
4. Select the **Properties** you want to View. Options are *Global* (view all properties) and *Role* (view properties associated with a specific Role). Note that any Global property will override a property of the same name from the security provider.
5. To delete a property, right click on the property and select *Clear* from the popup menu.
6. To change the value of an existing property, click in the same row under Value.
   - If values are pre-defined, Rhythmyx will display the available properties in a drop list. Select the value you want to assign to the property.
   - If values are free-form, Rhythmyx will display the current value of the property in the Edit Value field, with the cursor at the end of the last value. You can add a new value on a new line or modify or delete an existing value.
7. Click [**OK**].

Deleting a Member from a Role

To delete a Member from a Role:

1. In the Rhythmyx Server Administrator, click the Security tab along the top, then the Roles tab along the bottom.
2. You have two options to delete a Member from a single Role:
   - In the **View** field, select *Roles by Member*, expand the Role from which you want to delete the Member, and select the Member.
   - In the **View** field, select *Members by Role*, expand the Member you want to delete from a Role, and select the Role from which you want to delete the Member.

3. To delete a Member from all Roles, in the **View** field, select *Roles by Member*, then select the Member you want to delete.
4. Click the [**Delete**] button.
5. Rhythmyx displays a confirmation message. Click [**Yes**] to confirm the delete action or [**No**] to abort the delete action.

States

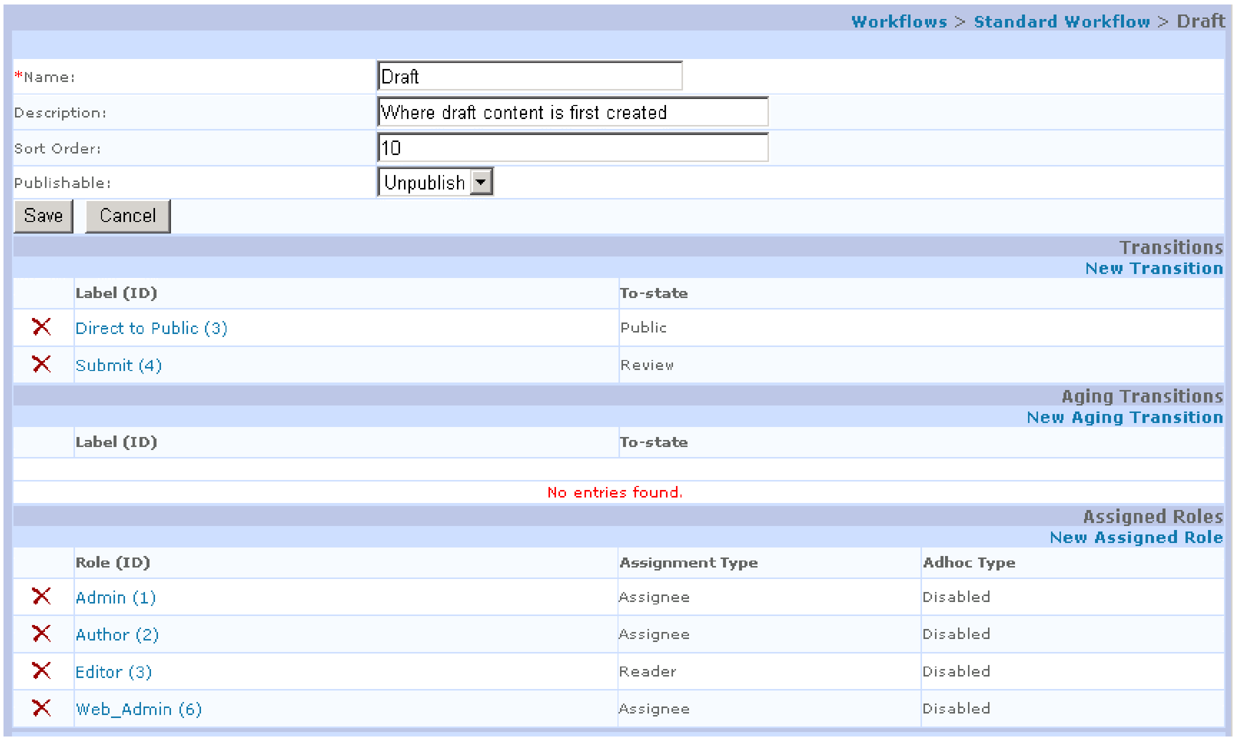
A State is a step in the workflow process of a document. While in a State, a user may be able to modify, review, or approve a document.

Edit State Page

Use the Edit State page to maintain States.

You can access the Edit State page in the following ways:

- On the Edit Workflow page, click **State** in the New box (accesses as New State page);
- On the Edit Workflow page, click the name of the State you want to edit.



Workflows > Standard Workflow > Draft

*Name:	Draft
Description:	Where draft content is first created
Sort Order:	10
Publishable:	Unpublish

Transitions

New Transition

	Label (ID)	To-state
✕	Direct to Public (3)	Public
✕	Submit (4)	Review

Aging Transitions

New Aging Transition

	Label (ID)	To-state

No entries found.

Assigned Roles

New Assigned Role

	Role (ID)	Assignment Type	Adhoc Type
✕	Admin (1)	Assignee	Disabled
✕	Author (2)	Assignee	Disabled
✕	Editor (3)	Reader	Disabled
✕	Web_Admin (6)	Assignee	Disabled

Figure 184: Edit State Page

Field Descriptions

Name Mandatory. The name of the State.

Description Optional. Free-form description of the State.

Publishable Drop List. Indicates how to treat content in the State when Publishing content. The default installation of Rhythmyx includes the following values for this property:

Value	Processing
Unpublish	Default. Content in this State is not published, or will be unpublished the next time the Publisher runs.
Publish	Content in this State is published when the Publisher runs.
Ignore	Do not evaluate whether content in this State is eligible to be published. If the content has already been published, it remains Public. Any links to the item will also be published. If it has not yet been published, it will remain unpublished. Links to the item also remain unpublished.

NOTE: You can extend the Publishable property by adding more values to the property. For details, see ***Extending Publishable States*** (on page 245).

Creating a New State

To create a new State:

1. Access the New State page.
2. Enter the State **Name** (required) and optional **Description**.
3. **Sort Order**: Enter a number to specify the position of this state name in the list of state names displayed for the workflow on the Content Explorer Workflow tab. You can use any integer. Lower numbers are higher in the list as you can see in the list of States for the Simple Workflow shipped with Fast Forward. (The sort order is used only for this list.)
4. **Publishable**: Choose a value from this drop list to specify what Rhythmyx should do with content items that are in this state.
   - Choose Unpublish when Rhythmyx should unpublish items in this state.
   - Choose Publish when Rhythmyx should publish items in this state.
   - Choose Ignore when Rhythmyx should not evaluate whether content in this State is eligible to be published. If the content has already been published, the last Public version remains Public. Any links to the item will also be published. If it has not yet been published, it will remain unpublished. Links to the item also remain unpublished. This is useful if an item is in the Quick Edit state; it tells Rhythmyx to ignore the Quick Edit version and to publish the last version that reached the Public state.
5. Click [**Save**] to add the State to the workflow.

Editing a State

To edit a State:

1 Access the Edit State page for the State you want to edit.

2 You can modify any of the three fields in the State.

3 Click [**Save**].

Rhythmyx updates the data for the State.

Deleting a State

When you delete a state, you delete all the Transitions, Notifications, and State-assigned Roles associated with the State you are deleting.

To delete a state from a workflow:

1 Access the Edit Workflow page for the workflow from which you want to delete a State.

2 Click the Delete button ✕ the row of the State you want to delete.

Rhythmyx displays a confirmation message.

3 To confirm the delete action, click [**OK**]. To abort the delete action, click [**Cancel**].

Extending Publishable States

The values of the Publishable property for States are maintained in the Rhythmyx Keyword Editor on the System tab of the Content Explorer. You can thus add more values to the Publishable property to extend it and make it more flexible.

WARNING! Do not delete the default values of this Keyword. If you delete any of these default values, Publishing will no longer work correctly.

These values are used in the Content List queries that extract content for Publishing from the database. The value of Publishable is stored in the STATES.CONTENTVALID column in the database. When devising the SQL query for your content list, the WHERE clause of the query must include a phrase that selects Content Items in States where the value of STATES.CONTENTVALID equals the value of the Publishable property for the State whose content you want to Publish.

For example, suppose you wanted to implement a staging area where you could evaluate content before publishing it to your live web site. You could add a new value to the Publishable property, s (for staging). You would also create a State in the Workflow (perhaps also called Staging) and would assign S as the value of Publishable for this State.

When creating a Content List to publish content from this state, the WHERE clause would include the phrase

```
WHERE STATES.CONTENTVALID='s'
```

Set the Publishable value for a State using the ***Edit State Page*** (on page 243). The default Publishable values are:

Value	Processing
N	Default. Content in this State is not published, or will be unpublished the next time the Publisher runs.
Y	Content in this State is published when the Publisher runs.
I	Do not evaluate whether content in this State is eligible to be published. If the content has already been published, it remains Public. Any links to the item will also be published. If it has not yet been published, it will remain unpublished. Links to the item also remain unpublished.

Transitions

A Transition is a process that moves a document from one State to another State. A Transition specifies:

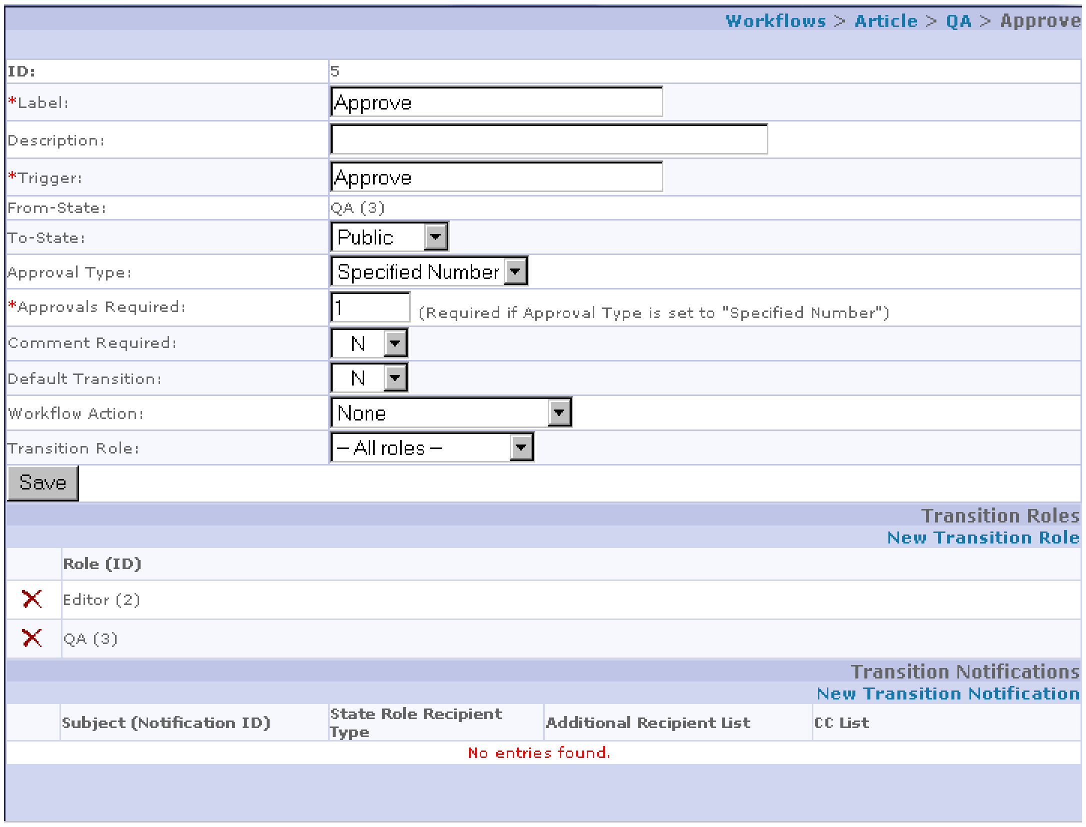
- when a document can move from a State;
- the new State to which the document will move.

Edit Transition Page

Use the Edit Transition page to maintain Transitions.

You can access the Edit Transition page in the following ways:

- On the Edit Workflow page, click **Transition** in the New box (accesses as New Transition page);
- On the Edit Workflow page, click the name of the Transition you want to edit in the Transition box.



Workflows > Article > QA > Approve

ID:	5
*Label:	Approve
Description:	
*Trigger:	Approve
From-State:	QA (3)
To-State:	Public
Approval Type:	Specified Number
*Approvals Required:	1 (Required if Approval Type is set to "Specified Number")
Comment Required:	N
Default Transition:	N
Workflow Action:	None
Transition Role:	-- All roles --

Transition Roles

	Role (ID)
✗	Editor (2)
✗	QA (3)

Transition Notifications

	Subject (Notification ID)	State Role Recipient Type	Additional Recipient List	CC List
	No entries found.			

Figure 185: Edit Transition Page

Edit Transition Fields

ID Display only. The identifier associated with this Transition.

Label Mandatory. The label for this Transition.

Description Optional. Free-form description of the Transition.

Trigger Mandatory. The pre-requisite for the Transition. Each Transition in the State requires a unique value in the Trigger field.

From State Display Only. The State from which the content is moving in the workflow.

To State Drop list. The State to which the content will move in the workflow. It is valid for a Transition to move the content to the same State as it came from. This is known as a *circular transition*.

Approval Type Drop List. Specifies whether the Transition requires a specific number of approvals or approvals by specified Roles before the content can make this Transition. Options are *Specified Number* (specific number of approvals required for the Transition; approvals may come from any Role assigned to the current State of the content) and *Each Role* (specific Roles must approve content before it can make this Transition).

Approvals Required Required if the value of **Approval Type** is *Specified Number*. Number of approvals required before the content can make this transition.

Comment Required Drop list. Whether the user has the option to enter a comment, or is required to make a comment when making this Transition. Choices are *Optional* (Rhythmyx displays the comment popup, but the user can choose not to make a comment), *Required* (user must enter a comment when making the Transition), and *Do Not Show* (Rhythmyx does not display the comment popup).

Default Transition Drop list. Specifies whether the Transition is the default Transition used when a Relationship forces a Transition from the State. Options are *No* (default; the Transition is not the default Transition) and *Yes* (the Transition is the default Transition). NOTE: Rhythmyx does not validate that a State has only one default Transition. If multiple Transitions from a State have *Yes* selected in the **Default Transition** field, Rhythmyx uses the first in alphabetical order specified as a default Transition.

Workflow Action Drop list. The Workflow Action associated with the Transition. Default is *None*. Options include all Workflow Actions registered in the system.

Transition Role Drop list. Specifies whether a Member of each of the Roles assigned to the State must approve the Content Item before it can make the Transition, or only a Member of specific Roles. Options are *All Roles* (a Member of each of the Roles assigned to the State must approve the content before it can make the Transition) and *Specified Roles* (a Member of each Role specified must approve the content before it can make the Transition.).

If you want to notify users when content makes the Transition, add Transition Notifications.

Transitions Roles are required if you choose Each Role in the Approval Type field and Specified Roles in the Transition Role field for the Transition. Note that Rhythmyx does not preserve approvals. For example, suppose you have a Transition "Ready for Publish" from the QA State to the Public State, and this Transition requires approval from three Roles, QA, Editor, and Legal Reviewer. The QA and Editor Roles approve the Content Item, but the Legal Review Role rejects it, sending it back to a previous State. When the Content Item returns to the QA State, Rhythmyx does not remember that QA and Editor already approved it. They will have to approve it again before it can make the "Ready for Publish" Transition.

Creating a New Transition

To create a new Transition:

1 Access the New Transition page.

2 Enter the **Label** (name) of the Transition (this field is mandatory) and **Description** of the Transition.

3 Enter a **Trigger** for the Transition. A Trigger is a string Rhythmyx uses to start processing the Transition and is mandatory.

4 Select the State the document will move to in the Transition **To State** field.

5 Choose the **Approval Type**. Options are *Specified Number* (specific number of approvals required for the Transition; approvals may come from any Role assigned to the current State of the content; if you choose this option, you must specify the number of **Approvals Required**) and *Each Role* (specific Roles must approve content before it can make this Transition; you must specify the Roles the must approve the content).

6 If you chose *Specified Number* in the **Approval Type** field, enter the number of **Approvals Required** for the Transition.

7 Select an option for **Comment Required**. Choices are *Optional* (Rhythmyx displays the comment popup, but the user can choose not to make a comment), *Required* (user must enter a comment when making the Transition), and *Do Not Show* (Rhythmyx does not display the comment popup).

8 If **Workflow Actions** are available, select the action you want to associate with the Transition.

9 If you chose *Each Role* in the **Approval Type** field, choose the **Transition Roles** option for the Transition. Options are *All Roles* (default; a Member of each of the Roles assigned to the State must approve the Transition) and *Specified Roles* (a Member of each Role specified must approve the content before it can make the Transition.).

10 If you chose *Each Role* in the **Approval Type** field and *Specified Roles* in the **Transition Roles** field, you must add ***Transition Roles*** (see "Maintaining Transition Roles" on page 250).

11 Add ***Transition Notifications*** (see "Notification Assignment" on page 263), if necessary.

12 Click [**Save**].

Editing a Transition

To edit a Transition:

1 Access the ***Edit Transition page*** (on page 247) for the Transition you want to edit.

2 Make your changes. The **Label**, and **Trigger** fields are mandatory. You cannot change the **ID** or **From State**, but can change all other fields, as well as add or delete Transition Roles and add, modify, or delete Transition Notifications.

3 Click [**Save**] to save your changes.

Deleting a Transition

To delete a Transition:

1. Access the Edit Workflow page of the workflow from which you want to delete the Transition.
2. Click the Delete button ✕ of the row of the Transition you want to delete.

   Rhythmyx displays a confirmation message.
3. To confirm the delete action, click [**OK**]. To abort the delete action, click [**Cancel**].

Maintaining Transition Roles

If you define a Transition to require approvals by specified Roles (in other words, if you chose *Each Role* in the **Approval Type** field and *Specified Roles* in the **Transition Roles** field, you must add Transition Roles to the Transition to specify the Roles that must approve the Content.

To add a Transition Role:

1. Click Add Transition Role.
2. Rhythmyx displays the Add Transition Role page.
3. Choose the **Role** you want to add. Options are all Roles in the Workflow not currently assigned as Transition Roles to this Transition.
4. Click [**Save**].

To delete a Transition Role, click the delete button ✕ in the row of the Transition Role you want to delete.

Aging

Aging is a Rhythmyx Workflow feature that defines the amount of time a content item can reside in a State before Rhythmyx automatically acts on it.

To implement Aging, you assign special Aging Transitions to a State where you want Aging process to take place. When you define the Transition, you specify the type of Aging processing you want to use for the State.

- Repeated

  A Repeated Aging Transition repeats after an interval specified in the Transition. Repeated Aging Transitions are most commonly used to send the same reminder Notification each time a specified period has passed. To work correctly, the To State of the Transition must be the same as the From State. If the To State of the Transition is different than the From State, the content item will Transition to the new State and the Aging will not repeat.
- Absolute

  An Absolute Aging Transition occurs once, after a specified period of time has passed. Absolute Aging Transitions are typically used to send escalating reminder Notifications before automatically Transitioning to a new State.

- System Field

  A System Field Transition occurs once after the specified system date (Start Date, End Date, Reminder Date) passes.

Typically, an Aging Transition also sends an e-mail message that notifies the users in the Role assigned to the Transition that the content item has aged and that some action is necessary.

Aging Prerequisites

In order for Aging to function, you must complete the following prerequisites:

- Set the mail host and mail domain in the rxworkflow.properties file.
- Define user e-mail addresses. User e-mail addresses are specified on the ***user's Member Properties in Role Main***tenance (see "Editing a Member's Properties" on page 240). Use the Global property *sys_email* to define the user's e-mail address. The User Administration button activates the Role Maintenance dialog of the Server Admininstrator. The sender of aging e-mail messages is the user `rxserver`. The e-mail address you define for rxserver should be the address of the person that should receive notifications of bounced e-mail messages.
- Activate Notification for the Role assignments to each State in which Aging will occur. To activate Notification, set the value of the Notify field on the Role Assignment page to *Y*. Leave notification off for any Role that should not receive notification e-mail messages. For example, the Workflow admin Role generally does not need to receive Aging e-mail messages.

Setting Aging Agent Parameters

The Java class com.percussion.server.agent.PSAging is the Rhythmyx Aging agent and executes the Aging function. The registration entry of this agent in the agentmanager.xml (`Rhythmyxroot/rxconfig/Server/requestHandlers /agentmanager.xml`) defines how often Rhythmyx will poll for Aging Transitions.

The default entry for this agent is:

```
<agent name="aging" servicetype="scheduled">
  <class>com.percussion.server.agent.PSAging</class>
  <schedule delay="600" interval="600"/>
</agent>
```

The `<delay>` attribute of the schedule element defines the initial delay, the number of seconds after the server starts that must pass before the Aging agent executes the first Aging task. This delay is required to allow all Rhythmyx applications to start. The `<interval>` attribute of the schedule element defines the amount of time, in seconds between each execution of the Aging task. The default values specify that the Aging agent makes its first Aging ten minutes after the server starts, and will repeat the poll every ten minutes.

To change the frequency of polling for Aging Transitions, change the value of the `interval` attribute and save the `agentmanager.xml`. To change the amount of time available for your server to complete initialization before polling begins, change the value of the `delay` attribute.

Creating an Aging Transition

Aging is associated with the Transitions defined for a State. An aging Transition is a Transition that will occur automatically after a specified period if another Transition has not occurred.

In some cases, you may need to define more than one Transition to make your Aging scenario function. For more information, see "***Aging Scenarios*** (on page 253)".

To define an Aging Transition:

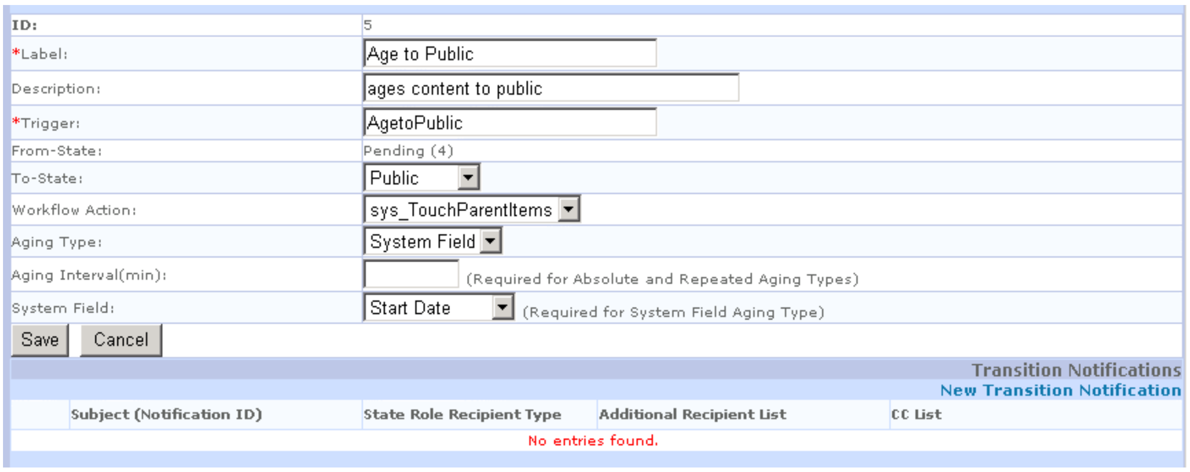
1. In the Rhythmyx Content Explorer, click the **Workflow** tab.

   Rhythmyx displays the Workflow Editor.
2. Click on the name of the Workflow to which you want to add an Aging Transition.

   Rhythmyx display the Edit Workflow page.
3. Click on the name of the State to which you want to add a Transition.

   Rhythmyx displays the Edit State page.
4. Click **New Aging Transition**.

   Rhythmyx displays the New Aging Transition page.



ID:	5
*Label:	Age to Public
Description:	ages content to public
*Trigger:	AgetoPublic
From-State:	Pending (4)
To-State:	Public
Workflow Action:	sys_TouchParentItems
Aging Type:	System Field
Aging Interval(min):	(Required for Absolute and Repeated Aging Types)
System Field:	Start Date (Required for System Field Aging Type)

Subject (Notification ID)	State Role Recipient Type	Additional Recipient List	CC List
No entries found.			

Figure 186: New Aging Transition Page

5. Enter the **Label** (name for the Transition; this field is mandatory) and **Description** of the Transition.
6. Enter a **Trigger** for the Transition. A Trigger is a string Rhythmyx uses to start processing the Transition and is mandatory.
7. Select the State the document will move to in the Transition **To State** field. The To State may be the current State or it may be another State in the Workflow.
8. Select the **Aging Type**. Options are:
   - *Absolute*: The Transition will happen after a specified period of time (defined in the **Aging Interval** field) has passed, unless another Transition occurs that supersedes this Transition or moves the content item to another State.

- *Repeated*: The Transition will repeat each time a specified period of time (defined in the **Aging Interval** field) has passed, until another Transition occurs that supersedes this Transition or moves the content item to another State.
- *System*: The Transition will occur once a specified system date (either the content item Start Date, the End Date, or the Reminder Date; the date must be specified in the **System Field** field) has passed. The Start Date and End Date are default system fields. If you want to use the Reminder Date, you must define the Reminder Date field in the Content Editor for the Content Type. (NOTE: If the user does not enter a date in the field you specified, the Content Item does not Age. To prevent this problem, you may want to use Field-level Validation to ensure that the field contains a valid date.)

9 Click [**Save**] to save the Transition.

10 Rhythmyx saves the Transition and return to the Edit State page.

11 Create a ***Notification*** (see "Creating a Notification" on page 258) and ***assign it to the Transition*** (see "Notification Assignment" on page 263).

Aging Scenarios

The most common aging scenarios are:

- A repeating notice until the content item Transitions to a new State.
- An automatic transition when a system date is reached.
- An escalating series of notices to the user to take action on an item.

Repeating Notices

Scenario: When a system's article content item enters the Approve State, the Editor Role is reminded every day that action is required.

The implementer creates a single Aging Transition, with the following data:

- The To State is *Approval* (which is the same as the From State; the Transition returns the content item to the same State).
- The Aging Type is *Repeating*.
- The Aging Interval is *1440* (the number of minutes in a day, 60 minutes per hour times 24 hours in a day).

The implementer also creates a Notification to the Members of the Editor Role that the content item needs action, and assigns this Notification to the Transition.

Each day the Members of the Editor Role have not Transitioned a content item to a different State manually, Rhythmyx sends them an e-mail message that the content item needs action.

Automatic Transition

Scenario: When a content item in the Approve State reaches its Start Date, the system automatically moves it to a Public State so it can be published.

The implementer creates a single Aging Transition, with the following data:

- The To State is *Public* (which is the Public State for this Workflow).
- The Aging Type is *System Field*.
- The System Field is *Start Date*.

When a content item is in the Approve State and its Start Date passes, Rhythmyx automatically Transitions it to the Public State, and it is published on the next run of the Publisher.

Escalating Notices

Scenario: A Workflow must specify that when a content item enters the approve State, Editors have three days to act on it. Each day they are sent a reminder that they must act on it or it will Transition automatically. If no one has acted on it in three days, the content item automatically moves to the QA State.

To implement escalating Notices, the implementer creates four Transitions:

- The first Transition is the normal Transition to the Approve State. The Notification assigned to this Transition informs the editors that they have three days to approve or reject the content item, or it will automatically be approved.
- The second Transition is an Aging Transition with the following data:
  - The To State is *Approval* (which is the same as the From State; the Transition returns the content item to the same State).
  - The Aging Type is *Absolute*.
  - The Aging Interval is *1440* (the number of minutes in a day, 60 minutes per hour times 24 hours in a day).
  - The Notification assigned to the Transition informs the editors that they have two days to approve or reject the content item or it will automatically be approved.
- The third Transition is an Aging Transition with the following data:
  - The To State is *Approval* (which is the same as the From State; the Transition returns the content item to the same State).
  - The Aging Type is *Absolute*.
  - The Aging Interval is *2880* (the number of minutes in a day, 60 minutes per hour times 24 hours in a day, times 2 for two days after the initial Transition).
  - The Notification assigned to the Transition informs the editors that they have one day to approve or reject the content item or it will automatically be approved.

- The fourth Transition is an Aging Transition with the following data:
  - The To State is *QA* (which is the next State after Approval; the Transition advances the content item in the Workflow automatically).
  - The Aging Type is *Absolute*.
  - The Aging Interval is *4320* (the number of minutes in a day, 60 minutes per hour times 24 hours in a day, times 3 for three days after the initial Transition).

Notifications

A Notification is an e-mail message that Rhythmyx automatically sends to a Role when a document enters a State.

Setting the Workflow Configuration Properties for Notification

For Notification to work, you must define the SMTP mail host in the `rxworkflow.properties` file, located in the Rhythmyxroot\rxconfig\Workflow directory. Set the value of `SMTP_Host` to the IP address or name of the SMTP mail server. Setting the value to the name of the server is generally preferable. While the IP address of the server may change, it is unlikely that the name of the server will change.

To prevent errors when trying to mail Notifications to users whose e-mail address is not defined in their Member Properties, set the `MAIL_DOMAIN` property to your domain name. If Rhythmyx needs to send a Notification to a user that does not have a value for sys_email, it will create an e-mail address by concatenating the user name and the value of the `MAIL_DOMAIN` property.

The aging agent requires host and port definitions to generate the correct links in Notification e-mail messages:

```
RX_SERVER_HOST_NOTIFICATION=<Rhythmyx server name or IP Address>
RX_SERVER_PORT_NOTIFICATION=<port number>
```

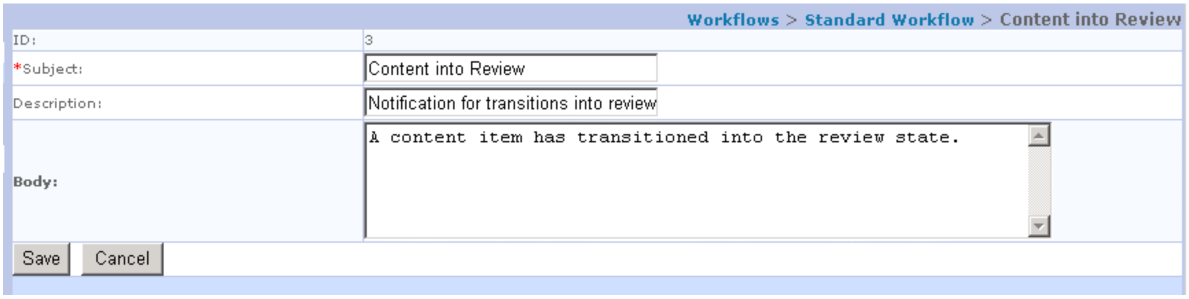
If you enable SSL on your server, you must also set the property `RX_SERVER_IS_SSLLINK_NOTIFICATION` to "yes" to ensure that the links in the Notification e-mail messages use SSL to link to Content Items on your server.

Edit Notification Page

Use the Edit Notification page to maintain the Notifications associated with a Workflow.

You can access the Edit Notification page in the following ways:

- On the Edit Workflow page, click New Notification;
- On the Edit Workflow page, click the name of the Notification you want to edit.



ID:	3
*Subject:	Content into Review
Description:	Notification for transitions into review
Body:	A content item has transitioned into the review state.

Figure 187: Edit Notification Page

Edit Notification Fields

ID Read Only. Unique identifier of the Notification.

Subject Mandatory. The subject line for the Notification e-mail message.

Body Optional. The body for the Notification e-mail message.

Creating a Notification

To create a Notification:

1. On the Edit Workflow page, click **New Notification**.

   Rhythmyx displays the New Notification dialog.
2. Enter the mandatory **Subject** and the optional **Body** of the e-mail sent as the Notification.
3. Click [**Save**] to save the Notification.

Editing a Notification

To edit a Notification:

1. On the Edit Workflow page, click the name of the Notification you want to edit.

   Rhythmyx displays the Edit Notification page with the current data for the Notification.
2. Change the **Subject** or **Body**.
3. When you have made your changes, click [**Save**].

Deleting a Notification

To delete a Notification:

1. On the Edit Workflow page, click the delete button ✕ of the Notification you want to delete.

   Rhythmyx displays a confirmation message.
2. To confirm the delete action, click [**OK**]. To abort the delete action, click [**Cancel**].

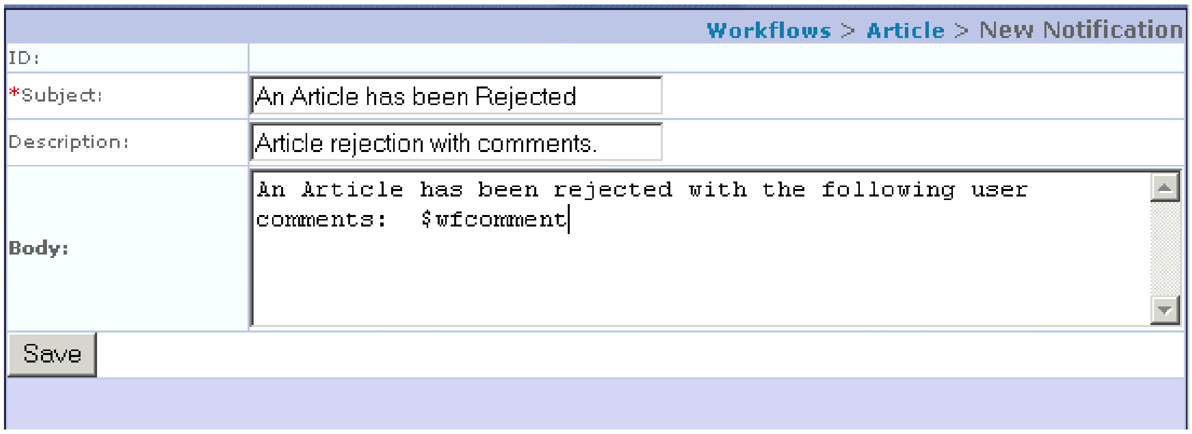
Including User Comments in Notification E-mail Messages

You may want to include user comments in the Notification e-mail messages. To include user comments enter the macro

```
$wfcomment
```

Note that the macroname is case-sensitive, and must be all lower-case.

The following graphic shows an example:

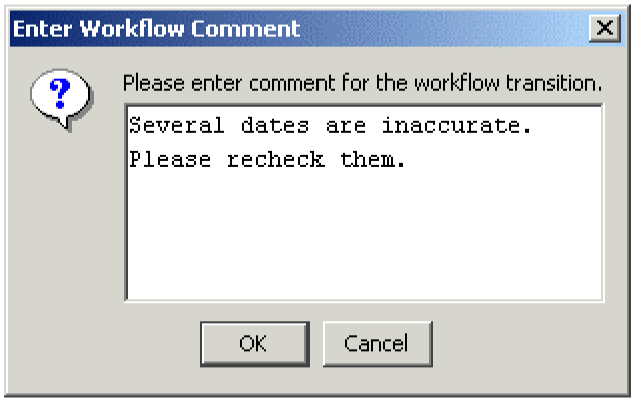


Workflows > Article > New Notification

ID:	
*Subject:	An Article has been Rejected
Description:	Article rejection with comments.
Body:	An Article has been rejected with the following user comments: $wfcomment

Figure 188: Example Notification using the $wfComment macro

This Notification is attached to the Reject Transition from Approval to Article. When the approver rejects the article, Rhythmyx asks for a comment:



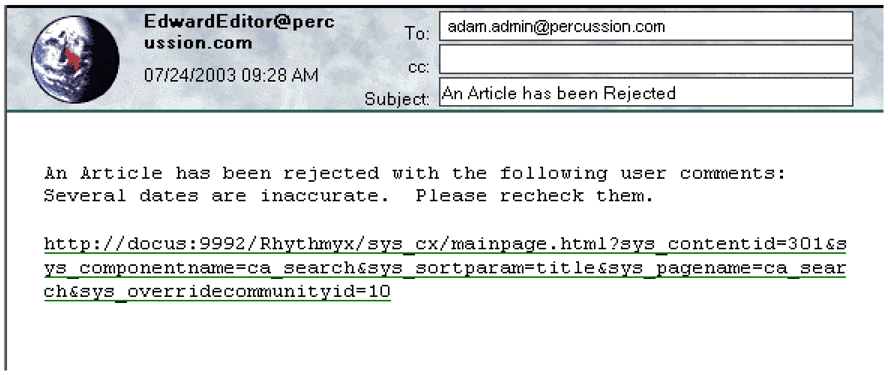
Enter Workflow Comment

Please enter comment for the workflow transition.

```
Several dates are inaccurate.
Please recheck them.
```

Figure 189: Sample user comment during Transition

This comment results in the following message:



EdwardEditor@percussion.com
07/24/2003 09:28 AM

To: adam.admin@percussion.com
cc:
Subject: An Article has been Rejected

```
An Article has been rejected with the following user comments:
Several dates are inaccurate.  Please recheck them.

http://docus:9992/Rhythmyx/sys_cx/mainpage.html?sys_contentid=301&s
ys_componentname=ca_search&sys_sortparam=title&sys_pagename=ca_sear
ch&sys_overridecommunityid=10
```

Figure 190: Notification e-mail message showing example comment

You can add the macro to either the subject or the body of the Notification message. The comments will be appended in all locations where you specify the macro.

To set a default value for the comment, add the following line to the workflow configuration properties file (`<rxroot>\rxconfig\workflow\rxworkflow.properties`):

```
    wfcomment=defaultcomment
```

Where `defaultcomment` is the comment you want Rhythmyx to include when the user does not specify a comment. Rhythmyx uses this value when the comment field is null or an empty string.

Assignment

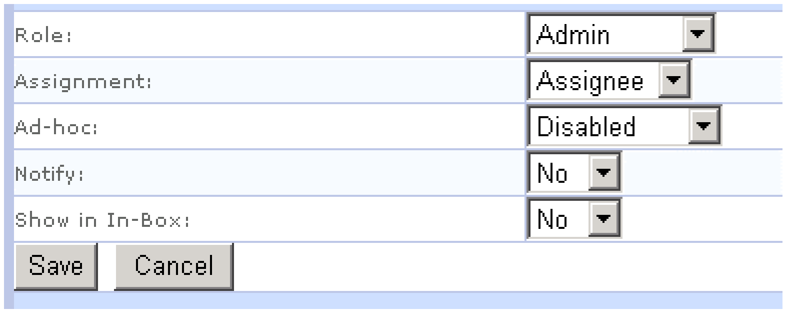
The Roles assigned to a State determine who can act on the State and who is Notified when a document enters and leaves the State. A Role must already exist in the Workflow before you can assign it to the State.

Notifications must be assigned to Transitions to prompt the system to send the Notification.

Role Assignment

Edit Assigned Role Page

Use the Edit Assigned Role page to maintain the association between a Role and a State.



Role:	Admin
Assignment:	Assignee
Ad-hoc:	Disabled
Notify:	No
Show in In-Box:	No

Figure 191: Edit Assigned Role Page

Field Descriptions

Role Drop list. The Role assigned to the State. Options are all Roles defined in the Workflow.

Assignment Drop list. Defines the access the Role has to content items in the State. Options are *None* (State is not assigned to anyone), *Reader* (Role has read-only access to content items in the State), and *Assignee* (Role has full access to content items in the State). If you choose *None,* the users in this Role cannot view Content Items in this State in the Content Explorer (in Views, Searches, or Folders).

Ad-hoc Drop list. Defines the access ad hoc assignees have to the State. Options are *Disabled* (default; ad hoc assignment is not permitted to the State), *Enabled* (ad hoc assignment is allowed to the State), and *Anonymous* (only Anonymous ACL access is permitted to the State).

Notify Drop List. Defines whether the Role is eligible to receive Notification e-mails triggered by Transitions to the State. Options are *Y* (default; Notification is allowed) and *N* (Notification is not allowed).

Show in Inbox Defines whether to show Content Items in the current State to the Role. Options are *Y* (default; Content in this State will be displayed in the Inbox of users in this Role) and *N* (Content in this State will not be displayed in the Inbox of users in this Role). This option is used primarily to reduce the volume of content displayed in the user's Inbox. For example, Authors should have access to content in the Archive State so they can revive it if necessary, so the Author Role should be assigned to the Archive State. However, you do not want members of the Author Role to see all archived content because their Inbox would become useless. Therefore, in the assignment of the Author Role to the Archive State, the value of **Show in Inbox** is *N*. Archived content will thus not appear in the Inbox of users in the Author Role, but they will still be able to search for the content and can act on it when the find it.

Ad Hoc Assignment

When a user Transitions a Content Item from one State to another, the item is typically available to all members of the Role or Roles assigned to that State. In some cases, however, you may want to give users the ability to assign the content item to specific individuals. In other cases, you may want to allow users to assign Content Items to individuals that are not in a Role assigned to the State to which the item is Transitioning. Ad hoc assignment provides the flexibility to make these assignments.

To make ad hoc assignment available, on the Edit Assigned Role page assigning the Role to the State, choose *Enabled* or *Anonymous* in the **Ad-hoc** drop list.

If you enable ad hoc assignment, when a user Transitions a Content Item into the State, Rhythmyx will pop up a dialog allowing the user to make an ad hoc assignment. Users can then assign the Content Item to specific individuals. If you choose *Enabled* in the **Ad-hoc** drop list, only users in the Role assigned to the State are eligible to be assigned the Content Item. If you choose *Anonymous* in the **Ad-hoc** drop list, any user in the system is eligible to be assigned the Content Item.

Assigning a Role to a State

To assign a Role to a State:

1. Access the Edit State page.
2. Click **New Assigned Role**.

   Rhythmyx will display the New Assigned Role page.
3. Select the **Role** you want to assign to the State. Options include all Roles defined for the workflow.
4. Select an **Assignment**. Options are *None* (State is not assigned to anyone), *Reader* (Role has read-only access to content items in the State), and *Assignee* (Role has full access to content items in the State).
5. Select an option for **Ad-hoc** assignment. Options are *Disabled* (default; ad hoc assignment is not permitted to the State), *Enabled* (ad hoc assignment is allowed to the State), and *Anonymous* (only Anonymous ACL access is permitted to the State.
6. Select a **Notify** option. Options are *Y* (default; Notification is allowed) and *N* (Notification is not allowed).

7 Select an option for **Show in Inbox**. Options are *Y* (default; Content Items in this State will appear in the Inbox of users in this Role) and N (Content Items in this State will not appear in the Inbox of users in this Role). In most cases, you should only choose *N* for when assigning a Role to a State that will have a very large volume of Content Items, such as an Archive State.

8 Click [**Update**] to add the Role to the State.

Editing a Role Assigned to a State

To edit a Role Assigned to a State:

1 Access the Edit State page for the State whose Role assignment you want to edit.

2 Click the name of the Role whose assignment you want to edit.

Rhythmyx displays the Role Assignment page.

3 You can change the data in all fields.

4 When you complete your edits, click [**Update**].

Rhythmyx updates the Role assignment and returns you to the Edit State page.

Deleting a Role from a State

To delete a Role from a State:

1 Access Edit State page of the State from which you want to delete the Role.

2 Click the Delete button ✕ in the row of the Role you want to delete.

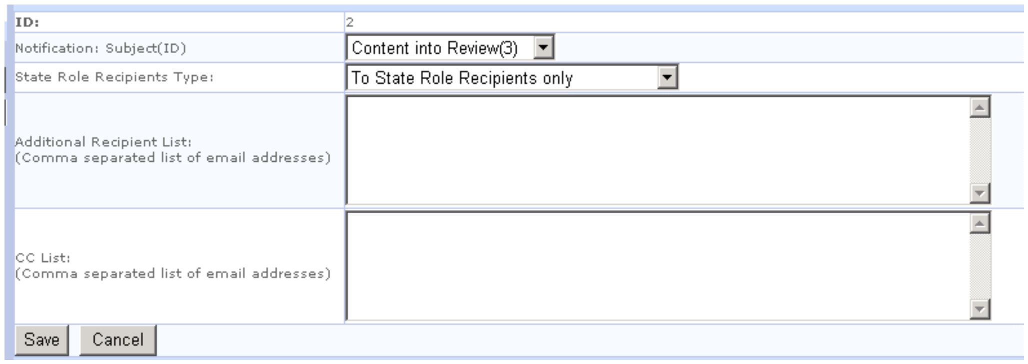
Rhythmyx displays a confirmation message.

3 To confirm the delete action, click [**OK**]. To abort the delete action, click [**Cancel**].

Notification Assignment

Transition Notification Page

Use the Transition Notification page to assign Notifications to Transitions.



ID:	2
Notification: Subject(ID)	Content into Review(3)
State Role Recipients Type:	To State Role Recipients only
Additional Recipient List: (Comma separated list of email addresses)	
CC List: (Comma separated list of email addresses)	

Figure 192: Transition Notification Page

Field Descriptions

ID Display Only. The identifier associated with this Notification.

Notification: Subject Mandatory. The subject line of the e-mail sent as the Notification.

State Role Recipients Mandatory. Defines the users that will receive the Notification e-mail message. Options are *To State Role Recipients Only* (only Members of the To State will receive messages), *From State Role Recipients Only* (only Members of the From State will receive messages), *Both To and From State Role Recipients* (Members of both the From State and the To State will receive messages), or *No State Role Recipients* (messages will not be sent to any Role Members; addresses listed in the Additional Recipients and CC lists will still receive messages).

Additional Recipient List Optional. A list of additional e-mail addresses to send the Notification to.

CC List Optional. A list of e-mail addresses that will receive a copy of Notification messages.

Assigning a Notification to a Transition

To assign the Notification to a Transition:

1. In Rhythmyx Content Explorer, click the **Workflow** tab.

   Rhythmyx displays the Workflow tab.
2. Click the Preview icon for the Workflow to which you want to add the Transition.

   Rhythmyx displays the Preview of that Workflow.
3. Click the Transition to which you want to add a Notification.

   Rhythmyx displays the Edit Transition page for the Transition you selected.
4. Click **New Transition Notification**.

   Rhythmyx displays Transition Notification page.
5. Select the **Notification** you want to assign to the Transition. Notifications are identified by Subject and unique identifier.
6. Select the State Role Recipients Type. Options include:
   - *To State Role Recipients Only*: Sends Notification e-mail messages only to Members of the Roles assigned to the State to which the content item is Transitioning.
   - *From State Role Recipients Only*: Sends Notification e-mail messages only to Members of the Roles assigned to the State from which the content item is Transitioning.
   - *Both To and From State Role Recipients*: Sends Notification e-mail messages to Members of the Roles assigned to the State from which and to which the content item is Transitioning.
   - *No State Role Recipients*: Notification e-mails will not be sent based on Roles. E-mail messages will still be sent members of the Additional Recipient and CC lists.
7. Enter the e-mail address of any additional recipients of the Notifications in the **Additional Recipient List** field or **CC** field. Separate the addresses with comma.
8. Click [**Save**] to save the Transition Notification.

Rhythmyx returns you to the Edit Transition page for the Transition to which you added the Notification.

Editing a Notification Assignment

To edit a Notification Assignment:

1. In the Rhythmyx Content Explorer, click the **Workflow** tab.

   Rhythmyx displays the Workflow Editor.
2. Click the Preview icon for the Workflow to which you want to add the Transition.

   Rhythmyx displays the Preview of that Workflow.
3. Click the Transition whose Notification you want to edit.

   Rhythmyx displays the Edit Transition page for the Transition you selected.
4. Click the name of the Notification Assignment you want to edit.

   Rhythmyx displays the Transition Notification page.
5. You can change the **Notification** assigned to the Transition, change the **State Role Recipients Type**, or add e-mail addresses to the **Additional Recipient List** or **CC** List, or delete e-mail addresses from those lists.
6. Click [**Save**] to save your changes.

Rhythmyx returns you to the Edit Transition page for the Transition whose Notification assignment you edited.

Deleting a Notification from a Transition

To delete a Notification from a Transition:

1. In the the Rhythmyx Content Explorer, click the **Workflow** tab.

   Rhythmyx displays the Workflow Editor.
2. Click the Preview icon for the Workflow to which you want to add the Transition.

   Rhythmyx displays the Preview of that Workflow.
3. Click the Transition to which you want to add a Notification.

   Rhythmyx displays the Edit Transition page for the Transition you selected.
4. Click the delete icon ✕ of the Notification Assignment you want to delete.

Rhythmyx deletes the Notification and returns you to the Edit Transition page.

Implementing Quick Edit

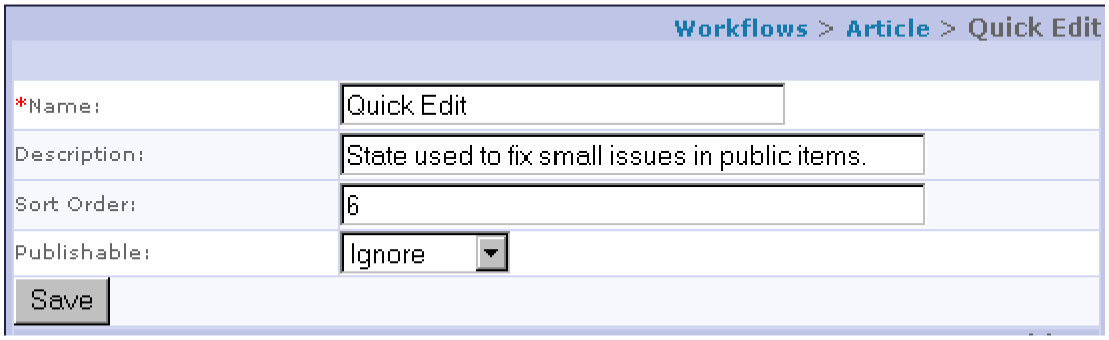
Rhythmyx does not allow users to check out Content Items in a Public State. Therefore, a user must Transition an item from the Public State to an editable State, then check it out, in order to edit the item. This process can be cumbersome if the user only needs to correct a minor misspelling.

To make the process easier for the user, For any Content Item in a Public State, Rhythmyx includes the menu option *Edit > Quick Edit*. This menu option provides a single action that Transitions the Content Item from Public and checks it out to the user. It requires some special implementation in each Workflow, however.

To implement Quick Edit in a Workflow:

1 The Workflow must include a Quick Edit State.

To make a State a Quick Edit State, set the Publishable parameter to *i* (Ignore). Any State with this value in the Publishable parameter is a Quick Edit State. A Workflow can include any number of Quick Edit States.

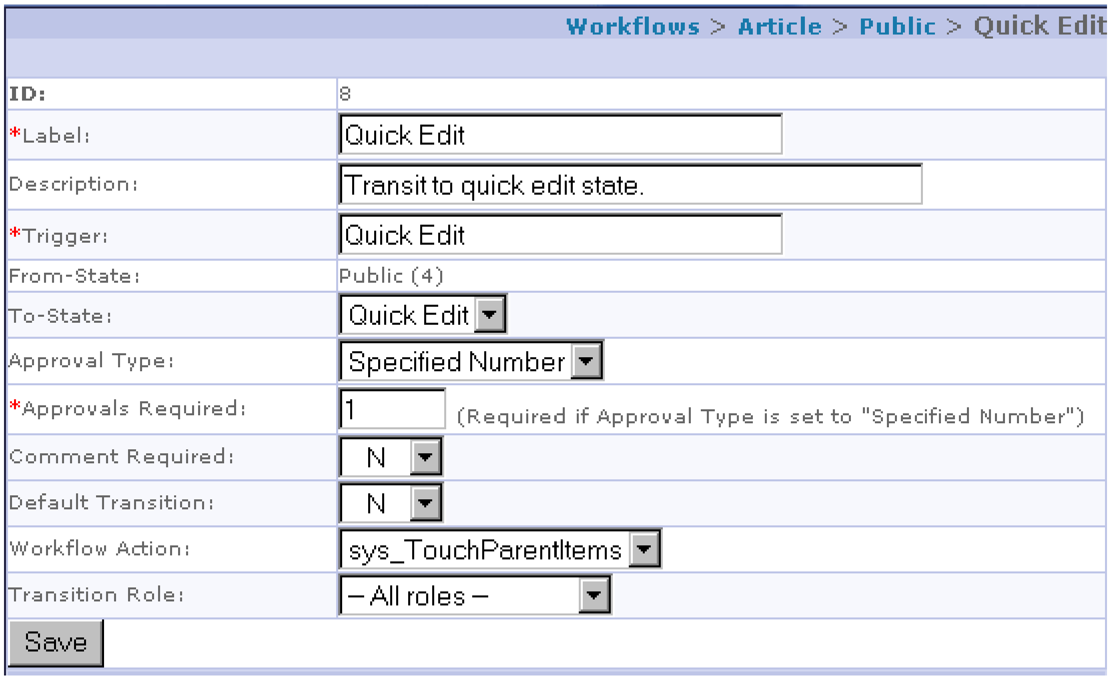


*Name:	Quick Edit
Description:	State used to fix small issues in public items.
Sort Order:	6
Publishable:	Ignore

Figure 193: Quick Edit State Definition in Default Article Workflow

2 Define a Quick Edit Transition from the Public State to the Quick Edit State.

To make a Transition a Quick Edit Transition, the value in the **Trigger** field must be *Quick Edit*. The server requires this value to activate the Quick Edit processing.



Workflows > Article > Public > Quick Edit

ID:	8
*Label:	Quick Edit
Description:	Transit to quick edit state.
*Trigger:	Quick Edit
From-State:	Public (4)
To-State:	Quick Edit
Approval Type:	Specified Number
*Approvals Required:	1 (Required if Approval Type is set to "Specified Number")
Comment Required:	N
Default Transition:	N
Workflow Action:	sys_TouchParentItems
Transition Role:	-- All roles --

Figure 194: Quick Edit Transition from the Default Article Workflow

Note that each Public State can have only one Quick Edit Transition because the value of the **Trigger** field for each Transition in the State must be unique. If multiple Transitions share the same Trigger, Rhythmyx cannot determine which to execute.

A Workflow can include multiple Public States, however, and each State may have its own Quick Edit Transition, because Transitions in different States can have the same Trigger.

3 Define a Transition from the Quick Edit State back to the Public State. This Transition should have the sys_TouchParentItems in the **Workflow Actions** field of the Edit Transition Page to ensure that all related content items are updated with the changes made during Quick Edit.

The following graphic shows an portion of a State diagram of a Workflow in which Quick Edit has been implemented. Note the Quick Edit State, the Quick Edit Transition to the State, and the Publish Transition back to Public.

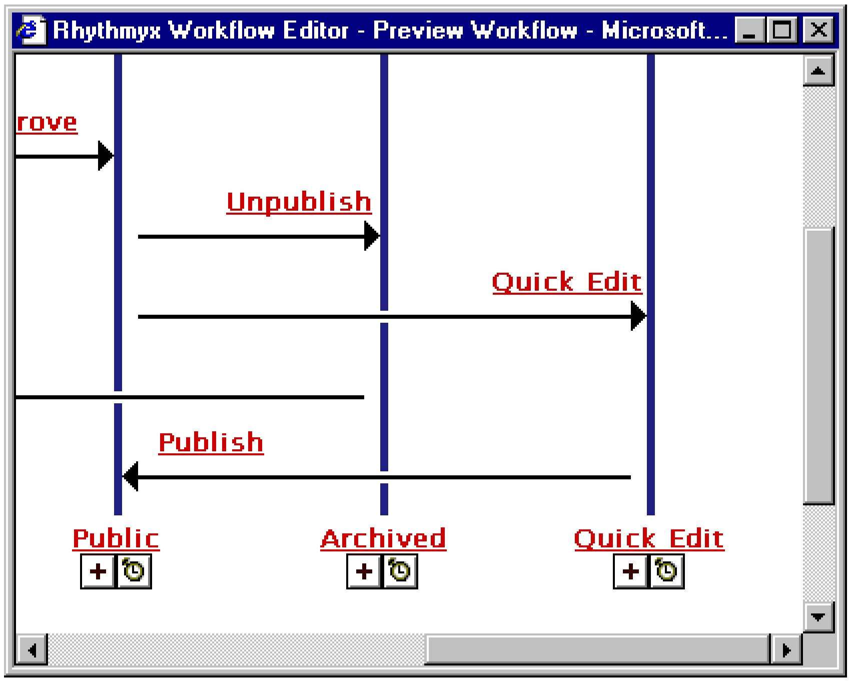


Figure 195: Workflow Diagram Showing Quick Edit

NOTE: If a Content Item is in a Quick Edit State, the default Content List application (rx_PubContentLists) directs the Rhythmyx Publisher to publish the Content Item's last public revision. If a Content Item is linked to a Related Content Item that is currently in a Quick Edit State, rx_PubContentLists directs the Rhythmyx Publisher to publish the link to the last public revision of the Related Content Item.

When you create a Content List application, you must be sure to direct the Publisher to publish the correct revisions of content in Quick Edit States. For instructions, see "Mapping a Content List Resource" in the *Rhythmyx Workbench Online Help*.

Index

A

About the Workflow Administrator • 226
About Workflows in Rhythmyx • 221
Ad Hoc Assignment • 261, 262
Add/Edit Role Dialog • 232, 233, 234
Adding a new Folder within an Existing Site • 168
Adding a New Role • 232, 233
Adding Additional WebDAV Servlets in Rhythmyx's Installation of Tomcat • 137, 141
Adding Content to a Site Folder • 177
Adding Existing Members to a Role • 235, 236
Adding Modified Stylesheets to the Navon Assembly Application • 209
Adding Navigation Elements to Page Variants • 199
Adding Navigation to Rhythmyx Site Folders • 172
Adding New Fields to a Content Editor • 44
Adding New Fields to an Assembly Resource • 47
Adding New Location Schemes to Publish Contexts • 214
Adding New Members to a Role • 238, 239
Adding the sys_WebImageFX Control to a Content Editor • 131
Agent • See Aging
Aging • 250, 251, 252, 253, 254
Aging Prerequisites • 251
Aging Scenarios • 252, 253
Assembling Binary Outputs • 65
Assigning a Content Item to Site section • 177
Assigning a Landing Page to a Navon • 173
Assigning a NavImage to a Navon • 174
Assigning a Notification to a Transition • 264
Assigning a Role to a State • 262
Assigning Permissions to a Site Section • 168
Assignment • 261
  Notifications to Transitions • 263, 264, 265
  Roles To States • 261, 262, 263
Automated Indexes • 83
Automatic Transition • 254
Automatically Populating a Slot using AutoRelatedContent • 95

B

Best Practices
  sys_WebImageFX • 135
Bio Content Type • 34

C

Catalogging • 235, 236
Comments • 258
Contacts Content Editor • 36
Contacts Content Type • 36
Content Editor • 35, 39
Content Type Definitions • 33
Continuous Conversion Example • 117, 125
Copying a Site • 163
Copying a Site Sub-Section • 164
Copying a Workflow • 222
Corporate Investments Development Plan • 33
Creating a Content Assembler Resource • 66
Creating a Content Editor that Extracts Text • 105
Creating a NavImage • 176
Creating a Navon • 172
Creating a NavTree • 172
Creating a New Content Assembler • 63
Creating a New Content Editor • 53
Creating a New Role in Rhythmyx Workflow • 225, 230
Creating a New State • 225, 244
Creating a New Transition • 249
Creating a Non-Text Assembler • 113
Creating a Notification • 225, 253, 258
Creating a Site Folder for a New Site • 165
Creating a Site Folder in an Existing Site Tree • 166
Creating an Aging Transition • 252
Creating an Automated Index Query Assembler • 84
Creating Custom Navigation Variants • 194
Creating Page Templates • 68
Creating Site Structures in Rhythmyx • 161
Creating the Bio Assembly Application • 111
Creating the Bio Content Editor • 109
Creating the Common Stylesheet • 206
Creating the Include Stylesheet • 208
Creating the Location Scheme • 216
Creating the Server-side Include Output Location Resource • 214

Creating the SSI Stylehseet • 208
Creating Variant Templates • 67
Customizing FastForward Applications • 9, 43
Customizing Global HTML Templates • 81
Customizing Navigation CSS • 188
Customizing Navigation Look and Feel • 188

D

Defining a Global Template for a Site Section • 187
Defining Conditions for the Common Stylesheet • 213
Defining Conditions for the Include Stylesheet • 211
Defning Conditions for the Server-side Include Stylesheet • 212
Deleting a Global HTML Template • 82
Deleting a Member from a Role • 240
Deleting a Notification • 225, 258
Deleting a Notification from a Transition • 265
Deleting a Role • 232, 235
Deleting a Role from a State • 263
Deleting a Role in Workflow • 225, 230
Deleting a Site Folder Section • 171
Deleting a State • 225, 245
Deleting a Transition • 250
Deleting a Workflow • 225
Displaying Extracted Text in a Content Editor • 107, 109

E

Edit Assigned Role Page • 261
Edit Notification Page • 257
Edit Role Page in Workflow • 229
Edit State Page • 243, 246
Edit Transition Page • 227, 247, 249
Edit Workflow Page • 224
Editing a Member's Properties • 240, 251
Editing a Navigation Item • 181
Editing a Notification • 225, 258
Editing a Notification Assignment • 265
Editing a Public Navigational Element • 182
Editing a Role • 234
Editing a Role Assigned to a State • 263
Editing a Role in Workflow • 225, 230
Editing a Site Section • 170
Editing a State • 225, 245
Editing a Transition • 249
Editing a WebDAV Configuration File • 138, 142
Editing a Workflow • 225
Editing an Item in a Site Section • 181
Elements of the Assembly Process • 66
Escalating Notices • 254
Extending FastForward Applications • 51
Extending Publishable States • 244, 245

G

Global Templates • 25, 34

I

Implementing a New Content Type • 52
Implementing Managed Navigation as Server-side Includes • 203
Implementing Quick Edit • 267
Implementing Text Extraction and Non-Text Assembly • 104
Implementing WebDAV in Rhythmyx's Installation of Tomcat • 136
Including User Comments in Notification E-mail Messages • 258
Initial Implementation Tasks • 8
Initial State • 221
Introduction • 5

J

Job Listing • 38

L

Locating an Item with Impact Analysis • 178
Locating an Item within a Site Tree • 179

M

Maintaining a Site • 177
Maintaining Global Templates • 80
Maintaining Server Roles • 230, 231
Maintaining Transition Roles • 249, 250
Maintaining Workflows • 223
Mandatory Relationships and Workflows • 227
Member Properties • See Role and Member Properties
Merging Navigation Items • 185
Migration Example • 125
Modeling and Design • 15
Modeling the CMS • 16
Modelling Content Assemblers • 22
Modelling Content Editors • 21
Modelling Publishing • 30
Modelling Workflow • 27
Modify Member List for • 235, 238

Modifying the Conditions on the Existing Navigation Stylesheet • 210
Modifying the Main Navigation Bar Stylesheet • 207
Modifying the Rhythmyx Slots Stylesheet to Process Server-side Includes • 209, 214
Modifying Variant Templates • 49
Modifying XSL Stylesheets for Server-side Includes • 206

N

Navigation Communities • 158
Navigation Reset • 159
New Member Dialog • 238
Notification • 257, 258
  Assigning to Transition • 263, 264, 265
  Including Comments • 258
Notification Assignment • 249, 253, 263
Notifications • 224, 257

O

Overview • 7

P

Page Markup Summary • 100
Prerequisites • 251
Properties • See Role and Member Properties

R

Recursive Content Roll-up • 65
Registering a new Navigation Menu Slot • 200
Registering New Page Variants for the Server-side Include Files • 203
Removing a Content Item from a Site Section • 181
Removing a Landing Page from a Navon • 173
Removing a NavImage from a Navon • 174
Reordering a Submenu • 186
Repeating Notices • 253
Role and Member Properties • 240
Role Assignment • 261
Roles • 221, 229, 230, 231
  Assigning to States • 261, 262, 263
  Server Roles • 231, 232, 233, 234, 235, 236, 238, 239, 240
rxserver Member • 251

S

Scope • 33
Secondary Implementation Tasks • 10
Setting Aging Agent Parameters • 251
Setting the Workflow Configuration Properties for Notification • 257
Setting up a Keyword for Automated Indexes • 84
Setting up a Site • 161
Shared Variants • 101
Simple Example Implementation of WebDAV • 136, 144
Sites and Managed Navigation • 155
Sites and Navigation • 12
Snippet Templates • 75
Specialized Implementations • 11, 103
Splitting Navigation Sections • 182
States • 221, 224, 225, 226, 243, 244, 245
  Assigning Roles to • 261, 262, 263
Summary • 41
sys_WebImageFX and the WebImageFX Editor • 127
sys_WebImageFX Control • 129

T

Testing the Bio Content Type • 115
Testing the Event Auto Index • 94
Text Extraction • 105
The Assembly Process • 64
Transition • 221, 247, 249, 250
  Aging • See Aging
  Assigning Notifications to • 263, 264, 265
Transition Notification Page • 263
Transitions • 247

U

Updating Global HTML Templates • 81
Uploading External Binary Files into Rhythmyx • 105
Using Managed Navigation • 158
Using Navigation Themes • 201
Using Variable Selectors • 191

V

Validating the Manged Navigation Using Server-side Includes Implementation • 218
Variants • 36, 38, 40
Viewing a Workflow • 226

W

WebImageFX • 127
Workflow • 221, 223, 224, 225, 226

Defined • 221
Workflow Editor • 223
Workflow Roles • 221, 224, 225, 229
Workflows in Rhythmyx • 13