



Kurant StoreSense Designer's Reference

Version 5.7.0

StoreSense™ KURANT™

© 2004 Kurant Corporation.
Kurant, StoreSense, and the
Kurant logo are trademarks of
Kurant. All other products
mentioned are trademarks of
their respective companies.



Table of Contents

Chapter 1: Introduction.....	4
The Importance of Store Design	4
Additional Documents	4
Chapter 2: Understanding Store Design.....	5
How Data is Displayed.....	5
Understanding Templates	7
Understanding SSML Tags.....	8
Understanding Business Objects	9
Chapter 3: Wizards and Assistants.....	10
Setup Wizard	10
Design	11
Settings.....	12
Build Store	13
Using the Template Assistant.....	13
Modifying a Store's Look and Feel	14
Chapter 4: Integrating an Existing Site with StoreSense	16
Transferring the Look and Feel of Your Existing Site to Your Storefront	16
Choosing a Theme	16
Customizing a Theme.....	16
Copying Content From Another Web Site.....	18
Linking to and from Your Storefront.....	19
Using Link Builder	19
Connecting to Other Sites.....	20
Chapter 5: Working with StoreSense Design Studio.....	21
Working in StoreSense Design Studio	21
Managing Your Work	22
Setting Baselines.....	23
Resetting Templates.....	24
Previewing Your Work.....	24
Publishing Templates.....	24
Common Tasks	25
Selecting Content	25
Adding Text	28
Adding Links.....	28
Adding Images.....	29
Displaying Store Data.....	30
Building Lists.....	31
Using Conditions.....	32
Saving and Undoing Your Work	34
Advanced Tasks.....	34
Template Validation.....	34
Template Caching	36
Exporting Templates	36
Importing and Adding Templates	37
Changing Template Properties	38
Using Template Edit Modes	39

Working in Design Mode	41
Working with Template Source Code	43
Chapter 6: Using Template Sets	45
Understanding Template Sets.....	45
Managing Template Sets	45
Setting the Template Set Language	47
Applying and Defining Template Set Rules	47
Chapter 7: Advanced Design Concepts	50
Working with Images	50
Uploading Images.....	50
Using and Updating Store Images.....	51
Using Product (Catalog) Images	52
Using an Image in Place of the "Submit" Button	52
Customizing Tag Images	52
Working with Files	53
Uploading Files	53
Creating a Splash Page	54
Cascading Style Sheets	55
JavaScript	55
Multimedia Files	55
Managing Files.....	55
Omitting the Header and Footer	56
Customizing the Display of the Category List	56
Using Multiple Templates	57
Where You Can Use Multiple Templates	57
How to Create and Use Multiple Templates	58
User Defined Templates.....	58
Design FAQ	59
Chapter 8: Code Snippets.....	62
Adding Multiple Products to a Cart	62
Quick Search Form	62
UPS Tracking Link.....	63
Collapsible Category Lists	63
Product Price Formatting	63
Hide Attribute Information When Not In Stock.....	64
Order Extension Information on the Invoice Email	65
Suppressing A Shipping Method	65
Subcategory "Drill-Down" on Catalog List	65
Comparison Category List	67
Display Related Products on Cart Template	68
Show Only Attributes That Are On Sale.....	68
Show Only Attributes That Are On Sale.....	69
Chapter 9: Testing Your Design	70
Equipment and Software to Test	70
How to Test.....	70
Testing Store Information Templates.....	70
Testing the Product Catalog Templates.....	71
Testing the Checkout Templates.....	71
Testing Membership and Subscription Templates	71
Testing Customer Service Templates	72
Troubleshooting.....	72
Troubleshooting Validation Errors.....	72

Troubleshooting Images	72
Troubleshooting Pages	73
Troubleshooting Search Results	74
Appendix A: Glossary	75
Appendix B: Template Reference	78
Appendix C: Tag Reference	98
Appendix D: Tag Mappings	124
Index	161



Chapter 1: Introduction

The Designer's Reference is designed to provide you with all the information you need to create the perfect look and feel for your store. This guide covers all phases of the design process, from using the automated wizards, to advanced custom store design using JavaScript and Cascading Style Sheets. In addition, this guide contains detailed StoreSense Tag and Template reference.

The Importance of Store Design

The design of your store is critical when customers are deciding whether or not to purchase products from your store. The design of your store affects:

- ◆ How customers perceive your credibility. A design that looks unprofessional or outdated may cause a customer to look elsewhere for products.
- ◆ Whether or not a customer will complete a purchase. A design that is hard to use may cause a customer to cancel their purchase.
- ◆ A customer's willingness to trust you with payment information. Again, an unprofessional design may cause a customer to think you don't take security concerns seriously.
- ◆ Whether or not a customer will return to your store. If the design makes it difficult for a customer to find what they are looking for or check out, they may be less likely to return to your store.

The themes and layouts developed by StoreSense take all these factors into consideration. Before making modifications, or developing your own design, we recommend that you become familiar with web design principles before continuing.

Additional Documents

Please see the following documents for additional information:

- ◆ The Online Guide available in the StoreSense application.
- ◆ StoreSense Store Administration Users Guide: this may be downloaded from the Online Guide start page.
- ◆ StoreSense SSML Object Reference: this may be downloaded from the Online Guide start page.



Chapter 2: Understanding Store Design

Starting with the storefront through order confirmation, templates define your store's look and feel. One template defines your store's navigation, another defines how your product catalog is displayed, and another defines how the e-mails your store sends will appear.

StoreSense is shipped with a default set containing dozens of templates that cover every phase of the shopping process. However, you can modify these templates to provide your own custom look and feel.

In addition, StoreSense supports the uses of additional pages and files (like non-store generated HTML pages, JavaScript and CSS) that allow for unlimited possibilities when developing the design of your store.

This reference will provide you with the information you need to create the perfect store design.

How Data is Displayed

Usually, web pages are made up of just a few components, such as HTML Code, text content, and images. More advanced pages might also have a style sheet, JavaScript and other multimedia. Regardless of the specific elements, once you have created a page, it will always appear the same every time it is viewed. These are considered "static" pages.

StoreSense, however, combines standard HTML elements with StoreSense Markup Language tags (SSML) and "Business Objects." These tags and objects display data from your store's database and generate pages based on your catalog, customer information, search information, store settings, etc. These are considered "dynamic" pages.

For example, if you were using static pages to create your store and wanted to create a table that would display a list of your products, it might look something like this:

```
<table>
  <tr>
    <td>Sleeping Bag</td>
    <td>$300.00</td>
  </tr>
  <tr>
    <td>Tent</td>
    <td>$150.00</td>
  </tr>
  <tr>
    <td>Outdoor Cooking Set</td>
    <td>$750.00</td>
  </tr>
</table>
```

This would display the product information for the three products listed on this page. Once you've created this page, the only way to update it for your new products, or to change pricing information, is to open the page and add a new row for each product.

Using a StoreSense template however, you would still code a table, but instead of having one row for each product, you would create code that defines how you want a product row to appear. You would then use SSML tags and objects in place of the actual product information. The next image shows an example of a template - SSML tags begin with <ss:.

```

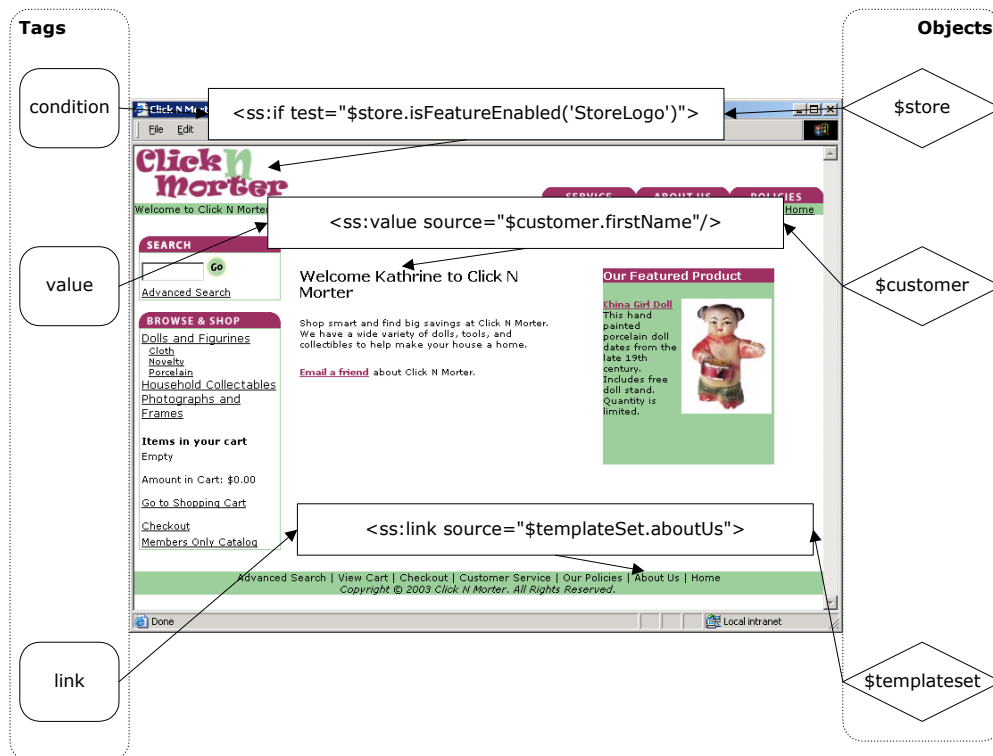
<table>
  <ss:foreach item="listing" within="$catalog.results">
    <ss:foreach item="product" within="$listing.products">
      <ss:sortby item="product" />
      <tr>
        <td><ss:value source="$product.name" /></td>
        <td><ss:value source="$product.price" /></td>
      </tr>
    </ss:foreach>
  </ss:foreach>
</table>

```

- ◆ The `<ss:foreach>` and `</ss:foreach>` tags tell the page to repeat the information between those tags for each product.
- ◆ The `<ss:value source="$product.name" />` tag and object displays the product name.
- ◆ The `<ss:value source="$product.price" />` tag and object displays the price of the product, formatted as currency.

Now, whenever you add a new product, that product will appear on this page. Or, if you change a product's price, that new price will appear the next time the page is viewed.

StoreSense templates also work with your store's information when generating pages. For example, if you set the maximum search results per page to 10, then "next" and "back" links are automatically added if your customer's search results generate more than 10 products.

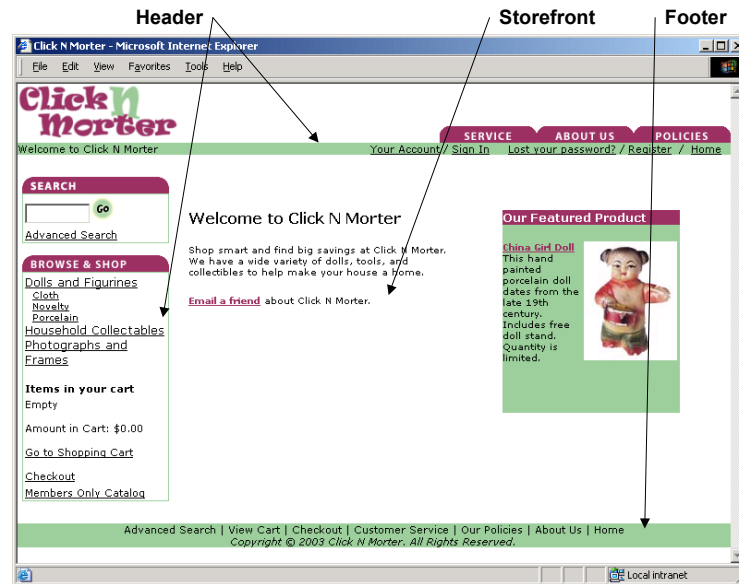


How store information is used to build templates

Understanding Templates

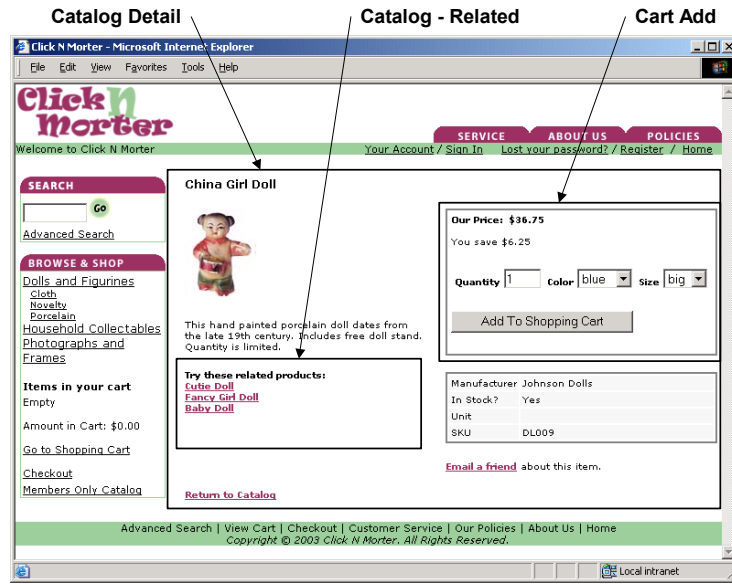
Unlike standard HTML pages, StoreSense template pages are “dynamic”, meaning they are built as the customer requests them. In addition, what appears to be a single page when viewed in the browser is actually at least three files put together into a page - the header template, footer template, and then a body template.

This is done for two reasons: to simplify the design of your store, and to offer maximum customization of your store. For example, by changing the navigation on your header template, you have changed the navigation for your entire store.



Templates used to build your store's opening page.

However, some pages are built using even more templates. For example, the Catalog Detail template can contain both the Related Products and Cart Add templates.



Templates required to build the Catalog Detail page.

To prevent the Header and Footer templates from displaying on a template, simply add a `<HTML>` tag as the first item on the appropriate template, and a `</HTML>` template as the last item. This allows you to achieve custom designs for individual templates (such as giving the home page a different look than the rest of your templates).

Understanding SSML Tags

StoreSense Markup Language (SSML) Tags are the equivalent of a markup language such as Hypertext Markup Language (HTML) or Standard Generalized Markup Language (SGML), and define the type of information that will be presented (for example, a link is created using the `<ss:link/>` tag). Tags can:

- ◆ Display information from the store (`<ss:value/>`, `<ss:image/>`).
- ◆ Create form fields (`<ss:edit/>`, `<ss:select/>`, `<ss:password/>`, `<ss:radio/>`).
- ◆ Apply formatting (`<ss:font/>`).
- ◆ Create links (`<ss:link/>`).
- ◆ Display information based on test conditions (`<ss:choose/>`, `<ss:if/>`).
- ◆ Create and sort lists (`<ss:foreach/>`, `<ss:sortby/>`).

In addition, SSML includes more advanced tags that can assist in the creation of complex store functionality and design.

The following are examples of how SSML tags are constructed:

```
<ss:edit source="$customer.FirstName" size="20"/>
```

```
<ss:link source="$templateSet.aboutUs">About Us</ss:link>
```

- 1: <ss:/> defines this as a StoreSense tag.
- 2: The tag type. Generally indicates what this tag will resolve to on the storefront.
- 3: The Object Expression. Defines where the data is coming from, or the data to which input will be posted. Object expressions always start with the "\$" character.
- 4: Tag attributes that further define the tag.
- 5: The content to appear between the opening and closing tags. Not all tags have content.
- 6: The closing slash or tag.

Understanding Business Objects

StoreSense Business Objects provide direct access to the information in a your database, and combined with HTML or the StoreSense tags, display or collect information in the storefront. For example, <ss:value source="\$customer.firstName" /> displays the first name of the customer (the <ss:value/> tag displays a value, and the \$customer.firstName object expression tells the tag what item to display).

Business Objects provide access to information such as:

- ◆ Product Information: \$product.description, \$product.name, \$product.thumbnailImageName.
- ◆ Search Options: \$catalog.categoryList, \$catalog.keyword.
- ◆ System Information: \$system.daysTill('Christmas') .
- ◆ URL Paths: \$templateSet.images['filename.gif'], \$templateSet.AboutUs.



Chapter 3: Wizards and Assistants

While StoreSense offers the capability to develop unlimited store designs, it also offers a set of preset layouts from which you can choose to help you get started. In addition, if you wish to change the look and feel of your store at a later time, it offers a Template Assistant to help with that process.

Setup Wizard

The Setup Wizard walks you through the basics of configuring your new store. The entire process should take between 15 and 30 minutes. A few notes about the Setup Wizard:

- ◆ The Setup Wizard is intended for first time setup. To make global design changes after the initial setup, it is recommended that you use the Template Assistant in the Design Studio.
- ◆ The Setup Wizard will only affect your *default* template set - if you wish to work with multiple template sets, please use the Design Studio.
- ◆ Running the Setup Wizard will overwrite any edits you have already made to your default template set.
- ◆ If you have customized any images that are called by StoreSense tags (such as buyit.gif or join.gif), the Setup Wizard will overwrite those images. Please back up your custom images before continuing.

General navigation guidelines:

- ◆ Click **Cancel** to save your changes and return to the Store Administration home page.
- ◆ Click **Next** to save your settings and move to the next step in the process.
- ◆ Click **Back** to move back one page.
- ◆ When browsing themes and layouts, click **Previous** to view the prior list of options, and click **More** to view the next page of options.



Navigating the Setup Wizard

StoreSense provides a checklist to help you prepare for running the Setup Wizard. This checklist can be viewed by clicking the link on the Welcome page, or on the first page of the Setup Wizard.

Design

This section allows you to choose a layout scheme for your design. You can use the design as is or modify it later using the StoreSense Design Studio to meet your needs.

Page	Description
Theme Selections	This is where you set the overall look and feel for your site. Small images of each theme are presented on this page - for a larger view, click the image. To view additional themes, click More .
Select Pages	Select the pages you wish to include in your store. You can either include a Home Page in your store, or you can link to an existing site's home page by entering its URL.
Page Layouts	The page layouts set the body layout for three key pages in your store. ◆ Home Page: this is the first page a customer sees when they visit your store. ◆ Catalog List: this is the search results page that is displayed after searching for a product in your store, or when clicking a link that you've created to display a particular set of search results. ◆ Product Detail: this is the page a customer sees when they're reviewing detailed information regarding a product. To help you choose your layouts, your theme selection will appear to the left. In

Page	Description
	addition, if the designer of the theme included notes and suggestions regarding the theme, those notes will appear below the theme image. To view additional page layouts, click More.
Add Store Logo	This page allows you to add your business's logo to the store. If you have your store logo ready, click Browse to browse to the image file. The image must be either a *.gif or *.jpg image. If you do not want to add your store logo, select the I won't be using a store logo check box. You can add a logo later by using the Upload Images page in the Support Manager area.
Fonts and Colors	This page allows you to choose fonts and colors for three categories of fonts: ◆ Heading is the text that appears as the page title. ◆ Sub Heading is any sub heading fonts (usually used as category sub headings, or just a smaller heading font on a page). ◆ Paragraph is the standard body text. This page also displays the recommended font faces, sizes and colors, and displays them against the background color in which they will appear.
Review	This page allows you to review the design choices you have made. To change a selection, click the element you want to change. To confirm selections, click Continue.

Settings

This section describes the Settings portion of the Setup Wizard.

Page	Description
Store Preferences	In this section, you provide basic information about your store including its name, your contact information, and the payment methods your store accepts. See <i>Field Help: Setup Wizard Store Preferences</i> (page 12) for more information.
Custom Text	Enter the text that you want customers to see on the pages you selected during the first step in the Setup Wizard.
Add Products	This section allows you to enter the minimal product information necessary to begin selling online. To add additional products or to modify your existing products, use the Product Manager. You can also import information for multiple products using the Import feature, which is recommended if you wish to add a large number of products.

Field Help: Setup Wizard Store Preferences

Store Information

Enter your store's name, business address, phone numbers, and your contact e-mail address.

Store Locale

Field	Description
Locale	Select your business' locale from the list. This will let the system know what formats to use for locale specific information, such as currency and date formats.

Field	Description
Time Zone	Select the time zone in which you wish this store to be based. This will affect how dates and times are recorded.
Current Selections	Displays your current locale selections. This information will be updated after clicking Submit .
Examples	Displays samples of how dates, currency and percentages will be displayed. This information will be updated after clicking Submit .

Payment Options

Select the payment options that your store will accept. You must select at least one payment option before you can open your store.

In order to accept credit cards you must establish a merchant account. This is handled through your bank. If you wish to process credit cards online, you will need to establish a credit card payment processor account.

Select the **Require customers to provide credit card information** check box to indicate if the customer is required to submit credit card information with their order, rather than sending it separately. If this check box is selected, the customer will not be allowed to complete the order process without entering their credit card information. However, they can complete the order if they go back and choose another payment option.

Build Store

This is the final step and officially opens your store. You have the following options:

- ◆ **Build and Open** will create the design of your store, upload your products, save your store settings, and then open your store so that it's available to the general public.
- ◆ **Build** will create the design of your store, upload your products and save your store settings, but will not open your store. This allows you more time to develop your store before making it available to the general public. You can open your store later using the **Open/Close** button on the toolbar.
- ◆ **Cancel** ends the Setup Wizard.

Using the Template Assistant

The Template Assistant assists you in redesigning your templates, and operates much like the Setup Wizard. You can further modify your templates by using Design Studio after the assistant has done the initial setup.

Running the Template Assistant will overwrite all design and text edits you have made to these templates. To make changes to your storefront without overwriting your edits, see *Modifying a Store's Look and Feel* (page 14). Also, if you have customized any images that are called by StoreSense tags (such as buyit.gif or join.gif), the Template Assistant will overwrite those images. Please back up your custom images before continuing.

To use the Template Assistant:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Under **Template Design**, click **Template Assistant**. The Template Assistant begins. See *Setup Wizard* (page 10) for more information regarding the selections.

Modifying a Store's Look and Feel

The “Look and Feel” item on the Design Studio navigation allows you to change selected portions of your storefront’s design without overwriting *all* of your edits. You can modify the store’s theme, layouts, fonts, or the locations to where each link will point.

While using the “Look and Feel” tool will not overwrite *all* edits you’ve made, it will overwrite some. Please see the instructions for each item for more information

To change the theme used by your store:

Selecting and publishing a new theme will overwrite any edits you have made to the Header and Footer templates.

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Browse to **Initial Set | Look and Feel | Themes**. The Themes Selections page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Select the theme you wish to use.
- 4 To save your selection without publishing it to the live store, click **Save**. To make your selection live, click **Publish**.

To change the layouts used by your store:

Selecting and publishing a new layout will overwrite any edits you have made to the current layout template.

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Browse to **Initial Set | Look and Feel | Layouts**. The Themes Selections page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Select the layout you wish to modify.
- 4 Select the layout you wish to use.
- 5 To save your selection without publishing it to the live store, click **Save**. To make your selection live, click **Publish**.

To change the fonts used by your store:

Selecting and publishing new fonts will *not* overwrite any edits you have made to your templates. However, changes made here will only affect font tags that use the “\$font” object (default). It will not affect any hardcoded font tags (such as).

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Browse to **Initial Set | Look and Feel | Styles**. The Fonts and Colors page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Make your new font selections.
- 4 To save your selections without publishing it to the live store, click **Save**. To make your selection live, click **Publish**.

See *Connecting to Other Sites* (page 20) for information regarding using the Site Connector.



Chapter 4: Integrating an Existing Site with StoreSense

While in many cases, StoreSense's pre-designed pages and extensive capabilities make maintaining a separate web site and web store unnecessary, you may prefer to keep your site. If so, you'll want to design your store so that it flows seamlessly with your existing site. This section provides design tips and instructions for linking your site with your store.

Transferring the Look and Feel of Your Existing Site to Your Storefront

The first step in transferring the look and feel of your existing site to your storefront is to identify those elements that make up your site's design.

- ◆ **Text Fonts and Colors:** What fonts and colors are you using for your web page text? Make a note of these, and when you run the StoreSense Setup Wizard, make the selections that match your existing site.
- ◆ **Page Layout:** How would you describe the layout of your store? Does your site's navigation run along the top or one side? If so, you'll want to modify your storefront's theme to use the same navigation.
- ◆ **Images:** While you may have many images on your page, identify those that are consistent across most pages - those images are the ones to use in your store.

Once you've identified your design elements, you're ready to transfer that design to your storefront.

Choosing a Theme

If you have not already run the complete Setup Wizard and selected a theme, you can run it again by either:

- ◆ Selecting the link on the Welcome page;
- ◆ Browsing to **Store | Setup Wizard**, or;
- ◆ Clicking the **Design Studio** button on the toolbar, selecting **InitialSet (Default)**, and then running the Template Assistant. This runs just the design portion of the Setup Wizard.

Customizing a Theme

StoreSense ships with dozens of pre-designed themes that you can use. You can browse these themes to see if there's one that matches the general design/colors of your site. If you can't find a perfect match, find one that comes the closest, and then modify the Header and Footer templates so that they match your existing site.

The Header and Footer templates are those portions of the your site design that appear on every template. Typically, the Header runs along the top and left portion of the page, and the Footer runs along the bottom, though there are some exceptions.



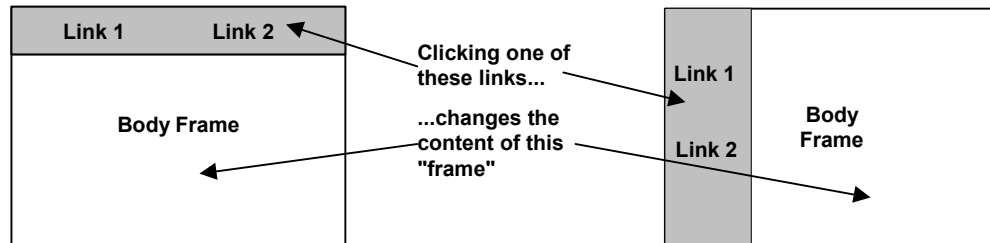
The Header and Footer Templates

To modify your Header or Footer templates:

- 1 Open Store Administration, and then open Design Studio by clicking the **Design Studio** button on the toolbar.
- 2 If you're not familiar with HTML, you'll want to work with these templates in Design Mode. To do so, browse to **InitialSet (Default) | Home Page**. Place your cursor in the Header or Footer area and make your edits.
- 3 If you want to work with the HTML code, browse to **InitialSet (Default) | Header Source** or **InitialSet (Default) | Footer Source** - the template source code appears and is ready for editing.

If Your Existing Site Uses Frames

Some sites are designed using Frames. "Framed" sites divide your pages into separate regions (for example, to contain the navigation in one frame, and the content in another). Typically in a framed site, one region stays the same all the time, but another region will change based the selected links (and usually, these pages will "scroll" independently of each other).



Two examples of framed sites. One frame is shaded, the other is white.

If you created your existing site using frames, you'll want to modify your store's theme so that it works with your existing site's frames. For example, if you have a frameset made of a top banner and body, you'll want to remove or move the content in your store's top banner section. If your frameset is made up of a left and right column, you'll want to modify the theme to move or remove the content on the left side.

In most of the themes, the top and left content of the store is contained in the Header template - modifying this template will allow you to move and remove content so that it fits with your frame design.

To modify your Header or Footer templates:

- 1 Open Store Administration, and then open Design Studio by clicking the **Design Studio** button on the toolbar.
- 2 If you're not familiar with HTML, you'll want to work with these templates in Design Mode. To do so, browse to **InitialSet (Default) | Home Page**. Place your cursor in the Header or Footer area and make your edits.

- 3 If you want to work with the HTML code, browse to **InitialSet (Default) | Header Source** or **InitialSet (Default) | Footer Source** - the template appears and is ready for editing.

Copying Content From Another Web Site

If you want to copy portions of another web site (such as a site you already maintain) to your storefront, StoreSense provides a tool to help you. The Copy Center tool allows you to view both your storefront and another web site at the same time, and then “drag and drop” pieces of the web site into your storefront.

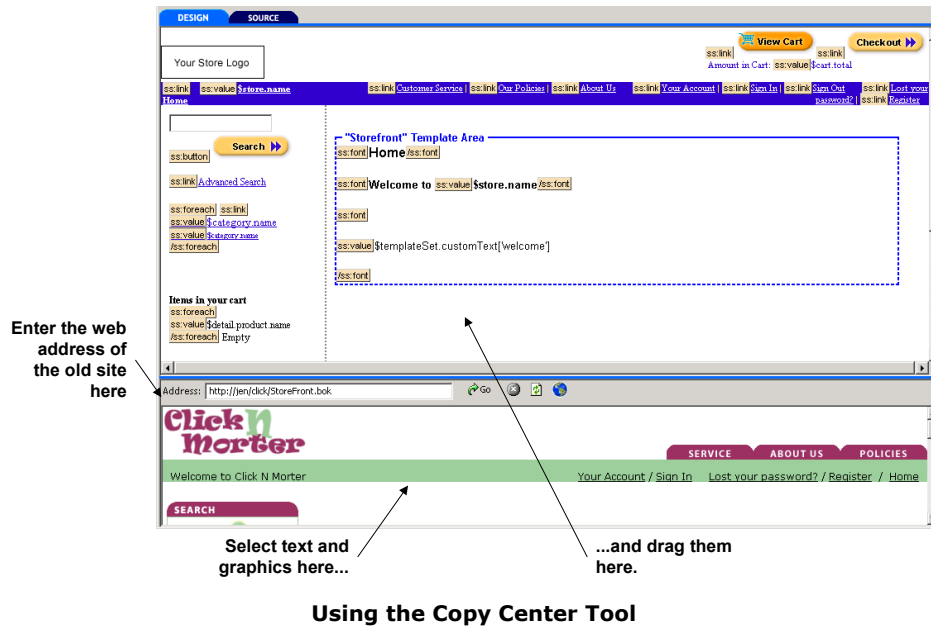
This tool is only available if you are using Internet Explorer version 5.5 or higher.

To copy content from another site to your store:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Browse to the template to which you wish to copy content. Most likely, this will be **InitialSet (Default) | Home Page**.
- 3 Click  (Copy Center). The Copy Center pane appears at the bottom of the Design Mode.

You can quickly access this tool by clicking the **Copy Center** link on the Design Studio Home page, or from the **Make my store look like my existing site** link on the Store Administration Welcome page.

- 4 In the Copy Center pane, enter the full HTTP address of the site from which you wish to copy content, and click **Go**. That site appears in the Copy Center pane.
- 5 Select the content you wish to copy in the Copy Center pane with your mouse, and then drag it to the appropriate location in the Template pane. If the content contains an image, you will have two choices:
 - ♦ **Leave the Source URL** inserts the image, but continues to point to the original source file (instead of uploading the image to your store).
 - ♦ **Copy the image** uploads the image to your store’s directory on the server, and then points to that uploaded image.
- 6 When done, click  to close the Copy Center pane.



Linking to and from Your Storefront

Once you have finalized the design of your storefront, you'll want to add links from your site to your store, and from your store back to your site.

Using Link Builder

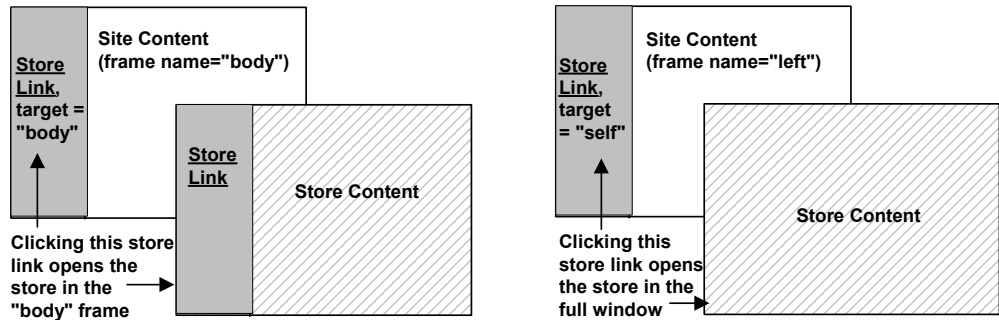
The Link Builder helps you to create the code you need to create links to your storefront from another web site (or e-mail), or add forms to a static site that add products to a cart. You can then copy this link and paste it into your external site.

Do not use this tool to create links for use *within* your storefront.

To create a link to your storefront:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Under **Global Tasks**, select **Link Builder**. The Link Builder page appears.
- 3 Select the type of link or form you wish to create from the **Type** list.
- 4 Some link types support the use of or require additional information. For example, if you've selected a **Add Product To Shopping Cart** link, you'll want to specify the product by selecting it from the drop-down list. However, a **Lost Password** link does not require any additional information.
- 5 Enter the text to appear in the link, and if necessary, enter a target for the link.
 - ♦ If your existing site uses frames and you want the link to your store to appear within your framed site (for example, the link is in the navigation frame, and you want it to open in the frame named "body"), be sure to add a "target" attribute to the link.

- ◆ If your existing site uses frames and you do *not* want your store to appear in your framed site, but instead want it to replace your site in the same browser window, use the “self” target.



Understanding Frames and Targets

- 6 Once you have created the link, you can test it by clicking the **Test Your Link** link.
- 7 When done, if you are using Internet Explorer version 5.5 or higher, click **Copy to Clipboard** to copy this link to the clipboard. If you are using another browser, select the link code and use your browser's Copy tool. You can now paste this link into an external web site or e-mail.

Connecting to Other Sites

The Site Connector allows you to point links on your StoreSense storefront to another site you may have. For example, if you have another web site that contains your home page, the “Home” link on your storefront can point to that page. This gives your visitors a seamless transition between your web site and your store.

If you have modified the header or footer templates and deleted the code that creates these links, these links will not be created. The code that creates these links appears similar to the following:

```
<ss:if test="$templateSet.templateExists('about')">
  <ss:link source="$templateSet.aboutUs">About Us</ss:link>
</ss:if>
```

To point a storefront link to another web site:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Under **Template Design**, click **Site Connector**. The Site Connector page appears.
- 4 For each page in your *other* site that you would like to add as a link to your *store*, select **Use My Other Site's Page** and enter the full URL (for example, <http://www.mysite.com/page.html>).
- 5 Click **Submit**. The links will be added to your header or footer template.

In addition, the Site Connector allows you to *add* templates to your store that you did not select when using the Setup Wizard. To do so, select the check box of the templates you wish to add and click **Submit**. The templates are added to your store.



Chapter 5: Working with StoreSense Design Studio

StoreSense provides several options for working with templates. You can use the built-in template tools in Store Administration, or you can use other HTML editors and import your files.

Prior to customizing the templates, you should have a basic working knowledge of HTML, including tables. You should also have a good idea in mind of how your store will look.

StoreSense Design Studio behaves differently depending on the type of browser you are using - this guide will provide alternate instructions for the major browser differences. In addition, many of its features are enhanced when JavaScript is enabled. While you will still be able to edit templates if you have disabled JavaScript, this user guide assumes that JavaScript is enabled. For the best user experience, enable JavaScript in your browser, or upgrade to a browser that supports JavaScript.

Working in StoreSense Design Studio

Opening Design Studio

Design Studio can be accessed by clicking the Design Studio button on the toolbar, or by browsing to **Support | Design Studio**.

To return to Store Administration, click the **Exit Design Studio** button on the toolbar.

The Workspace

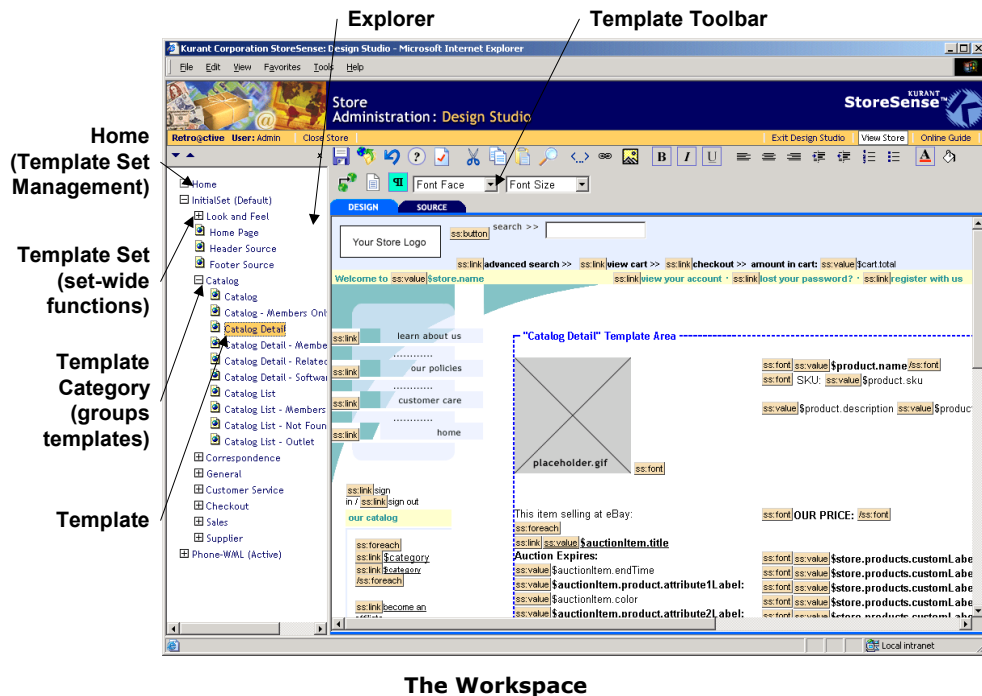
The Design Studio Workspace contains the following areas:

- ◆ The **Toolbar**, which provides access to your site, the Online Guide, and allows you to return to Store Administration.
- ◆ The **Design Studio Explorer**, which allows you to browse to the templates.
- ◆ The **Workspace**, where you can edit templates.

How Design Studio Explorer is Organized

Design Studio Explorer functions are organized as follows:

- ◆ The **Home** page provides access to Template Set management, and to other common tasks, like previewing and publishing your work.
- ◆ The **Template Set** landing page provides access to set-wide functions, including: Template Assistant, Publish, Preview, Export Templates, Add Templates, Baselines and Resets.
- ◆ Each template set contains a look and feel “branch” on the Explorer.
- ◆ Individual templates are contained within each template category. However, the three most commonly edited templates - the Home Page (Storefront Template), Header and Footer - are placed before the template categories list starts.



Getting Help

Design Studio offers the following tools to help you in designing your store.

- ◆ The **Online Guide** button on the toolbar opens a full user guide with information on all the features of Design Studio.
- ◆ Clicking  on a form will bring up a window with help on entering information in the fields just below it.
- ◆ Some pages display help directly on the page in the Help Panel (on the right side of the page).
- ◆ When working in Design mode in Design Studio, click  for template specific help, including help on StoreSense Tags and Business Objects. In addition, if you are using Internet Explorer version 5 or higher, you can copy object expressions and shortcuts from the Online Guide and paste them into your source code or Tag Dialog.

Managing Your Work

Before you begin work on your site, it's important to understand the process of editing and publishing templates.

All templates are contained within sets. Sets allow you to customize your store's appearance based on your users browser settings (among other things), or even allow you to develop different designs that you change based on seasons, holidays, etc. See *Managing Template Sets* (page 45) for more information.

If you have created multiple template sets, be sure to select the correct set in the StoreSense Explorer when making changes, previewing and publishing.

The process of editing and publishing involves the following:

- ◆ If you want to backup your existing design, first “baseline” your templates. See *Setting Baselines* (page 23) for more information.
- ◆ The first step in working with templates is to edit a specific template. See *Using Template Edit Modes* (page 39) for more information.
- ◆ Once you’ve edited a template, you should **save** your work. This saves the changes that you’ve made, but **does not** make them “live.” Once you have saved your work on a template (but have not published it “live”), an “in progress” symbol will appear next to the template in the Explorer. This allows you to see at a glance what templates have been edited and need to be published.
- ◆ If you do not like the changes you have made, you can undo them either using the **Undo** button on the toolbar, or Undo option on the Template Category page. This undoes all changes since the last time you published a template.
- ◆ When you’re done working on all your templates, and want to test them, you can **preview** the templates. See *Previewing Your Work* (page 24) for more information.
- ◆ Finally, when you’re satisfied with your changes and are ready to have them appear to your customers, you can **publish** them to your site. This makes your changes live. See *Publishing Templates* (page 24) for more information.
- ◆ This might also be a good time to “baseline” your templates again. See *Setting Baselines* (page 23) for more information.

This table describes what happens with each step in both the Design Studio and the live storefront.

Action	What happens in Design Studio	What happens in the live storefront
Baseline	No changes are visible, but the existing design is “backed up” for future reference.	No impact.
Edit	Edits can be viewed as they are made.	No impact.
Save	Changes are saved, and templates are marked as “in progress”.	No impact, but you can “preview” your changes in the storefront with your store data.
Publish	Changes are saved, and the “in progress” indicator disappears.	Changes can now be viewed in the live storefront.
Reset	All changes since the last baseline are removed.	Live storefront reverts to the last baselined design.

Setting Baselines

Setting a baseline for all templates means that you can return the templates to this state should you make any additional changes. This function saves a baseline for *all* of your templates in the set. This action is *not* reversible.

It is recommended that you make a baseline of your templates before and after making significant changes to them.

To set a baseline:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Under **Advanced Template Functions**, click the **Set Baseline** link. The Set Baseline page appears.
- 4 Type “yes” and click **Set Baseline**.

Resetting Templates

Resetting your templates returns your templates to the last baseline. Any changes you have made since setting the baseline will be lost.

This action is *not* reversible. Please proceed with caution.

To reset your templates:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Under **Advanced Template Functions**, click the **Reset Templates** link. The Reset Templates page appears.
- 4 Type “yes” and click **Reset Templates**.

Previewing Your Work

Once you have edited your templates, you can preview how they will look before they are published to your live site.

Any transactions that occur while previewing your site (page views, orders, etc.) are “live”, meaning they will count toward report values, order counts, payment processor transactions, etc.

To preview your site:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Under **Template Design**, click the **Preview** link. Your site will appear in a new window for your review.

Publishing Templates

Once you are satisfied with the design of your site, you can publish it - this makes your changes “live” for others to view.

To publish your site:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Under **Template Design**, click the **Publish or Undo Templates in Progress** link. The Templates in Progress page appears, listing all templates that have been saved but not published.
- 4 Select the templates you wish to publish and click **Publish**. Your changes are now live.

To publish individual templates:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Browse to the template you wish to publish.
- 3 In the Template Editor toolbar, click  (Publish). Your changes to that template are now live.

If you are using Design View, changes to the Header and Footer template will be published as well.

To publish more than one template within a template category:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Browse to the template category whose templates you wish to publish.
- 4 Select the template(s) you wish to publish and click **Publish**. Your changes are now live.

Common Tasks

This section contains topics on common tasks to perform in StoreSense Design Studio. All these instructions assume you are working in Design Mode.

Selecting Content

If you are using Internet Explorer 5.5 or higher, Design Mode allows you to select text, tables, images, etc., by using your mouse. The following are tips on how to select different types of content.

Sometimes, you may have a difficult time selecting content because a table has been selected (indicated by the handlebars - the small squares that surround the table). To unselect the table, you can try clicking somewhere outside of the handlebars, or press the [Tab] key on your keyboard (which moves the selection to the next object on the page).

Text

To select text (necessary when applying formatting or deleting a block of text), move the cursor to the left of the text you wish to select, press the mouse button, and “drag” the mouse to the right (and/or down) until you reach the end of the text. Release the mouse button. The text will appear highlighted.



Selected text

Images

You have two options for selecting images.

To select an image in normal Design Mode view, move the mouse over the image, and click once. Handlebars will appear around the image. If there is an `<ss:link>` tag surrounding the image, the tag will be highlighted as well.



Selecting images in normal Design Mode view

To select an image using markers (especially useful when selecting tiny or “hidden” images), click  on the toolbar, move the mouse over the “img” marker, and click once. Handlebars will appear around the image. If there is an `<ss:link/>` tag surrounding the image, the tag will be highlighted as well.



Selecting images using markers

SSML Tags

To select a tag, move the mouse over the tag icon and click once. If the tag is a link or a value, both the tag icon and its text/object expression will appear highlighted. Otherwise, just the opening tag will be highlighted.



Selecting ss:link and ss:value Tags

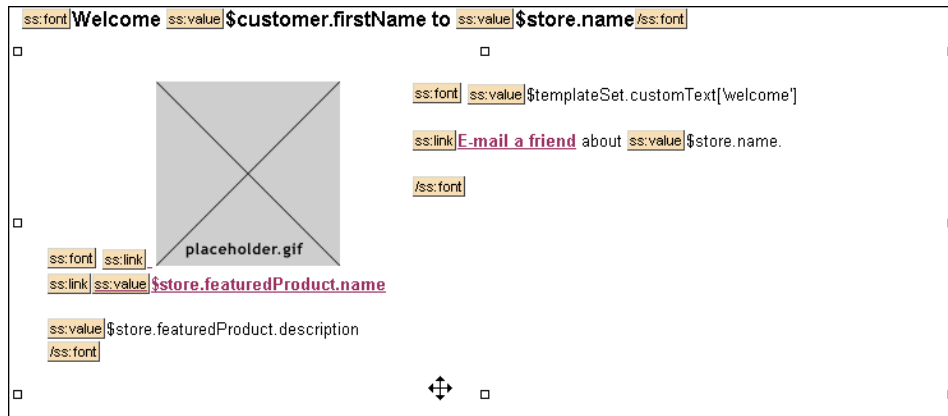


Selecting all other tags

Tables

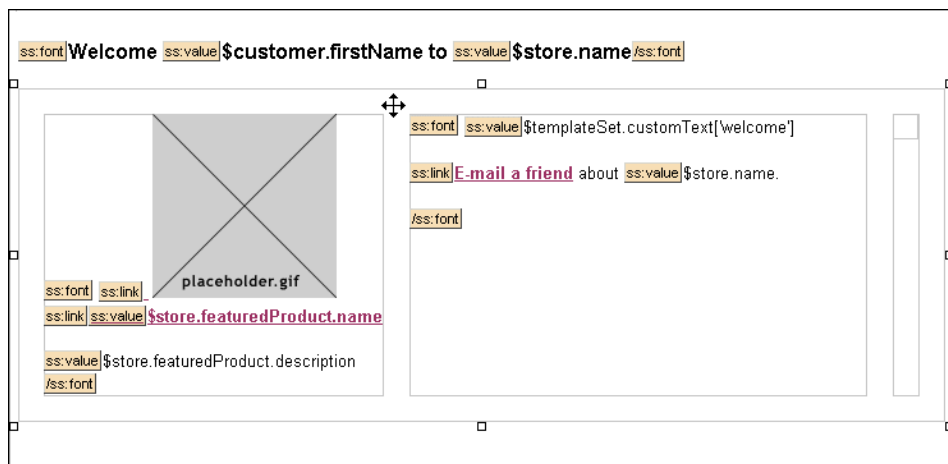
There are several tools to assist you in selecting tables.

If you know where your table borders are, move the mouse over any border and click once. The table will be selected, as indicated by the handlebars.



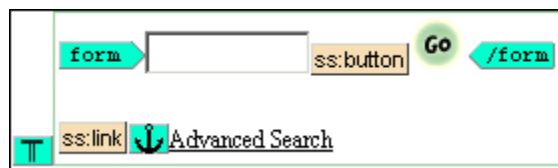
Selecting tables

If you do not know where your borders are, click  on the toolbar - the table borders will appear. Move the mouse over any border and click once. The table will be selected, as indicated by the handlebars.

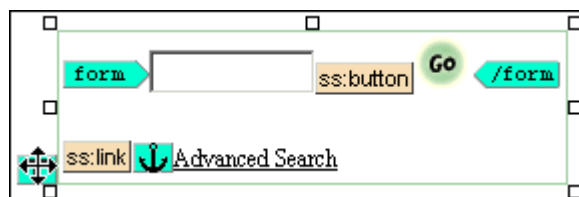


Selecting tables using borders

Another method for selecting tables is by displaying the markers. To do so, click , move the mouse over the  marker, and click once. The table will be selected, as indicated by the handlebars.



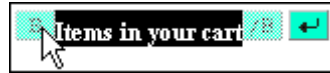
Selecting tables using markers. First find the marker...



...then select it

Formatting

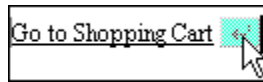
To select a block of content that is formatted (for example, bold or italic text), click , move the mouse over the appropriate marker, and click once. All text within that formatting tag will appear selected.



Selecting formatting

Special Characters

To select special characters such as line breaks, paragraph marks, or comments, click , move the mouse over the appropriate marker, and click once. The marker (and therefore the special character) will appear highlighted.



Selecting special characters

Adding Text

If you are using Internet Explorer 5.5 or higher

To add text to a template:

- 1 Place the cursor in the location in which you want to add text. The cursor is indicated by the flashing bar (|).
- 2 Type the text you wish to add.

For all other browsers

To add text to a template:

- 1 Click on any existing content. A menu appears.
- 2 Select either **Insert Text Before...** or **Insert Text After...** to specify the exact location of the text. The Edit Text page appears.
- 3 Enter the text you wish to add.
- 4 If necessary, specify any formatting options.
- 5 Click **Save** when done. The text appears at the specified location.

Adding Links

If you are using Internet Explorer 5.5 or higher

To add a link to a template:

- 1 Place the cursor in the location in which you want the link to appear. The cursor is indicated by the flashing bar (|).
- 2 Click  on the toolbar. The Add/Edit Link window appears.
- 3 At this point, you have three options:

- ◆ **Pre-Built Links:** The first portion of the **Link Type** list contains frequently used, pre-built links. To use one of these links, select it from the list. If the link requires additional information, it will appear within square brackets ([...]) in the second field. Enter the necessary information.
 - ◆ **<ss:link>:** The next option is to select an `<ss:link/>` tag from the **Link Type** list. If you select this option, you'll then need to enter an object expression in the second field. For help on selecting an object expression, click **Help**.
 - ◆ **Legacy Tags:** Finally, you can select a legacy tag from the **Link Type** list (legacy tags start with "<:"). However, these tags are intended only for use by stores using older versions of StoreSense.
- 4 If necessary, enter the text you wish to have appear hyperlinked in the **Link Value** field.
 - 5 If you wish to have this link open in a new window, select that option from the **Target** list.
 - 6 Click **OK**. The link will appear in the location you specified.

For all other browsers

To add a link to a template:

- 1 Click on any existing content. A menu appears.
- 2 Select either **Insert Link Before...** or **Insert Link After...** to specify the exact location of the link. The Edit Link page appears.
- 3 At this point, you have three options:
 - ◆ **Pre-Built Links:** The first portion of the **Link Type** list contains frequently used, pre-built links. To use one of these links, select it from the list. If the link requires additional information, it will appear within square brackets ([...]) in the second field. Enter the necessary information.
 - ◆ **<ss:link>:** The next option is to select an `<ss:link/>` tag from the **Link Type** list. If you select this option, you'll then need to enter an object expression in the second field. For help on selecting an object expression, click **Help**.
 - ◆ **Legacy Tags:** Finally, you can select a legacy tag from the **Link Type** list (legacy tags start with "<:"). However, these tags are intended only for use by stores using older versions of StoreSense.
- 4 If necessary, enter the text you wish to have appear hyperlinked in the **Link Value** field.
- 5 If you wish to have this link open in a new window, select that option from the **Target** list.
- 6 If necessary, select any formatting options.
- 7 Click **Save**. The link will appear in the location you specified.

Adding Images

If you are using Internet Explorer 5.5 or higher

To add an image to a template:

- 1 Place the cursor in the location in which you want the image to appear. The cursor is indicated by the flashing bar (|).
- 2 Click  on the toolbar. The Image window appears.

- 3 At this point, you have two options:
 - ◆ **Specific Images:** you can point to a specific image that you've already uploaded to your store. This is recommended primarily when adding store design images. To do so, select the **Store Image** option, and select an image from the list.
 - ◆ **Dynamic Images:** if you wish to insert a "dynamic" image (for example, you want to display a product image), select the **URL/Source/Shortcut** option. Then enter the appropriate URL, object expression or shortcut. For help on selecting an object expression or shortcut, click **Help**.
- 4 Enter text to describe the image in the **Alternate Text** field. This can be very helpful in making your site accessible to users with disabilities.
- 5 If necessary, select any other formatting options.
- 6 Click **OK**. The image will appear in the location you specified.

For all other browsers

To add an image to a template:

- 1 Click on any existing content. A menu appears.
- 2 Select either **Insert Image Before...** or **Insert Image After...** to specify the exact location of the image. The Edit Image page appears.
- 3 At this point, you have two options:
 - ◆ **Specific Images:** you can point to a specific image that you've already uploaded to your store. This is recommended primarily when adding store design images. To do so, select the **Store Image** option, and select an image from the list.
 - ◆ **Dynamic Images:** if you wish to insert a "dynamic" image (for example, you want to display a product image), select the **URL/Source/Shortcut** option. Then enter the appropriate URL, object expression or shortcut. For help on selecting an object expression or shortcut, click **Help**.
- 4 Enter text to describe the image in the **Alternate Text** field. This can be very helpful in making your site accessible to disabled users.
- 5 If necessary, select any other formatting options.
- 6 Click **Save**. The image will appear in the location you specified.

Displaying Store Data

Store data includes store address and contact information, system information, customer information, etc.

If you are using Internet Explorer 5.5 or higher

To display store data:

- 1 Place the cursor in the location in which you want to display the data. The cursor is indicated by the flashing bar (|).
- 2 Click  on the toolbar. The Insert/Edit Tag window appears.
- 3 At this point, you have the following options:

- ◆ **Frequently Used Data:** select the data you wish to display from the Frequently Used Tags portion of the **Tags** list.
 - ◆ **<ss:value/> Tag:** if the data you wish to display isn't in this list, then select StoreSense Value Tags from the **Tag Type** list, and then select the `<ss:value source="" />` tag. Enter the object expression that represents the data you wish to display (for example, `$store.name`). For help on selecting an object expression, click **Help**.
- 4 Click **OK**. The `ss:value` tag icon will appear in the location you specified, with the data source appearing to its right.

For all other browsers

To display store data:

- 1 Click on any existing content. A menu appears.
- 2 Select either **Insert Tag Before...** or **Insert Tag After...** to specify the exact location of the link. The Edit Tags page appears.
- 3 At this point, you have the following options:
 - ◆ Select the data you wish to display from the Frequently Used Tags **Tags** list.
 - ◆ If the data you wish to display isn't in this list, then select StoreSense Value Tags from the **Tag Type** list, and then select the `<ss:value source="" />` tag. Enter the object expression that represents the data you wish to display (for example, `$store.name`). For help on selecting an object expression, click **Help**.
- 4 Click **OK**. The `ss:value` tag icon will appear in the location you specified, with the data source appearing to its right.

Building Lists

"Lists" allow you to display the same information for a group of items. For example, you might want to list the product name and price for all items returned in a search. Or, you may want to list the date and status for each order a customer has made in your store.

If you are using Internet Explorer 5.5 or higher

To add a list of information to a template:

- 1 Place the cursor in the location in which you want to add the list. The cursor is indicated by the flashing bar (|).
- 2 Click  on the toolbar. The Insert/Edit Tag window appears.
- 3 From the **Tag Type** list, select **StoreSense Other Tags**.
- 4 From the **Tags** list, select the `<ss:foreach item="" within="" />` tag.
- 5 Enter the name you will use for the items in the list in the *item* attribute, then enter the object expression that represents the source of the list in the *within* attribute. For help on selecting an object expression, click **Help**.
- 6 Click **OK**. The `ss:foreach` tag icons will appear in the location you specified.
- 7 Add the items you wish to have appear in the list. To do so:

- ◆ Place the cursor between the ss:foreach tag icons. One way to do this is to press the Arrow keys on your keyboard until the cursor is between the icons. If you use your mouse to place your cursor between the icons, you may end up *editing* the <ss:foreach> tag you just entered instead of entering a new tag (you'll know this because you'll see the tag in the Markup field). If this occurs, click **Cancel**, and move your cursor using the arrow keys.
- ◆ Add text, tags and images as necessary by clicking the appropriate button on the toolbar.

For all other browsers

To add a list of information to a template:

- 1 Click on any existing content. A menu appears.
- 2 Select either **Insert Tag Before...** or **Insert Tag After...** to specify the exact location of the link. The Edit Tags page appears.
- 3 From the **Tag Type** list, select **StoreSense Other Tags**.
- 4 From the **Tags** list, select the `<ss:foreach item=" " within=" "></ss:foreach>`.
- 5 Enter the name you will use for the items in the list in the *item* attribute, then enter the object expression that represents the source of the list in the *within* attribute. For help on selecting an object expression, click **Help**.
- 6 Click **Save**. The ss:foreach tag icons will appear in the location you specified.
- 7 Add the items you wish to have appear in the list. To do so:
 - ◆ Click on the first ss:foreach tag icon. A menu appears.
 - ◆ Add text, tags and images as necessary by clicking the appropriate button on the toolbar.

Using Conditions

“Conditions” allow you to control the information displayed based on customer properties, store type, system information, etc. For example, you might display a special message to a returning customer, or you could display a certain image on a specified holiday.

If you are using Internet Explorer 5.5 or higher

To display content based on a condition:

- 1 Place the cursor in the location in which you want to add the content. The cursor is indicated by the flashing bar (|).
- 2 Click . The Insert/Edit Tag window appears.
- 3 From the **Tag Type** list, select **StoreSense Other Tags**.
- 4 To control the display of information based on whether a single condition is met or not, select the `<ss:if test=" "></ss:if>` tag.
 - ◆ Enter the condition that must be met within the *test* attribute. For help on creating test conditions, click **Help**.
 - ◆ Enter the content that is to appear if the condition is met between the `<ss:if test=" "></ss:if>` tags. If you want to display content if the condition *is not met*, type `<ss:else/>`, and then type the alternate content.

- 5 To control the display of information based on whether one of several conditions is met, select the `<ss:choose></ss:choose>` tag. Now you need to add your test conditions.
 - ◆ Place the cursor between the `<ss:choose></ss:choose>` tags.
 - ◆ Type `<ss:when test=" "></ss:when>`.
 - ◆ Enter the condition that must be met within the *test* attribute. For help on creating test conditions, click **Help**
 - ◆ Enter the content that is to appear if the condition is met between the `<ss:when test=" "></ss:when>` tags.
 - ◆ Keep adding `<ss:when/>` tags until you have entered all test conditions.
 - ◆ If you want to display content if *none* of the conditions are met, add an `<ss:otherwise>` tag (using the same process).
- 6 Click **OK**. The content to appear if a condition is met will appear (the ss:if, ss:choose and ss:when tag icons do not appear in Design Studio).

For all other browsers

To display content based on a condition:

- 1 Click on any existing content. A menu appears.
- 2 Select either **Insert Tag Before...** or **Insert Tag After...** to specify the exact location of the link. The Edit Tags page appears.
- 3 From the **Tag Type** list, select **StoreSense Other Tags**.
- 4 To control the display of information based on whether a single condition is met, select the `<ss:if test=" "></ss:if>` tag.
 - ◆ Enter the condition that must be met within the *test* attribute. For help on creating test conditions, click **Help**
 - ◆ Enter the content that is to appear if the condition is met between the `<ss:if test=" "></ss:if>` tags. If you want to display content if the condition *is not met*, type `<ss:else/>` and then type the alternate content.
- 5 To control the display of information based on whether one of several conditions is met, select the `<ss:choose></ss:choose>` tag. Now you need to add your test conditions.
 - ◆ Place the cursor between the `<ss:choose></ss:choose>` tags.
 - ◆ Type `<ss:when test=" "></ss:when>`.
 - ◆ Enter the condition that must be met within the *test* attribute. For help on creating test conditions, click **Help**
 - ◆ Enter the content that is to appear if the condition is met between the `<ss:when test=" "></ss:when>` tags.
 - ◆ Keep adding `<ss:when/>` tags until you have entered all test conditions.
 - ◆ If you want to display content if *none* of the conditions are met, add an `<ss:otherwise>` tag (using the same process).
- 6 Click **OK**. The content to appear if a condition is met will appear (the ss:if, ss:choose and ss:when tag icons do not appear in Design Studio).

Saving and Undoing Your Work

The following table details your save, undo and delete options.

Function	Button	Description
Save		Saves your changes without making them “live.” (Internet Explorer 5.5 and higher only)
Save and Publish		Saves your changes <i>and</i> makes them live.
Undo		Undoes all changes since the last time you <i>saved</i> the template.
Delete		Deletes the current template. Only appears on custom templates you have added.

Advanced Tasks

This sections contains topics on more advanced tasks you can perform in StoreSense Design Studio.

Template Validation

StoreSense offers you the option of checking your templates for valid SSML tag syntax. Validation will check for the following:

- ◆ Incorrect tag syntax, such as not closing a tag, or missing required attributes.
- ◆ Use of business objects and/properties that are not valid for your store type.
- ◆ Misspelled tags, objects and properties.

In addition, validation can check for the use of legacy StoreSense tags. To do so, go to the Design Studio home page, click the **StoreSense Options** link, and select the **Validation** check box.

Templates are validated when:

- ◆ A template is saved.
- ◆ A template is published.
- ◆ The **Validate All Templates** link is selected from the template set, Advanced Template Functions area.

The following is a list of common error messages, with explanations.

Error Message	Explanation
DEPRECATED: Tag TAG NAME is from an older version of StoreSense and support for it may be discontinued in the future. Please update to a more current alternative.	This error will only be displayed if you select the Validation check box in StoreSense Options . To update a legacy tag to an SSML tag, please refer to the Tag Mapping.
ERROR: Attribute ATTRIBUTE NAME in tag	This error will occur if an attribute value is not the correct type. For example, if you use a integer instead of an object expression.

Error Message	Explanation
TAG NAME does not have a valid datatype.	
ERROR: Attribute ATTRIBUTE NAME must have a valid value.	Occurs if an attribute value is empty. For example, <code><ss:value source="" /></code> .
ERROR: Attribute ATTRIBUTE NAME is required for the TAG NAME tag.	Occurs if a required attribute is missing from the tag.
ERROR: Root element not available for source string OBJECT EXPRESSION on this template.	Occurs if the object may not be used on the current template. For example, using the <code>\$product</code> object on the Registration template will produce this error. This error can also occur when using an <code><ss:foreach item="itemname" within="withinname"></code> tag, if the object expression does not use the <code>\$itemname</code> value. For example, <code><ss:foreach item="listing" within="\$catalog.results"><ss:value source="\$listing.name" /></ss:foreach></code> is valid code, but <code><ss:foreach item="listing" within="\$catalog.results"><ss:value source="\$catalogheading.name" /></ss:foreach></code> is <i>not</i> valid.
ERROR: Source string OBJECT EXPRESSION is not valid. The property or method does not exist (DETAIL).	Occurs when the specified object expression does not exist. This usually occurs if there is a typo in the expression (for example, <code>"\$sytem.currentDate"</code> instead of <code>"\$system.currentDate"</code>), or if an invalid expression is used (for example, <code>"\$system.date"</code> instead of <code>"\$system.currentDate"</code>).
ERROR: Tag TAG NAME does not possess any meaningful attributes.	Occurs if the specified tag has no valid attributes.
ERROR: Tag TAG NAME is invalid.	Occurs when the tag name is not valid, usually the result of a typo (for example, <code>"ss:valu"</code> instead of <code>"ss:value"</code>).
ERROR: Tag TAG NAME requires children tags or text, none are present.	Occurs when a tag requires content, but none is present. For example, <code>StoreSense<ss:if test="\$customer.isLoggedIn"/></code> will result in this error, since content needs to be defined that will display if the test condition is met.
ERROR: Tag TAG NAME was not closed.	Occurs when a tag that must be closed is not closed. For example, an <code><ss:if></code> tag that does not have a corresponding <code></ss:if></code> closing tag will produce this error.
ERROR: The TAG NAME tag must have a parent of one of the following tags: LISTOFVALIDTAGS.	Occurs when a required parent tag cannot be found. For example, if an <code><ss:when/></code> appears without an <code><ss:choose></ss:choose></code> tag surrounding it.
ERROR: This property or method referenced by ErrorSecurityFailed may not be accessed from this template due to security restrictions.	Occurs if the object expression is not valid for the store or tag. For example, using a "read-only" object expression (which is indicated in the documentation with an asterisk) within an <code><ss:edit/></code> tag will produce this error (because read-only object expressions do not allow information to be "written" to them). In addition, using an object expression that is only valid within Professional Edition stores on a template that's in a Standard Edition store will produce this error as well.
ERROR: Validation error assigning name ITEM ATTRIBUTE value WITHIN ATTRIBUTE (DETAIL).	Usually occurs when the object expression specified in the "within" attribute is not valid. Invalid object expressions include those caused by typos, or objects that are not valid on the current template.

Error Message	Explanation
ERROR: Validation error assigning name NAME ATTRIBUTE value ATTRIBUTE VALUE (DETAIL).	Usually occurs when the object expression specified in the “value” attribute is not valid. Invalid object expressions include those caused by typos, or objects that are not valid on the current template.
ERROR: Value ATTRIBUTE VALUE is not legal for attribute ATTRIBUTE NAME in tag TAG NAME.	Occurs if the value specified for an attribute is not valid.
The date formatting string: FORMAT was invalid.	Occurs if the date format specified in the <i>format</i> attribute is not valid.
WARNING: Attribute ATTRIBUTE NAME in tag TAG NAME does not have a valid datatype.	Occurs if the specified attribute contains the wrong type of data. For example, this error would occur if you used an object expression instead of an integer in a “maxlength” attribute (<code><ss:value source="\$store.name" maxlength="\$store.customers"/></code>).
WARNING: Attribute ATTRIBUTE NAME is not supported by the TAG NAME tag, and will be ignored.	Occurs when an attribute that is not supported by the specified tag appears in the tag. The unsupported attribute will be ignored.
WARNING: Tag TAG NAME was not closed.	Occurs when a tag that should be closed is not closed. For example, <code><ss:value source="\$store.name"></code> or <code><ss:image source="\$store.storeLogo"></code> . Since these tags <i>never</i> contain content, StoreSense will ignore the missing close symbol, and the tag will function as expected.

Template Caching

While StoreSense templates are dynamic (meaning, the tags and data are added when the user requests the page), StoreSense does “cache” templates in order to speed up the time it takes to view the page. This means that StoreSense saves a partially resolved version of the template and uses that saved version to display to the customer, instead of resolving the template from scratch.

When you make changes to the design and content of your store, StoreSense clears those cached templates. However, if you find that changes you’ve made aren’t appearing in the storefront, you can “force” the template cache to clear.

To do so, click the **Clear Template Cache** link from the template set page, Advanced Template Functions area.

Exporting Templates

If you want to edit your templates in an external editor, you can export them to your local machine.

To export a template:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Under **Advanced Template Functions**, click **Export Templates**. The Export Template page appears.
- 4 Select the template you wish to export from the list.
- 5 Select the platform to which you will be exporting the file from the list.
- 6 Click **Export**. The Export Results page appears.
- 7 Click the **Download** link to download the file.
- 8 Browse to the location on your local drive to which you would like to download the template, and click **OK**. The template will begin downloading.
- 9 After the template has been exported, another window will appear telling you that the download is complete.

You can now edit the template(s) that you have downloaded in the editor of your choice. Once you are done with your edits, you will need to import the template.

The default file extension for downloaded templates is *.tpl. While most HTML editors should be able to open files with this extension, you can download the file with a different extension by changing it in this window.

Importing and Adding Templates

If you've exported your templates and edited them in an external editor, you will need to import them back into your store.

Some WYSIWIG HTML editors insert proprietary tags into the HTML, or will convert special characters into ASCII code (for example, changing "<" to "<"). Please be sure to review the actual HTML code before importing a template, or use your editor's HTML mode for editing.

To import an existing template:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Under **Advanced Template Functions**, click **Import or Add Templates**. The Import or Add Template page appears.
- 4 Under **Existing Templates**, select the template you will be importing, and browse to the template file you wish to import.
- 5 Click **Add** to import the template.

Importing existing templates replaces the existing template. If you are working in an active template set, these imported files are then immediately live. Use this function with caution. You can import any text file (.htm, .txt, etc.).

You can add templates to your store, either to add more information, or to customize search results or product detail pages.

To add a new template:

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Select **Initial Set**. The Templates page appears.

If you are using multiple template sets, select the appropriate template set.

- 3 Under **Advanced Template Functions**, click **Import or Add Templates**. The Import or Add Template page appears.
- 4 Under **New Templates**, enter the short **Name** (no spaces or special characters allowed) and **Description** for the template. The name is what you will refer to in links to the template, and the description will appear in any template selection lists. In addition, you will need to select the **Type** of template you are importing from the list.
- 5 Select the method for creating the new template.
 - ◆ **Import a New Template** imports a template you've created outside of StoreSense.
 - ◆ **Copy an Existing Template** creates a template from the template you select. This is ideal when creating multiple catalog lists or detail templates.
 - ◆ **Copy from Base Template Set** creates a template from the template you select. However, this template is pulled from a default template set (defined by your Site Administrator), and contains none of the unique formatting you've applied to the template in your current set.
 - ◆ **Add a Blank Template** creates an empty template.
- 6 Click **Add** to create the template.

Once you have added the template, you can link to it by using the `<ss:link source="$templateSet.Templates['ShortName']" />` tag.

Changing Template Properties

Template properties include global page properties such as margins, meta-data, background images, etc. While you can edit these properties on any template, they only impact the Header template.

To change a template's properties:

- 1 Click . The Edit Properties page appears.
- 2 Enter the information in each field. See *Field Help: Property Elements* (page 38) for more information.
- 3 Click **OK** or **Save** when done. The properties are updated.

Field Help: Property Elements

Title

Enter the title of the page. This title will appear in the customer's browser title bar.

Header and Footer

If you have created custom headers and footers, you can specify that this template will be displayed with those footers by selecting them from the lists. This option will only appear if you have added custom headers and footers.

Background

Field	Description
Background Image	To use an image that has already been uploaded to your store, select Store Image and then select the image from the list. To specify an image using a URL, select URL , and enter the full URL. To display the image in a fixed position (that remains in place even as the user scrolls the page), select the Fixed Position Background check box.
Background Color	Enter the color name or hexadecimal color value (using the '#' symbol) for your background color.

Margins

Enter the number of pixels to use as your margins.

Meta Data

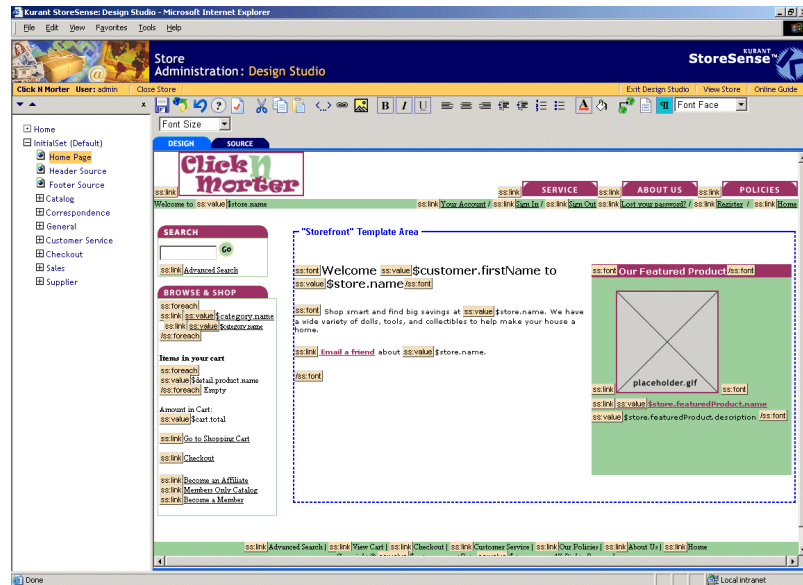
Field	Description
Keywords	Enter keywords that describe your site and that will make it easier for search engines to find your site.
Description	Enter a brief description of your site.
Copyright	Enter your site's copyright information.
Content Language	Use this to describe the primary language of your site.

Using Template Edit Modes

StoreSense Design Studio offers two “modes” in which you can edit your templates:

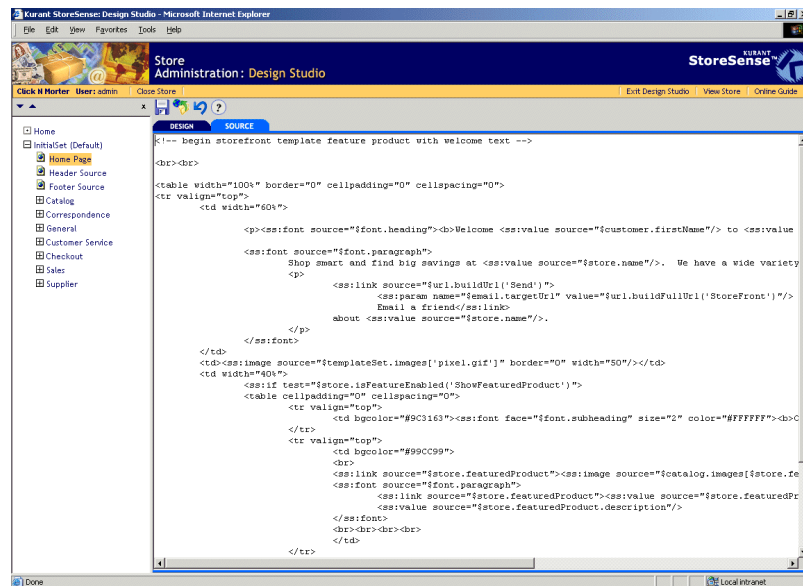
Design vs. Source Mode

Most templates will open in Design Mode. This gives you a WYSIWIG view of your template, with the Header and Footer template as well. See *Working in Design Mode* (page 41) for more information.



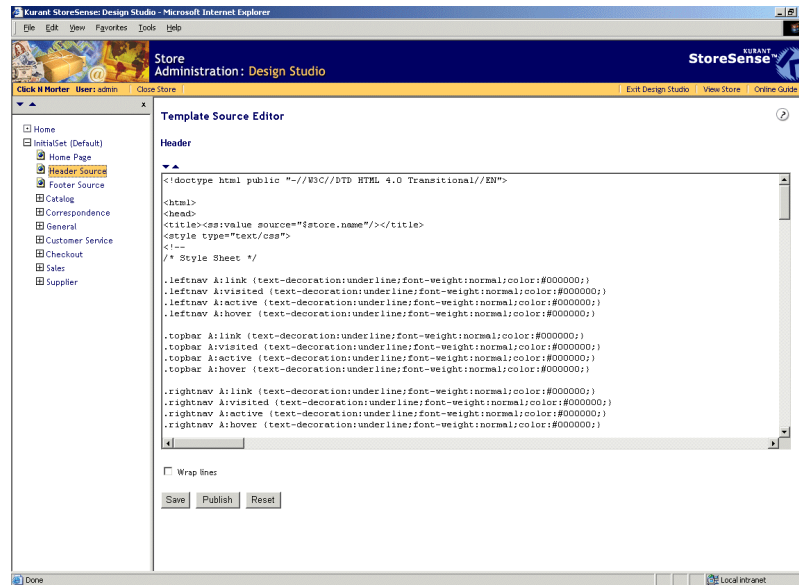
Design Mode

Templates that open in Design Mode also offer a Source Mode. This displays the source code of the template, without the Header or Footer template code. See *Working with Template Source Code* (page 43) for more information.



Source Mode

Other templates - like templates used to generate e-mails - will open in the Template Source Editor. This displays the source code of the template, without the Header or Footer template code, but functions differently than the Source Mode. See *Working with Template Source Code* (page 43) for more information.



Template Source Editor

You can select the default mode you would like to use in Design Studio | Options. To do so, click the **StoreSense Options** link, and select the **Design/Source Preference** option.

Browser Differences

While you can use StoreSense Design Studio in any browser version that supports HTML 4.0 and Cascading Style Sheets (such as Microsoft Internet Explorer 4.0 or Netscape versions 6.2 or higher), it is optimized for use with Internet Explorer 5.5 or higher.

- ◆ If you use Internet Explorer 5.5 or higher, you can do all your editing and formatting using the Template Toolbar.
- ◆ If you use Netscape 6.2 or higher, you will be able to add or insert elements using a “pop-up” menu, and then do all your editing and formatting in the resulting page.
- ◆ If you use another browser, you will be able to add or insert elements by making a selection from a form, and then do all your editing and formatting in the resulting page.

This guide assumes you are using Internet Explorer 5.5 or higher, or Netscape 6.2 or higher.

Working in Design Mode

The Design Mode gives you a WYSIWYG (“what you see is what you get”) view of your templates, with the Header and Footer templates active for editing as well. In Design Mode, your page appears almost exactly as it will when it’s live, with the exception of the StoreSense Tags (which appear as tags, and not as their resolved values).

While Internet Explorer versions 5.5 and higher allow you to edit text directly in Design Mode, do **not** type any source code (such as HTML or StoreSense tags) directly in Design Mode. Instead, use the Tag Element to enter or edit these tags. If you prefer to work with source code, you can switch to Source Mode. See *Working with Template Source Code* (page 43) for more information.

When saving a template in Design Mode, your source code may be reformatted (white space may be removed, element names will be capitalized, etc). This will not change the content of your code, or affect the look of your template. However, if you prefer to retain your source code formatting, you may want to work exclusively in the Source Mode, or export and import your templates instead. Please see *Working with Template Source Code* (page 43) or *Exporting Templates* (page 36) for more information.

Templates open in Design Mode **with** the Header and Footer template as well, allowing you to edit all three templates. It's important to remember that changes made to the Header or Footer area when working on another template will affect the look of your entire site. If you make changes to these areas, the Header and Footer templates will display the "in progress" symbol. In addition, when you save or publish a template while in Design Mode, the Header and Footer templates will be saved/published as well.

If you are using Internet Explorer version 5.5 or higher, the first time you use Design Mode, you will be prompted to "Run Design Studio". This installs the control required for some of the Design Mode's features. Click **Yes**, and the control will be installed.

Template Toolbar

The Template Toolbar appears when editing a template in Design Mode, and provides quick access to key template editing tasks, depending on the browser version you are using.

Internet Explorer 5.5 and Higher

- ◆ Save functions (save, publish, undo since publish). See *Saving and Undoing Your Work* (page 34) for more information.
- ◆ Delete function (only available for custom templates).
- ◆ Help on the current template.
- ◆ Formatting Functions (bold, font color, alignment, lists etc).
- ◆ Editing Functions (copy, paste, etc.).
- ◆ Element Functions (insert or edit element).
- ◆ Visual Aids (show gridlines, show markers).

Other Browsers

If you are working in a browser other than Internet Explorer 5.5, the only functions that will appear on the toolbar are Save and Publish, Undo, Delete (if applicable) and the Properties Element button.

Keyboard Shortcuts

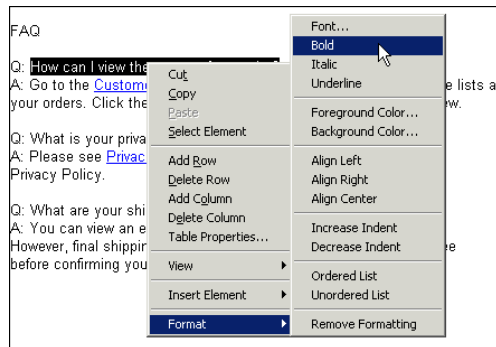
If you are using Internet Explorer 5.5 and higher, you can use several keyboard shortcuts to simplify editing tasks.

- | | |
|-------------------------------|-----------------------|
| ◆ Move the cursor: Arrow keys | ◆ Redo: [CTRL]+Y |
| ◆ Copy: [CTRL]+C | ◆ Bold: [CTRL]+B |
| ◆ Paste: [CTRL]+V | ◆ Italic: [CTRL]+I |
| ◆ Cut: [CTRL]+X | ◆ Underline: [CTRL]+U |
| ◆ Undo: [CTRL]+Z | |

The Context Menu

If you are using Internet Explorer version 5.5 or higher, and you're using a mouse with two or more buttons, StoreSense offers a "context" menu to provide quick access to common functions. Some of these functions include basic editing (cut, copy, paste), formatting (bold, italics), selecting a visual aid, or inserting and adding HTML code.

To access the menu, put your cursor in the location where you want to add or edit content, and press the *right* mouse button. The context menu appears.



Using the Context Menu

Visual Aids

If you are using Internet Explorer version 5.5 or higher, StoreSense offers several tools to help you understand the design and layout of your page.

“**Show Borders**” will display a light gridline around all tables and images. This can help you understand the structure of your page, and place elements in the correct location. To enable this view, click . Click the button again to hide the borders.

“**Show Markers**” provides visual descriptions of the HTML code on your page, such as line breaks, comment codes, and anchor tags (hyperlinks). To enable this view, click . Click the button again to hide markers.

In addition, most SSML tags are indicated by an icon. Exceptions include images and some advanced tags.

Some SSML tag icons may not appear in the location in which you expect them. For example, some `<ss:foreach/>` tags may appear outside of the table in which they are located. This is due to the tag's location in the source code, and does not affect the functionality of the tag.

Working with Template Source Code

In addition to WYSIWIG template editing using Design Mode, StoreSense Design Studio offers two options for source code editing.

Source Mode

Templates that open in Design Mode also offer a Source Mode. This displays the source code of the template, without the Header or Footer template code. You can view or edit the source code for the Header and Footer template by browsing to those templates in the StoreSense Design Studio Explorer.

Working with Source Mode is much like working in a basic text editor. However, while the Toolbar is still visible, the only functions available while in this view are Save, Save and Publish, Undo Since Last Publish and Delete (if applicable).

Template Source Editor

Some templates - like the Header and Footer templates, templates used to generate e-mails, or those templates that are displayed within other templates - open in the Template Source Editor. This displays the source code of the template, without the Header or Footer template code, and functions differently than the Source Mode.

- ◆  makes the area containing the source code shorter.
- ◆  expands the area containing the source code.
- ◆ Select the **Wrap Lines** check box to wrap the code (instead of allowing the code to run past the right of the box). This check box will only appear in browsers supporting this feature.
- ◆ The Save, Publish, Undo and Delete options are available as buttons on the bottom of the page (there is no Template Toolbar in this view). In addition, **Reset** undoes all changes since the last save.

When working with template source code, it is important that you avoid using the ASCII equivalent of the ampersand symbol (&). If you must escape an ampersand, use '&'; *not* '&';



Chapter 6: Using Template Sets

Template Sets are the set of templates that make up the look and feel of your store. You can create multiple template sets that can customize a user's experience, or that are for draft development.

This chapter provides instructions on using template sets.

Understanding Template Sets

With StoreSense you can create multiple designs of your store, without having to duplicate your store, customer or product information. Some examples of reasons why you might create multiple designs are:

- ◆ You want to change the design of your store based on the season or time of year.
- ◆ You have a large number of customers who use older browsers, and you want to customize the design of your store for them.
- ◆ You have customers in different countries, and you want to create a version of your store in different languages.
- ◆ You want to completely redesign your store, but want to do so without touching your current store's design.

StoreSense does this by allowing you to create multiple “template sets”. “Template Sets” are the complete set of templates that make up a store design. When you first purchase a StoreSense store, your store contains one template set, which you can then design using the Store Setup Wizard or StoreSense Design Studio. Most likely, this is all you'll need to develop a functional and creative store design.

However, if you want to create an additional design, as described in the examples above, you can do so by adding another template set. You can create a blank design, create a new set based on your existing design, or create a set design by running the Template Assistant. See *Managing Template Sets* (page 45) for more information on creating template sets.

Once you've created your additional design, you can then specify when you want to display that design to your customers. If you want to replace your old design with your new one, you would make the original template set inactive, and then make the new template set the default. If you want to have both designs active, but display one to one type of browser user or language, then you would modify that set's rules to only display that set when a certain browser setting is in effect. See *Managing Template Sets* (page 45) for more information on editing browser rules, or *Setting the Template Set Language* (page 47) for editing language rules.

Managing Template Sets

To add a new template set:

- 1 Open Design Studio and click the **Manage Template Sets** link. The Manage Template Sets page appears.

- 2 Under **Add Template Set**, enter a unique name for your set. The set name cannot contain spaces or special characters.
- 3 Select the method you wish to use to create the set.
 - ◆ **Template Assistant** allows you to create a new set by going through the Setup Wizard.
 - ◆ **StoreSense Template Set** allows you to copy one of the application's default template sets. Select the set you wish to copy from the list.
 - ◆ **Store Template Set** allows you to copy one of your store's existing template sets. Select the set you wish to copy from the list.
- 4 Click **Add**. The set is created, and appears in the **Current Template Sets** list. If you chose the **Template Assistant** method, the Setup Wizard will appear.

To change the settings for an existing template set:

- 1 Open Design Studio and click the **Manage Template Sets** link. The Manage Template Sets page appears.
- 2 Under **Current Template Sets**, click the name of the template set with which you wish to work.
 - ◆ The **Language** section allows you to display this set when a customer's browser is configured for a particular language. See *Setting the Template Set Language* (page 47) for more information.
 - ◆ The **Rules** sections allow you to tell the application what templates to display to a customer based on information their browser sends to it. See *Applying and Defining Template Set Rules* (page 47) for more information.

The following tools are available for the current template set:

- ◆ **Set Default** makes this the default set to display if a customer does not match the rules or language settings for any other set. See *Applying and Defining Template Set Rules* (page 47) for more information.
- ◆ **Delete** removes the set from the store completely. You cannot delete the default template set, or your only template set.
- ◆ **Activate** makes the set available to customers to your store.
- ◆ **Deactivate** makes the set unavailable to store customers.
- ◆ **Redesign** will start the Setup Wizard, allowing you to redesign the complete template set. Redesigning the template set will overwrite all design and text edits you have made to these templates. Also, if you have customized any images that are called by StoreSense tags (such as buyit.gif or join.gif), the Template Assistant will overwrite those images. Please back up your custom images before continuing.
- ◆ **Preview** allows you to view a template set design without having to make the set your active set. A new window will open with a banner indicating you are in preview mode. Any transactions completed while in preview mode are live, and will be included in reports and orders.

To set options for your template sets:

- 1 Open Design Studio and click the **Manage Template Sets** link. The Manage Template Sets page appears.
- 2 Select the check box of the set whose option you wish to change.

- 3 Click the desired button.

You can “lock” a visitor to your store to a specific template set by having them click a link that uses the following syntax:

```
<a href="$url.buildUrlWithParameter('storefront','tsetlock','AnotherSet') ">Another Set</a>
```

- ◆ The first item in the parentheses ('storefront') indicates the specific template to which you wish to send the visitor.
- ◆ The second item is the command that will set a cookie to the visitor's machine locking them to a template set.
- ◆ The third item is the name of the template set.

Once the visitor clicks this link, they will only see the specified template set for the rest of their visit.

Setting the Template Set Language

To display this set when a customer's browser is configured for a particular language:

- 1 Open Design Studio and click the **Manage Template Sets** link. The Manage Template Sets page appears.
- 2 Under **Existing Template Sets**, click the name of the set with which you wish to work. The Template Set Rules page appears, listing any existing rules at the top of the page.
- 3 Select the language in which this set will appear.
- 4 Click **Submit**. Now this template set will appear whenever a customer's browser language setting matches the selection made here.

Applying and Defining Template Set Rules

Template Set Rules allow you to tell StoreSense what templates to display to a customer based on information their browser sends to the application. For example, if a customer is accessing your store from an older browser, you can create a custom set for those customers, and then apply a rule that will tell StoreSense to display that set when your store is visited by that browser.

To apply a standard rule set to this template set:

- 1 Open Design Studio and click the **Manage Template Sets** link. The Manage Template Sets page appears.
- 2 Under **Existing Template Sets**, click the name of the set with which you wish to work. The Template Set Rules page appears, listing any existing rules at the top of the page.
- 3 Under **Add Standard Rule**, select the rule you wish to apply to the set. See *Field Help: Template Set Rules* (page 48) for more information.
- 4 Click **Add**. The rule is added to the **Existing Rules** list.

In addition, you can develop a custom rule (for example, to create a set that will be displayed based on the customer's browser settings).

To create a custom rule:

- 1 Open Design Studio and click the **Manage Template Sets** link. The Manage Template Sets page appears.
- 2 Under **Existing Template Sets**, click the name of the set with which you wish to work. The Template Set Rules page appears, listing any existing rules at the top of the page.
- 3 Under **Add Custom Rule**, enter the information in each field. See *Field Help: Template Set Rules* (page 48) for more information.
- 4 Click **Add**. The rule is added to the **Existing Rules** list.

To delete a rule:

- 1 Open Design Studio and click the **Manage Template Sets** link. The Manage Template Sets page appears.
- 2 Under **Existing Template Sets**, click the name of the set with which you wish to work. The Template Set Rules page appears.
- 3 Under **Existing Rules**, select the check box of the rule you wish to delete and click **Delete**. The rule is removed from the **Existing Rules** list.

Field Help: Template Set Rules

Add Standard Rule

Field	Description
Evaluation Logic	The And/Or options work as follows: all “or” rules are checked, and then all “and” rules are checked. For example, if you define five rules as follows: ◆ Rule 1: OR ◆ Rule 2: AND ◆ Rule 3: OR ◆ Rule 4: AND ◆ Rule 5: OR They will be read as follows: Rule 2 AND Rule 4 AND (Rule 1 OR Rule 3 OR Rule 5).
Standard Rule	The following are rules are provided with StoreSense: ◆ Default: If the customer’s browser does not match any other rule, the default template set will be displayed. ◆ WML: If the customer is viewing the store using the UP.Browser version 3.X, the template set with this rule applied will be displayed. ◆ HDML: If the customer is viewing the store using the UP.Browser version 4.X, the template set with this rule applied will be displayed.

Add Custom Rule

Field	Description
Evaluation Logic	See <i>Standard Rules</i> (page 48) for more information.
Rule Header	Working with custom rules requires in-depth knowledge of browser headers and values. For more information, see the HTTP specification at http://www.w3.org/Protocols/ . Do not enter header values with hyphens - use the

Field	Description
	underscore character instead.
Rule Match	The Match list determines how the Value will be matched. There are several Match options. See the table below for match rules options and examples.
Rule Value	Enter the value to be matched in the rule. For more information, see the HTTP specification at http://www.w3.org/Protocols/ .

The following table lists sample **Match** options and values, to demonstrate how matches are made:

Option	Value:	Successful Match	Unsuccessful Match
Contains	dog	cats and dogs are animals	cats are animals
Ends With	dog	cat and dog	dog and cat
Matches Exactly	dog	dog	dogs
Starts With	dog	dog and cat	cat and dog



Chapter 7: Advanced Design Concepts

This section covers advanced store design concepts, such as using multiple templates and working with files.

Working with Images

Images are used throughout your store to create your overall store design, as well as provide visual information regarding your products and categories.

Uploading Images

You can upload both single image files, and archived zip files. You can also upload images using the **Upload Images** wizard.

The maximum file size for uploading a single image or image archive is 10 MB. Image names may not contain spaces.

Category, Manufacturer and Product Images

These are images that are used to describe your product catalog. To upload a category, manufacturer or product image or archive:

- 1 Browse to **Support | Upload Images**. The Upload Images page appears.
- 2 Under **Shared Images**, select the image type you are uploading from the list.
- 3 Enter the file and path name, or click **Browse** to browse to its location on your local drive.
- 4 Select the type of file you're uploading from the list.
- 5 Click **Upload**. The image(s) will be uploaded to the appropriate directory in the Store Content Area. If you uploaded an archived zip file, the images will be expanded in the directory to which you have imported them, and the archive file will be deleted.

Store Logo

You can specify an image that serves as your store logo. Once you have indicated that you wish to use a logo, and have uploaded it, the image will appear in the storefront.

If the image does not appear in the storefront, add the following code to the location in which you would like the logo to appear:

```
<ss:if test="$store.isFeatureEnabled('StoreLogo') ">  
  <ss:image source="$store.storeLogo" border="0" />  
</ss:if>
```

To upload a store logo:

- 1 Browse to **Support | Upload Images**. The Upload Images page appears.
- 2 Under **Store Logo**, enter the file and path name, or click **Browse** to browse to its location on your local drive.

- 3 Click **Upload**. The image(s) will be uploaded to the appropriate directory in the Store Content Area.

To remove a store logo:

- 1 Browse to **Support | Upload Images**. The Upload Images page appears.
- 2 Under **Store Logo**, select the **I won't be using a store logo** check box.
- 3 Click **Upload**. The store logo will no longer appear in the storefront.

Store Design Images

These are images that used for your general store design, such as theme images. There are several methods for uploading these images.

To upload an image from your machine using StoreSense Design Studio:

You must be using Internet Explorer version 5.5 or higher to use this feature.

- 1 Click the **Design Studio** button on the toolbar. The StoreSense Design Studio appears.
- 2 Browse to the template to which you wish to add an image.
- 3 Find the image you wish to add on your local machine, and then “drag” it onto the template. The Upload and Insert Image window appears.
- 4 Set the options for the image, and click **Insert**. The image is inserted into the template.

To upload a store design image or an image archive using the Upload Images tool:

- 1 Browse to **Support | Upload Images**. The Upload Images page appears.
- 2 Under **Template Set Images**, enter the file and path name, or click **Browse** to browse to its location on your local drive.

If you are using multiple template sets, select the appropriate template set.

- 3 Select the type of file you're uploading from the list.
- 4 Click **Upload**. If you uploaded an archived zip file, the images will be expanded in the directory to which you have imported them, and the archive file will be deleted.

You can also upload an image from another web site using the StoreSense Design Studio Copy Center tool (if you are using Internet Explorer version 5.5 or higher).

Using and Updating Store Images

Once you've uploaded your store images, you'll want to make them available to your store.

If you have uploaded images with the same file names and pixel sizes as existing images, such as replacing banners (assuming pixel sizes are specified in the template), the image can immediately be viewed in your store (you may need to refresh the page you're currently viewing in the browser to see the change).

If you have uploaded images with the same file names but *different* pixel sizes, you'll need to edit the templates containing those images and update the width and height tags of those images.

For new images, you can add them to your templates by editing the templates in which you want to display the image. The `<ss:image source="$store.images['imagenam.e.xt']">` tag is used to define the path to the location where your images are stored.

Using Product (Catalog) Images

To display images with your products, you'll want to enter the appropriate file name when adding or editing a product (using the Store Administration Product), and then use the appropriate StoreSense tag is used so that image is displayed. There are two possible product images:

- 1 **Photo:** this is the larger of the two images, typically seen in the Catalog Detail template. The corresponding tag that displays this image is `<ss:image source="$product.photoImageName" />`.
- 2 **Thumbnail:** this is the smaller of the two images, and is typically seen in the Catalog List template. The corresponding tag that displays this image is `<ss:image source="$catalog.images[$product.thumbnailImageName]" border="0" />`.

Using an Image in Place of the "Submit" Button

To have a GIF or JPEG image appear in place of the standard Submit form button, use the following HTML code instead.

```
<ss:button source="$templateSet.images['imagenam.e.xt']">
```

Where "imagenam.e.xt" is the name of the image. You will need to upload this image before this code will work.

Customizing Tag Images

The store makes use of several images to perform an action. These images are standard GIF files that you can customize to give your store a unique look. It is highly recommended that you make a copy of the original images before making any changes.

To make changes to these images, you can download them to your local machine using the File Management feature, and then upload them using the Upload Images feature.

Customizations to these images will be overwritten if you run the Setup Wizard, Template Assistant, or redesign your template set. Please back up your custom images.

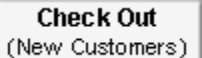
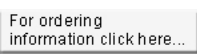
Image Name	Tag	GIF Name	Image*
Buy It	<code><ss:link shortcut="BuyIt"/></code>	buyit.gif	
Buy It Now	<code><ss:link shortcut="BuyItNow"/></code>	buyitnow.gif	
Change	<code><ss:button source="\$templateSet.images['change.gif']" border=0/></code>	change.gif	
Checkout New Customers	<code><ss:form shortcut="Checkout"/></code>	checkout.gif	

Image Name	Tag	GIF Name	Image*
Checkout Returning Customers	<ss:form shortcut="CheckoutPro"/>	checkoutpro.gif	
Closed	<ss:image source="\$templateSet.images['closed.gif']"/>	closed.gif	
In Stock	<ss:image source="\$templateSet.images['instock.gif']"/>	instock.gif	
Join/Subscribe	<ss:image source="\$templateSet.images['join.gif']"/>	join.gif	
Lead	<ss:image source="\$templateSet.images['lead.gif']"/>	lead.gif	
Lead	<ss:form shortcut="AddToCartImage"/>	lead.jpg	
Members Only	<source="\$templateSet.images['members.gif']"/>	members.gif	
Register	<ss:image source="\$templateSet.images['register.gif']"/>	register.gif	

*Some themes may come with alternatives for these images. However, the same principles for customizing those images apply.

Working with Files

In addition to the templates that StoreSense provides, you can also add static pages and files to complement your store.

Uploading Files

This option is only available if it has been enabled by your Site Administrator

You can upload both single files (such as HTML pages, multimedia files, JavaScript files, style sheets, etc.), and archived zip files. Images should be uploaded using the *Uploading Images* (page 50) feature.

The maximum file size for uploading a single file or file archive is 10 MB.

To upload an image or an image archive:

- 1 Browse to **Support | Upload Files**. The Upload Files page appears.
- 2 Select the file type you are uploading from the list.
- 3 Enter the file and path name, or click **Browse** to browse to its location on your local drive.
- 4 Select the type of file you're uploading from the list.

- 5 Click **Upload**. The file(s) will be uploaded to the root directory in the Store Content Area. If you uploaded an archived zip file, the files will be expanded in the directory to which you have imported them, and the archive file will be deleted.

Once you have added the file, you can create a link to it using the `$store.files['FILENAME.xxx']` syntax.

Creating a Splash Page

Advanced Mode

This option allows you to create a new HTML page that search engines can find and index to better market your store. However, some search engines will *not* spider “splash” pages, so use this feature with caution.

This tool creates a very simple page that you will likely want to edit. To do so, use the Support Manager | File Management tool to download the file, make your edits, and then upload it (either using FTP, or the Upload Static Page tool) when done.

To create a static page:

- 1 Browse to **Support | Create Static Page**. The Create Static Page page appears.
- 2 Enter a file name (including the .htm or .html extension) for the new page. If you use “default.htm”, this page will overwrite the page that has links to your Store Administration area and Storefront. If you do not want to overwrite this page, use another name.
- 3 Enter a **Title** for the static page. This appears in the title bar of the browser. Some search engines use this, some don’t.
- 4 Enter a brief description of your store in the **Meta Description** field. Again, some search engines use this, while others don’t. Also, some search engines will use this text to describe your site when displaying search results. While there are no formal limits on the number of characters you can use, generally most engines will look at the first 200 characters.
- 5 Select the desired options under **Place in Meta Tags** to place the categories and/or products inside meta tags. Some search engines look in meta tags and some don’t. Including product information could result in a very large page.
- 6 Select the desired options under **Place in Page Body** to place the categories and/or products inside the main body of the page. Again, including product information could result in a very large page.
- 7 Click **Submit** to save your work and create the page.
- 8 You can download and edit the page by using the *Managing Files* (page 55) feature. The page is located in the Store Content area of the File Manager.

Once the file is created, you can browse to it by entering one of the following addresses in your browser’s address box:

- ◆ If you have a dedicated domain name (such as `www.mystore.com`), use “`www.mystore.com/stores/[store shortname]/[filename.ext]`”
- ◆ If you have a shared domain name (such as `www.stores.com/mystore/`), use “`www.stores.com/[filename.ext]`”

Cascading Style Sheets

Linked style sheets allow you to define all of your font and color information in one central location.

If you are familiar with style sheets and wish to use them to format your store:

- 1 Upload your *.CSS file to your store using the Support Manager | Upload Files feature (if it has been enabled by your Site Administrator). See *Uploading Files* (page 53) for more information.
- 2 Edit your header template(s) to contain the following code (between the <head> </head> tags:

```
<link rel="STYLESHEET" type="text/css" href="$store.files['FILENAME.css']">
```
- 3 Edit all your templates to remove the font tags and replace them with the appropriate elements and classes (such as <p></p>, <h1></h1>, etc.).

JavaScript

JavaScript allows you to add scripting functionality to your store (such as hover buttons, popup windows, etc.).

If you are familiar with JavaScript and wish to use it in your store:

- 1 Upload your *.js file to your store using the Support Manager | Upload Files feature (if it has been enabled by your Site Administrator). See *Uploading Files* (page 53) for more information.
- 2 Edit your header template(s) to contain the following code (between the <head> </head> tags:

```
<SCRIPT LANGUAGE="JavaScript" SRC="$store.files['FILENAME.js']"></SCRIPT>
```

Multimedia Files

StoreSense has done some testing with including different multimedia in your pages (such as Flash files). However, we recommend that you thoroughly test including such files on different platforms and browsers.

Multimedia files should be uploaded to your store using the Support Manager | Upload Files feature (if it has been enabled by your Site Administrator). See *Uploading Files* (page 53) for more information. Use the <ss:media/> tag to reference your file.

Managing Files

File Management allows you to view, manage and move files located in your directory on the server.

There are three content areas:

Content Area	Description:
Store Content Area	Contains all files that can be viewed via a browser, such as static HTML pages and images.
Store Data Area	Contains all files that either cannot or should not be viewed via a browser, including configuration and log files.
FTP Area	If your Site Administrator has provided you with an FTP directory on the server that is outside of your Store Data Area, this link will provide you with access to that FTP area. See your Site Administrator about providing you with an FTP directory.

To switch from one area to another, click the link under **Switch to**.

To manage your server files:

- 1 Browse to **Support | File Management**. The File Management page appears.
- 2 To view a file or image, click its link. You can also “right-click” the link to download the file.
- 3 To open a folder, click its link.
- 4 To go back to the folder containing the current folder, click the **Parent Directory** link.
- 5 To delete a folder or file, select the folder or file and click **Delete**. Deleting a folder will also delete any folders and files contained within it.
- 6 To copy a folder or file:
 - ◆ Select the check box of the folder or file and click **Copy**. The folder(s) and file(s) are placed on the clipboard.
 - ◆ Go to the folder into which you want to copy the folder or file and click **Paste**.
- 7 To move a folder or file:
 - ◆ Select the check box of the folder or file and click **Cut**. The folder(s) and file(s) are removed from their current location and are placed on the clipboard.
 - ◆ Go to the folder into which you want to move the folder or file and click **Paste**.
- 8 To download a file:
 - ◆ Right-click on the file you want to download. A context menu appears.
 - ◆ Select “Save Target As...” or “Save Link Target As”. A Save As window appears.
 - ◆ Browse to the location to which you wish to save the file and click **Save**. The file is saved to the specified location.

Omitting the Header and Footer

StoreSense allows you to prevent the Header and Footer templates from displaying on a page by page basis. This allows you to achieve custom designs for individual templates. For example, if you want your storefront home page to have a different look than the rest of your templates, you can omit the Header and Footer from that template.

To do so, simply add a `<HTML>` tag as the first item on the appropriate template, and a `</HTML>` template as the last item. Then add your custom design to the template. In the example of creating a custom design for your home page, edit the Storefront template to include an `<HTML>` tag at the beginning, and an `</HTML>` tag at the end. Then add your custom design, and publish the template. Your home page now appears without the Header and Footer template.

Customizing the Display of the Category List

By default, the themes that are shipped with StoreSense will display all active categories, and all subcategories, in the store, with links to the search results for that category. This list is displayed on either the header or the footer of the store, and on some home page layouts.

You can control the “depth” of this list from within Store Administration, on the **Product | Categories** landing page. You can display:

- ◆ Top-level categories only.
- ◆ All categories and subcategories (default).
- ◆ Subcategories up to a specified level.

These settings only affect the use of the `$catalog.categoryList` member. Any other members that display categories are not affected.

Using Multiple Templates

StoreSense gives you the option of using different versions of the same template, depending on the information you are trying to display. Using different templates based on a category, product or manufacturer allows you to customize the look and feel of your store right down to the product detail level.

For example, if you have a pet store, you may decide to use a different background pattern and banner when displaying catalog lists for cat related products than for dog related products.

To do so:

- 1 Create unique header templates for each category (header templates are usually where banners and background patterns are defined).
- 2 Define the categories “Cats” and “Dogs”, assigning the appropriate header templates to each category.
- 3 Make sure you assign a category to each Cat and Dog related product.

Where You Can Use Multiple Templates

There are several different criteria for which you can define the use of different templates: Category, Product and Manufacturer.

Using multiple templates based on Category allows you to completely change the look of your catalog listing based upon the category that the customer chose (either by clicking a link you provided, or by selecting a category on your search page).

- ◆ The Header template allows you to change the body color and background, header banner, and navigation.
- ◆ The Catalog List template allows you to change how the listing of products will appear.
- ◆ The Footer template allows you to change the appearance of any navigation you have along the bottom of your pages, and completes the code that was started in the Header template.

Using multiple templates based on product allows you to change the layout for each product detail page, as well as define a different order extension template for each product.

- ◆ The Catalog Detail template affects the look and feel of the page that contains detailed information regarding this product.

- ◆ The Order Extensions template allows you to customize different order extension layouts for each product.
- ◆ If some of your products use different attribute information than Color and Size, the Cart Add template can be customized to display the appropriate attribute labels. If you create a custom Cart Add template, you'll also need to create a custom Catalog Detail template that refers to your new Cart Add template.

Using multiple Catalog List templates based on manufacturer allows you to display unique catalog lists when displaying search results by manufacturer.

Using multiple Supplier templates based on supplier allows you to customize the notifications (fax, EDI) you send for each supplier.

How to Create and Use Multiple Templates

To create and use multiple templates:

- 1 Add a new template by copying an existing template. See *Importing and Adding Templates* (page 37) for more information.
- 2 Open the page in your favorite HTML editor and customize the template.
- 3 Import the template. See *Importing and Adding Templates* (page 37) for more information. Since this is a new template, be sure to add a **Name** and **Description** for the template, and select a template **Type**.
- 4 Once the template has been imported, it will appear in the appropriate lists.
 - ◆ Header and Footer templates will appear on the Category page.
 - ◆ Catalog List templates will appear on the Category and Manufacturers pages.
 - ◆ Order Extensions templates will appear on The Product Manager page.
 - ◆ Supplier templates will appear on the Supplier Profile page.
 - ◆ Catalog Detail templates will appear on the Product Profile.
- 5 If you created a custom Cart Add template, you'll also need to create a custom Catalog Detail template that refers to your new Cart Add template.

User Defined Templates

User defined templates allow you to quickly add pages to your store without using the Static Page feature. User defined templates are displayed with the store's header and footer, and can be used for content such as adding pages to your "About Us" area, or additional Customer Service information.

To add a user defined template:

- 1 Create your new template.
- 2 Import the template. See *Importing and Adding Templates* (page 37) for more information. Since this is a new template, be sure to add a name and description for the template, and select *User Defined Template* from the **Type** list.

- 3 Once the template has been imported, you can create a link to it from any page in your store using the `<ss:link source="$store.files['TemplateName']"/>` tag.

To delete a user defined template:

- 1 Edit the template using the Design Studio.
- 2 On the toolbar, click **Delete**.

Only user defined templates may be deleted.

Design FAQ

The following are common design questions and answers.

How do I change the background colors used in a theme?

While StoreSense offers a variety of color combinations in its themes, you may want to customize the theme you've chosen by changing one or more of its background colors. There are three methods for changing the background color - the method you will use depends upon the theme you have selected, and the design of your store.

If you want to change the background color of the body of the page:

- 1 Open Store Administration, and then open Design Studio by browsing to **Support | Design Studio**.
- 2 Browse to **InitialSet (Default) | Home Page**.
- 3 Click . The Edit Properties page appears.
- 4 Enter the new color in the **Background Color** field.
- 5 Click **OK** or **Save** when done. The background color of the body page is updated.

If you want to change the background color in a specific table cell, and you are using Internet Explorer version 5.5 or higher:

- 1 Open Store Administration, and then open Design Studio by browsing to **Support | Design Studio**.
- 2 Browse to **InitialSet (Default) | Home Page**.
- 3 Place your cursor in the cell whose background color you wish to change.
- 4 Right-click the mouse - the context menu appears. Select the **Table Properties** option. See *The Context Menu* (page 43) for more information.
- 5 Enter the new color in the **Background Color** field, or click  to select a color.
- 6 Click **OK** when done. The background color of the table cell is updated.

If neither of these methods work, or don't fit your situation:

- 1 Open Store Administration, and then open Design Studio by browsing to **Support | Design Studio**.
- 2 Browse to **InitialSet (Default) | Header Source**.
- 3 Select the **Wrap Lines** check box for easier reading of the code.

- 4 Look for either “background-color” or “bgcolor”, and replace the value with your new color values.
- 5 Click **Save**. Your work is saved, but the changes are not live in your store.
- 6 Repeat these steps for the **Footer Source**.
- 7 Select the **Home Page** link in the Explorer and review your changes. If you like your changes, click  (Publish). Your storefront will now show the new background colors.

How do I add my company logo to my store?

You can add a company logo to your store either using the Setup Wizard, or by uploading an image and adding the `<ss:image source="$store.storeLogo"/>` tag to your store (if it's not already there). See *Uploading Images* (page 50) for more information.

How do I undo a mistake I made when working on my templates?

StoreSense Design Studio offers several opportunities for you to undo your work.

If you...	...then do this
Have <i>not</i> saved or published your changes:	Use your operating system's or browser's “Undo” command. In Windows, this is usually [CTRL]+Z. In certain browsers, you can find the “Undo” command on the “Edit” menu.
Have <i>saved</i> your changes, but haven't published them:	Click  . This undoes all changes since the last time you <i>saved</i> the template.
Have <i>published</i> your changes:	If you've already published your changes, you'll need to use the “Reset Templates” feature. See <i>Resetting Templates</i> (page 24) for more information. However, this will undo all changes you've made to <i>all</i> templates since the last time you set the baseline.

How do I redesign my store without running the full Setup Wizard?

If you want to redesign your store, but you don't want to go through the Settings section of the Setup Wizard, you can run the Template Assistant instead. This will allow you to select a new theme, layouts, and fonts and colors. However, running the Template Assistant will overwrite *any* changes you've made to your store design.

To make changes to your storefront **without** overwriting your edits, see *Modifying a Store's Look and Feel* (page 14).

How do I add meta-data to my store?

Most “spiders” sent out by today's search engines recognize dynamically created HTML pages, but you can increase the chances of your site being picked up by adding “meta” information to your store.

To do so:

- 1 Open Store Administration, and then open StoreSense Design Studio by browsing to **Support | Design Studio**.
- 2 Browse to **InitialSet (Default) | Home Page**.
- 3 Click . The Edit Properties page appears.
- 4 At the bottom of the page, enter the meta data.
- 5 Click **OK** or **Save** when done.

- 6 Publish your changes. Your store now contains meta data that can be picked up by search engine spiders.

How do I change the fonts used in my store?

To make changes to your fonts, see *Modifying a Store's Look and Feel* (page 14).

How can I create a new design for my store while leaving the existing design as it is?

Sometimes you may need to redesign your store, but you don't want affect your existing design while doing so. Template Sets offer one solution. By creating a new template set for your redesigned store, you can do your work, leaving the existing design untouched.

To do so:

- 1 Create a new template set. See *Managing Template Sets* (page 45) for more information. Make sure the new set is *Inactive*.
- 2 Now, to redesign your store, be sure to edit templates within the new set you created.
- 3 You can preview your changes by selecting the new set name, and then selecting "Preview".
- 4 When you are ready to launch your redesigned site, change the status of the new set to *Default*, and then change the status of the old set to *Inactive*. See *Managing Template Sets* (page 45) for more information.



Chapter 8: Code Snippets

This chapter contains snippets of SSML and HTML code that show the power and flexibility of StoreSense.

Adding Multiple Products to a Cart

You can create a Catalog List template that will allow your customers to select multiple products and add them to their cart with one click. To do so, insert the following code into the Catalog List template:

```
<table border="0" cellpadding="2" cellspacing="4">
<tr valign="bottom" bgcolor="indigo">
  <td align="center">Check To<br>Order</td>
  <td>Product</td>
  <td align="right">Our Price</td>
</tr>
<ss:form shortcut="CartStart">
  <ss:foreach item="listing" within="$catalog.results">
    <ss:foreach item="product" within="$listing.products">
      <tr valign="top">
        <td align="center">
          <ss:if test="$product.hasAttributes != true">
            <ss:checkbox name="cartchk" value="$product.productNumber"/>
          </ss:if>
        </td>
        <td><ss:link source="$product"/></td>
        <td align="right"><ss:value source="$product.displayPrice"
format="money"/></td>
      </tr>
    </ss:foreach>
  </ss:foreach>
</table>
<p><ss:button type="submit" value="Add To Shopping Cart"/></p>
</ss:form>
```

Quick Search Form

You can create a mini keyword search form that can appear on any template (such as on the header template), so that your registers customers don't have to go to the Member Login template. To do so, insert the following code on the desired template:

```
<form method=post action="$templateSet.searchResults">
<input type="text" name="keyword" size=8>
<ss:button source="$templateSet.images['imagename.gif']" alt="Search" border="0"/>
</form>
```

Where “imagename” refers to the name of a “go” image you have uploaded to your store. If you wish to use a standard form button instead of an image, use `<ss:button type="submit" value="Submit"/>`

You may also add font and table formatting as needed.

UPS Tracking Link

You can add a link to the Invoice template that will take your customers directly to the UPS Package Tracking site, and display tracking information for a specific order. To do so, edit the appropriate template (such as Invoice or Customer Service) and add the following code:

The values in this form may change if UPS makes changes to its tracking application.

```
<form method="post" target="_blank"
action="http://wwwapps.ups.com/etracking/tracking.cgi">
  <input type="hidden" name="InquiryNumber1" value="$invoice.shipTrackingNumber">
  <input type="hidden" name="TypeOfInquiryNumber" value="T">
  <input type="hidden" name="AcceptUPSLicenseAgreement" value="yes">
  <input type="submit" name="submit" value="Track Package">
</form>
```

Collapsible Category Lists

The following code, placed on the Header or Footer template, will create a list of categories that will “expand” to include a list of subcategories when a link is clicked.

```
<ss:foreach item="category" within="$catalog.categoryList()">
  <ss:if test="$category.treeLevel == '1'">
    <br><ss:link source="$category"/><br>
  <ss:else/>
    <ss:if test="$request.getParameter(category) == null">
      <ss:foreach item="i" within="$system.iterator($category.treeLevel)">
        &nbsp;
      </ss:foreach>
      <ss:link source="$category"/><br>
    <ss:else/>
      <ss:set name="clickedcat"
        value="$catalog.categories.getItem($request.getParameter(category))"/>
      <ss:if test="clickedcat">
        <ss:if test="$clickedcat.parent.name == $category.parent.name OR
          $clickedcat.name == $category.parent.name">
          <ss:foreach item="i" within="$system.iterator($clickedcat.treeLevel)">
            &nbsp;
          </ss:foreach>
          <ss:link source="$category"/><br>
        </ss:if>
      </ss:if>
    </ss:if>
  </ss:if>
</ss:foreach>
```

Product Price Formatting

By default, StoreSense uses the `<ss:value macro="attributedetailfull"/>` to display all product and attribute pricing information. If you want to change the way in which that information is displayed, you can replace that macro with the following code, and then make the desired modifications.

```
<ss:foreach item="productprice" within="$product.finalPriceList">
  <ss:choose>
    <ss:when test="$counter.totalRows == 1">
      <ss:value source="$productprice"/>
    </ss:when>
    <ss:otherwise>
      <ss:choose>
```

```

        <ss:when test="$counter.isLast">
            <font color="#FF0000"><ss:value source="$productprice"/></font>
        </ss:when>
        <ss:otherwise>
            <s><ss:value source="$productprice"/></s>
        </ss:otherwise>
        </ss:choose>
    </ss:otherwise>
</ss:choose>
    &nbsp;
</ss:foreach>
<ss:foreach item="promotion" within="$product.activePromotions">
    <ss:set name="detailText" value="$product.promotionDetailText($promotion)"/>
    <ss:if test="$detailText != null">
        &nbsp;(<ss:value source="$detailText"/>)
    </ss:if>
</ss:foreach>
<br>
<ss:if test="$product.hasAttributes">
    <ss:foreach item="attribute" within="$product.attributes">
        <ss:foreach item="price" within="$attribute.finalPriceList">
            <ss:choose>
                <ss:when test="$counter.isLast">
                    <font color="#FF0000"><ss:value source="$price"/></font>
                </ss:when>
                <ss:otherwise>
                    <s><ss:value source="$price"/></s>
                </ss:otherwise>
            </ss:choose>
            &nbsp;
        </ss:foreach>
        <ss:foreach item="promotion" within="$product.activePromotions">
            <ss:set name="detailText" value="$product.promotionDetailText($promotion)"/>
            <ss:if test="$detailText != null">
                &nbsp;(<ss:value source="$detailText"/>)
            </ss:if>
        </ss:foreach>
        <ss:value source="$attribute.attributeText"/><br>
    </ss:foreach>
</ss:if>

```

Hide Attribute Information When Not In Stock

(Professional Edition only)

This code can be used on the Catalog Detail Template to prevent the product attribute size information from displaying when the attribute quantity is 0. It may be modified to do the same for the color attribute.

```

<td>
    <font face='arial, helv' size='2'><b>Size</b>&nbsp;
        <select name='size'>
            <ss:foreach item=attr within=$product.attributes>
                <ss:if test=$attr.quantity <> 0>
                    <option value=$attr.size><ss:value source=$attr.size/></option>
                </ss:if>
            </ss:foreach>
        </select>
    </font>
</td>

```

Order Extension Information on the Invoice Email

To display Order Extension information on the Invoice - Email template, add the following code somewhere between the `<ss:foreach item="detail" within="$invoice.details">...</ss:foreach>` tag.

```
<ss:foreach item="extensions" within="$detail.extensions">
  <ss:value source=$extensions.extensionText1/>
  ...
</ss:foreach>
```

Suppressing A Shipping Method

By default, StoreSense expects the merchant to use the three available shipping methods provided by the StoreSense shipper. If you wish to use only one or two, you can use the code below.

Checkout and Checkout New Customer Templates

Delete the following code...

```
<p>
  <b>Shipping Method</b><br>
  <ss:foreach item="shipQuote" within="$cart.shippingQuotes">
    <ss:radio name="shipme" source="$shipQuote.shipMethod"
checked="$counter.currentRow==1"/>&nbsp;<ss:value source="$shipQuote.amount"
format="money" />)
    <br/>
  </ss:foreach>
</p>
```

...and replace it with this code (removing the option you do *not* wish to offer):

```
<p>
  <b>Shipping Method</b><br>
  <input type="radio" name="shipme" value="S" checked>&nbsp;US Postal Service Ground
($5.95)<br/>
  <input type="radio" name="shipme" value="2">&nbsp;US Postal Service Priorit
($7.95)<br/>
  <input type="radio" name="shipme" value="N">&nbsp;Next Day ($14.95)<br/>
</p>
```

Invoice Template

Replace...

```
<ss:include template="selectshipping"/>
```

...with

```
<ss:value source="$cart.shipping"/>
```

Subcategory "Drill-Down" on Catalog List

The following code modifies the Catalog List template to do one of the following:

- ◆ If the user clicks a category link and that category has subcategories, those subcategories will be displayed on the Catalog List template as links.

- ◆ If the user clicks a category or subcategory link that does *not* have any categories, the Catalog List will display a list of product search results.

```
<ss:set name="viewedcategory" value="$request.getParameter('category')"/>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
<tr>
    <td align="center" valign="bottom" class="mainlinks"></td>
</tr>
<tr>
    <td align="center" valign="bottom" class="mainlinks">
        <ss:set name="NOsubcat" value="1"/>
        <ss:foreach item="category" within="$catalog.categoryList()">
            <ss:if test="$viewedcategory">
                <ss:if test="$category.parentchaindisplay == $viewedcategory">
                    <ss:foreach item="subcat" within="$category.childrenToDepth(1)">
                        &nbsp;&nbsp;&nbsp;<ss:link source="$subcat"/><br>
                        <ss:set name="NOsubcat" value="$subcat"/>
                    </ss:foreach>
                </ss:if>
            </ss:if>
            <ss:else/>
                <ss:if test="$category.treeLevel == '1'">
                    <ss:else/>
                        <font size="-2">&nbsp;&nbsp;&nbsp;<ss:link source="$category"/></font><br>
                    </ss:if>
                </ss:if>
            </ss:foreach>
            <ss:if test="$NOsubcat == '1'">
                <ss:foreach item="listing" within="$catalog.results">
                    <ss:sortBy item="product"/>
                    <ss:foreach item="product" within="$listing.products">
                        <tr valign="top">
                            <td nowrap>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</td>
                            <td>
                                <ss:font source="$font.paragraph">
                                    <ss:link source="$product">
                                        <ss:image source="$catalog.images[$product.thumbnailImageName]"
border="0"/>
                                </ss:link>
                                </ss:font>
                            </td>
                            <td width="100%">
                                <ss:font source="$font.paragraph">
                                    <ss:link source="$product">
                                        <b><ss:value source="$product.name"/></b>
                                    </ss:link>
                                    <br>
                                    <ss:value source="$product.brief"/><br>
                                </ss:font>
                            </td>
                            <td nowrap align="left">
                                <ss:font source="$font.paragraph">
                                    <b>Price</b>
                                    <br>
                                    <ss:include macro="productprice"/>
                                </ss:font>
                            </td>
                            <td nowrap>
                                <ss:font source="$font.paragraph">
                                    <ss:link shortcut="BuyIt"/>
                                    <br><br>
                                    <ss:link shortcut="BuyItNow"/>
                                </ss:font>
                            </td>
                        </tr>
                    </ss:foreach>
                </td>&nbsp;&nbsp;&nbsp;</td>
                <td colspan="4"><hr noshade></td>
            </tr>
        </ss:foreach>
    </ss:foreach>

```

```

        </ss:if>
      </td>
    </tr>
  </table>
<!-- end catalog list -->

```

Comparison Category List

The following code, placed on a custom Catalog List template, will allow your visitors to compare products within a category.

Main Catalog List Template

```

<form method="GET" action="$templateSet.templates['prodcompare']">
<input type="hidden" name="template" value="prodcompare">
<input type="submit" name="action" value="Compare"/>
<ss:set name="productCount" value="$system.getMath(1)"/>

<ss:foreach item="listing" within="$catalog.results">
  ....
  <ss:foreach item="product" within="$listing.products">
    <input type="checkbox" name="$system.concat('p', $productCount.get())"
value="$product.productNumber"/>
    <ss:directive source="$productCount.set($productCount.add(1))"/>
    [Display usual product info here]
  </ss:foreach>
</ss:foreach>
</form>

```

Custom Catalog List Template

This template should have the short name specified in the <form> tag above.

```

<p><ss:font source="$font.heading">Product Comparison</ss:font></p>
<ss:set name="selCount" value="$system.getMath(0)"/>

<ss:if test="$request.getParameter('p1')">
<ss:directive source="$selCount.set($selCount.add(1))"/>
<ss:set name="p1"
value="$store.products.getProductWithNumber($request.getParameter('p1'))"/>
</ss:if>

<ss:if test="$request.getParameter('p2')">
<ss:directive source="$selCount.set($selCount.add(1))"/>
<ss:set name="p2"
value="$store.products.getProductWithNumber($request.getParameter('p2'))"/>
</ss:if>

<ss:if test="$request.getParameter('p3')">
<ss:directive source="$selCount.set($selCount.add(1))"/>
<ss:set name="p3"
value="$store.products.getProductWithNumber($request.getParameter('p3'))"/>
</ss:if>

[And so on... for the number of products that you support on a single page
of search results (since that's the max that can be compared at once).]

<ss:if test="$selCount.get() > 0">
  <p>You selected <ss:value source="$selCount.get()"/> products to compare.</p>
  <table>
    <tr>
      <td>Product name:</td>
      <ss:if test="$p1"><td><ss:value source="$p1"/></td></ss:if>
      <ss:if test="$p2"><td><ss:value source="$p2"/></td></ss:if>
      <ss:if test="$p3"><td><ss:value source="$p3"/></td></ss:if>
      <!-- And so on for max possible products that can be selected per page
    </tr>
  </table>

```

```

        <td>Description:</td>
        <ss:if test="$p1"><td><ss:value source="$p1.description"/></td></ss:if>
        <ss:if test="$p2"><td><ss:value source="$p2.description"/></td></ss:if>
        <ss:if test="$p3"><td><ss:value source="$p3.description"/></td></ss:if>
    </tr>
    <tr>
        <td>Price:</td>
        <ss:if test="$p1"><td><ss:value source="$p1.finalPrice"/></td></ss:if>
        <ss:if test="$p2"><td><ss:value source="$p2.finalPrice"/></td></ss:if>
        <ss:if test="$p3"><td><ss:value source="$p3.finalPrice"/></td></ss:if>
    </tr>

    [And so on... for each data element that's relevant to compare.
    You can also use custom fields and attributes to compare.]

</table>
<ss:else/>
    <p>Please select 2 or more products to compare and try again.</p>
</ss:if>

```

Display Related Products on Cart Template

(Professional Edition only)

The following code, added to the Cart template, will display a list of products related to those in a customer's cart (pro stores only)

```

<ss:if test="$cart.Details">
    <p>You may also be interested in the following products:</p>

    <ul>
        <ss:foreach item="detail" within="$cart.Details">
            <ss:foreach item="relatedproducts" within="$detail.product.relatedProducts">
                <li><ss:link source="$relatedproducts"><ss:value
source="$relatedproducts.name"/></ss:link></li>
            </ss:foreach>
        </ss:foreach>
    </ul>
</ss:if>

```

Show Only Attributes That Are On Sale

(Professional Edition only)

This code can be used on the Catalog Detail Template to only display those attributes that are on sale.

```

<ss:set name="previousColor" value="null"/>
<ss:foreach item="productattr" within="$product.attributes">
    <ss:if test="$previousColor != $productattr.color">
        <ss:if test="$productattr.basePrice != $productattr.finalPrice">
            <ss:foreach item="attrPrice" within="$productattr.finalPriceList">
                <ss:if test="$attrPrice != $productattr.finalPrice">
                    <ss:set name="previousColor" value="$productattr.color"/>
                    <ss:if test="$attrPrice != $productattr.basePrice">
                        <font color="#AA0000"><s><ss:value source="$attrPrice"/></s></font>
                    <ss:else/>
                        <s><ss:value source="$attrPrice"/></s>
                    </ss:if>
                </ss:if>
            </ss:foreach>
        </ss:if>
        <font color="#AA0000"><ss:value source="$attrPrice"/></font>
    </ss:if>
</ss:foreach>
<ss:value source="$productattr.size"/><ss:value source="$productattr.color"/>

```

```
<br>
</ss:if>
</ss:if>
</ss:foreach>
```

Show Only Attributes That Are On Sale

This code can be used on the display the total number of items that are in a cart. For example, if the cart contains five of one line item, and three of another line item, this code would display “8”.

```
<ss:set name="math" value="$system.getMath('0')"/>
<ss:foreach item="detail" within="$cart.details">
  <ss:directive source="$math.set($math.add($detail.quantity))"/>
</ss:foreach>
Total quantity: <ss:value source="$math.get()"/>
```

If you use this code on the Cart template, you can just add the `<ss:set>` tag before the existing `<ss:foreach>`, add the `<ss:directive>` tag anywhere within the `<ss:foreach>`, and then add the Total Quantity line to the end (after the `</ss:foreach>`).



Chapter 9: Testing Your Design

Once you've edited your templates, it's time to test your design.

This chapter makes testing recommendations, and provides instructions to help you test your store.

Equipment and Software to Test

To ensure that your store can be viewed and easily used by the maximum number of customers, we recommend that you test your system using a variety of software and platforms, including (but not limited to), the following:

- ◆ **Operating Systems:** such as Windows XP, Windows ME, and Macintosh. Don't assume that if your design works on one operating system, it will work in all versions of that system.
- ◆ **Browsers:** at a minimum, you should test using Internet Explorer versions 5 and 6, and Netscape versions 4 and 6. However, there are still customers with older browsers that you may want to consider testing.
- ◆ **Monitor Resolution:** the most common monitor resolution is 800x600, but there are still significant numbers of customers with smaller monitor resolutions, such as 640x480. We recommend setting your monitor resolution to the minimum you want to require of your customers when testing - refer to the software that comes with your operating system for more information on setting monitor resolution.
- ◆ **JavaScript:** some customers disable JavaScript in their browsers for security reasons. If you have added JavaScript to your pages, or if you are using the columnar Catalog List layouts provided by StoreSense, we recommended disabling JavaScript in your test browser and testing your store to ensure that it is still usable.

How to Test

Following are recommendations on how to test your store.

Testing Store Information Templates

Store Information is typically contained on the following templates. To test the design of these templates, you can either navigate to them from the storefront, or use the following URL structure:

`http://[storeurl]/Page.bok?template=[templatename]`

...where [templatename] is the template short name.

- ◆ About Us [about]
- ◆ Policies [policy]
- ◆ Store Locator [locator] (optional template)

- ◆ FAQ [faq] (optional template)
- ◆ Privacy Statement [privacy] (optional template)

Testing the Product Catalog Templates

Following are some scenarios to test your product catalog design:

- ◆ Do a product search. Does the Catalog List appear as expected?
- ◆ View the detail for any product. Does the page appear as expected?
- ◆ View the detail for any product that has color or size attributes. Does the “add to cart” portion of the page appear as expected?
- ◆ View the detail for any product that has related products. Does the “Related Products” portion of the page appear as expected? (*Professional Edition only*)
- ◆ If applicable, search for a product that uses a custom Catalog Detail template. Does the page appear as expected?

Testing the Checkout Templates

Following are some scenarios to test your checkout-related pages.

- ◆ Purchase a product and check out as a new customer. Do the templates appear as expected?
- ◆ If applicable, purchase a product that uses an Order Extensions template, and check out. Does the Order Extensions template appear as expected? (*Professional Edition only*)
- ◆ Purchase a product and check out, this time opting to ship this product as a gift. Does the Ship To - Gift template appear as expected? (*Professional Edition only*)

Testing Membership and Subscription Templates

(*Professional Edition only*)

Following are some scenarios to test your membership and subscription templates.

If you are setting up a store subscription program:

- ◆ In the storefront, is the link to “Join Now” visible or easily accessible?
- ◆ Click the “Join Now” link, and purchase the store subscription product. Do the templates appear as expected?

If you are setting up a membership area for your store:

- ◆ In the storefront, are the “Logon” and “Logoff” links visible or easily accessible?
- ◆ Register as a new customer. Do all the templates appear as expected?
- ◆ Log in as the customer you just registered as. Do any “members-only” customizations you have made appear?
- ◆ If you have customized your members-only templates, search for a product and view the Catalog List. Then, view the product detail. Do the templates appear as expected?

Testing Customer Service Templates

Your customer service templates are key to caring for your customers after they have purchased products. The following scenarios will help test the design of your customer service templates.

- ◆ Go to the Service page, and click the **Account Information** link. Does the Customer History template appear as expected?
- ◆ Go to the Service page, click the **Lost Passwords** link, and complete the form. You should receive your password via e-mail. Is the e-mail formatted as expected?

If you have selected any of the notification options in **Mail Preferences**:

- ◆ Check out as a new customer, using your own e-mail address. You should receive an order confirmation e-mail. Is the e-mail formatted as expected?
- ◆ In Store Administration, apply a credit to the order. You should receive a credit confirmation. Is the e-mail formatted as expected?
- ◆ In Store Administration, ship the order. You should receive a shipping confirmation. Is the e-mail formatted as expected?

Troubleshooting

This section provides basic troubleshooting tips.

Troubleshooting Validation Errors

Multiple, non-sensical, errors are reported

Situation: You've made a minor change to a template, but the Validator reports many errors, which don't make any sense.

Solution: Check the code prior to the location of the first error for any tag attributes that are missing a quote (for example, there is a starting quote, but not an ending quote).

Troubleshooting Images

Image is Broken

Situation: You're viewing a page with either a broken image, or missing image.

Solution: Check for the following:

- ◆ Ensure that you've uploaded the image as the correct type. Images in your header and footer should be uploaded as "Template Set Images", but product, category and manufacturer images should be uploaded as "Shared Images." See *Uploading Images* (page 50) for more information.
- ◆ Check that you're using the correct code to refer to the image.
 - ◆ **Product Thumbnail:** `<ss:image source="$catalog.images[$product.thumbnailImageName]" />`
 - ◆ **Product Image:** `<ss:image source="$product.photoImage" />`

- ◆ **Manufacturer Image:** `<ss:image source="$catalog.images[$oem.oemLogoName]" />`
- ◆ **Category Image:** `<ss:image source="$listing.category.photoImage" />` (assumes this tag is within an `<ss:foreach item="listing" within="$catalog.categoryList()"></ss:foreach>` tag)
- ◆ **Store Design Image:** `<ss:image source="$store.Images['imagename.ext']">`

Old image still appears after uploading a new one

Situation: You've uploaded a new image that overwrites an existing image used in your store. But when you view the page containing the image, the old image appears.

Solution: Check the following:

- ◆ First, did the image you uploaded have the EXACT same file name as the original? If not, rename the file and upload it again.
- ◆ Next, did you refresh the page after uploading the image? If not, click Refresh. Some browsers may require that you perform a "hard" refresh - please refer to your browser documentation for more information.
- ◆ If you're still having trouble, try clearing your browser's temporary files, close the browser, reopen it and check the page again.

Troubleshooting Pages

Page Appears Broken in Netscape

Situation: You can see a page in Internet Explorer, but not in Netscape.

Solution: Chances are you are missing a `</td>` tag somewhere in your code. Review the template to ensure that all `<td>` tags have matching `</td>` tags. If the template is very complex, you might want to export the template and use an HTML Validator tool to review it.

Tags are Not Resolving on a Page

Situation: When viewing a page in your storefront, you see StoreSense tags (`<ss:value source="$product.price" />` instead of \$30.00).

Solution: Check for the following:

- ◆ Check the tag syntax to ensure that it is spelled correctly. See the online tag help or the Designer's Reference for more information.
- ◆ Ensure that the Business Object member may be used on the page. See the online template help or the Designer's Reference for more information.
- ◆ If you have downgraded from one version of StoreSense to another, you may still have Business Object members in your storefront that only work in the edition you are no longer using. Refer to the Designer's Reference to ensure that the tag may be used by your store.
- ◆ If you are editing your templates in an external editor, check the source code to see that it is not "escaping" the tags - for example, changing `<ss:link/>` to `<ss:link/>`.

- ◆ Did you type the tag directly in Design Mode? You can tell if you did by viewing the **Source Mode** tab, and seeing if the tag's opening and closing symbols have been escaped - for example, changing "<ss:link/>" to "<ss:link/>". If you did, delete the tag, then add it using the Tag element.

The Spacing at the Bottom of the Catalog List is Too Wide in Netscape

Situation: You selected the "3 column" catalog list layout in the Setup Wizard, and sometimes, the spacing at the bottom of the Catalog List template is too wide.

Solution: This is because the 3 column layout requires that your search results "Max Per Page" setting be a multiple of 3. To change this, go to Store Manager | General Preferences.

Object is not resolving on the storefront

Situation: No errors are received when validating, but the object is not resolving on the storefront.

Solution: Check the object on the template to ensure that a "\$" is preceding it.

Troubleshooting Search Results

Products Appear Multiple Times in Search Results

Situation: When searching, one product will appear multiple times in the search results, in different categories.

This occurs when the following conditions are met:

- ◆ A product is assigned to multiple categories.
- ◆ The Catalog List template is sorted by category.
- ◆ The customer performs a search that returns multiple categories that contain this product.

Solution: This behavior is by design.

You can avoid this behavior by editing the Catalog List template, and sorting it by product instead. Doing so will display all products alphabetically, with no category headings.

However, this may not be an appropriate solution if your customers are searching for products using the **Category** search option. In this situation, it is recommended that you create custom Category List templates that are sorted by category. See *Importing and Adding Templates* (page 37) for more information.



Appendix A: Glossary

alt attributes

An attribute and value added to an `<img>` or `<ss:image>` tag that describes the image (and can be viewed by a visitor when they “mouse over” the image). For example, `<ss:image source="$product.photoImage" alt="$product.description"/>` will display the product description when the visitor mouses over the product’s photo image.

attribute

A property or characteristic of a tag. Most tags require at least one attribute. For example, in `<ss:image source="$product.photoImage"/>`, *source* is an attribute of the `<ss:image/>` tag.

background color

A color that fills an area (either the entire web page area, or just a portion of it).

background pattern

An image (GIF or JPEG) that fills an area (either the entire web page area, or just a portion of it).

baseline templates

Saves a copy of all the templates in a template set.

browser

An application that allows you to look at and interact with the World Wide Web. Examples of browsers include Microsoft Internet Explorer and Netscape Navigator.

cart

Contains any items the shopper has expressed interest in purchasing.

catalog

Contains all product information, and can be searched by the shopper.

cookie

Information that a web site puts on a user’s local computer so that it can remember something about them at a later time. Typically, a cookie records preferences when using a particular site.

css (Cascading Style Sheets)

Provides the formatting definitions (such as font, colors, etc) for a web page.

customer

A person who has shopped at your store.

domain name

Locates an organization or other entity on the Internet. For example, `http://www.mystore.com/`.

e-commerce

The buying and selling of goods and services on the Internet.

encryption

The conversion of data into a form (cipher text), that cannot be easily understood by unauthorized people.

ftp (File Transfer Protocol)

A method of exchanging files over the internet. Often used to upload images and files from a local machine to a storefront.

gif

One of the two most common file formats for graphic images on the World Wide Web. The other is the JPEG.

handlebars

Small boxes that surround a selected object.

hexadecimal colors

A system of identifying colors that uses a six-character combination of numbers and letters. For example, white is represented by “#FFFFFF”, black is represented by “#000000” and silver is represented by “#c0c0c0”.

html (Hypertext Markup Language)

The set of code and symbols used to tell a browser how to display a page on the World Wide Web.

jpeg

One of the two most common file formats for graphic images on the World Wide Web. The other is the GIF.

merchant

The store owner.

merchant account

An account that allows the store to accept credit cards as a method of payment.

meta tag

An HTML tag that describes some aspects of the contents of a Web page. Typically used by search engines to help potential visitors find your site.

multimedia

The combination of text, sound, and/or motion video. For example, music and Flash files are considered multimedia.

object expression

An SXML object expression defines where data is coming from, or the data to which input will be posted. Object expressions always start with the “\$” character. For example, in `<ss:image source="$product.photoImage" />`, `$product.photoImage` is the object expression which points to a product's photo image.

pixel

The basic unit of programmable color on a computer display or in a computer image. The physical size of a pixel depends on how the resolution has been set for the display screen.

publish templates

The act of applying any changes to a template to your “live” storefront.

reset templates

The act of reverting your templates to the last saved baseline.

shopping cart

See “cart”.

spider

A program that visits Web sites and reads their pages and other information in order to create entries for a search engine index.

splash page

A page that greets your customers before they go to your main storefront. Splash pages can also serve as the page that “spiders” can catalog (although not all spiders will catalog splash pages).

ssl (Secure Sockets Layer)

SSL is a Secure Sockets Layer that provides security in transferring data over the Internet, forming a layer of encryption between the server and client. This is particularly important in a commerce situation, as the shopper is providing merchants with their personal information and often with their credit card number.

ssml (StoreSense Markup Language)

A combination of tags and object expressions that allow you to work with information from a store.

url

The “address” of a web page. For example, both <http://www.mystore.com/> and <http://www.stores.com/mystore/> are URLs.

visitor

Someone who has browsed your store, but has not necessarily purchased anything.

wysiwig

“What you see is what you get”. WYSIWIG editing allows you to edit web pages while having an idea of how they’ll look. Source code editing only allows you to see and work with the code behind a web page. StoreSense Design Studio offers a WYSIWIG editing mode



Appendix B: Template Reference

This appendix provides information on each of the templates provided by StoreSense. Additional user-defined templates can be added to a template set using the Design Studio in StoreSense Store Administration.

When working with SSML, templates can be linked to using the TemplateSet Business Object. System defined templates can be linked to using `<ss:link source="$templateSet.cart"/>`. Custom templates can be linked to using `<ss:link source="$templateSet.Templates['templateshortname']"/>`.

About Us

All

General information about your store.

Short Name : about

Link to using: `<ss:link source="$templateSet.aboutUs"/>`

Auction Outcome Email

All

E-mail message that notifies the customer when they have won an auction, and includes a link that they must click to begin the checkout process.

Short Name : auctionwon-email

Link to using: Cannot link directly to this template

Billing Options

All

Available payment methods for the store.

If you use a payment processor that requires certain information (such as a customer phone number), then tags that allow you to collect that information can be added to this template. Also, if you do not want to collect credit card information (but want to allow customers to use credit cards and send their credit information separately), make sure the **Require customer to provide credit card information** check box is not selected on the Store Manager | Billing Preferences page.

Short Name : billing

Link to using: Cannot link directly to this template

Cart

All

Customer shopping cart.

If you want the customer to be able to modify the contents of their cart, you must include the `<ss:form shortcut="CartUpdate"/>` tag somewhere on the detail line. This is the default. If you do not, the `<ss:edit source="$detail.quantity" name="qty"/>` tag will not allow text to be entered.

Short Name : cart

Link to using: `<ss:link source="$templateSet.cart"/>`

Cart Add

All

Quantity, size, color and other product purchase options, displayed in the Catalog Detail template (at the location of the `<ss:include template="cartadd"/>` tag).

Short Name : cartadd

Link to using: Cannot link directly to this template

Cart Save

Pro

Saves the selected shopping cart.

Short Name : cartsave

Link to using: `<ss:link source="$templateSet.cartSave"/>`

Cart Select

Pro

Displays the customer's saved shopping carts.

Short Name : cartselect

Link to using: `<ss:link source="$templateSet.cartSelect"/>`

Catalog

All

Product search page.

The default number of products to be returned is set on the Store Manager | General Preferences page.

Short Name : catalog

Link to using: `<ss:link source="$templateSet.catalog"/>`

Catalog - Members Only

Pro

Product search page for logged on members.

Short Name : cataloga

Link to using: `<ss:link source="$templateSet.searchMembersOnly"/>`

Catalog Detail

All

Detailed product information.

Short Name : catalogdetail

Link to using: `<ss:link source="$templateSet.productDetail"/>`

Catalog Detail - Members Only

Pro

Detailed product information for logged on members.

The members only catalog detail is identical in function to the standard catalog detail, except that the customer may be required to provide a valid log on ID.

Short Name : catalogdetaila

Link to using: `<ss:link source="$templateSet.productDetail"/>`

Catalog Detail - Related Products

Pro

Product that have been attached to this product using the Related Products feature. This template is displayed within the Catalog Detail template (at the location of the `<ss:include template="catalogrelated"/>` tag).

Short Name : catalogrelated

Link to using: Cannot link directly to this template

Catalog List

All

Search results list.

If you are using either the 2 or 3 column layout, you may need to change your Max Per Page setting on the Store Manager | General Preferences page. The Max Per Page setting should be a multiple of the number of columns you are using in the layout (for example, if you are using the 3 column layout, the setting should be 6, 9, 12, etc.). In addition, there are instructions in the template source code itself to assist you in changing the number of columns in the layout.

Short Name : cataloglist

Link to using: <ss:link source="\$templateSet.searchResults"/>

Catalog List

Pro

Search results list for logged on members.

If you are using either the 2 or 3 column layout, you may need to change your Max Per Page setting on the Store Manager | General Preferences page. The Max Per Page setting should be a multiple of the number of columns you are using in the layout (for example, if you are using the 3 column layout, the setting should be 6, 9, 12, etc.). In addition, there are instructions in the template source code itself to assist you in changing the number of columns in the layout.

Short Name : cataloglista

Link to using: <ss:link source="\$templateSet.searchResults"/>

Catalog List - Not Found

All

Message displayed if no products meet the customer's search criteria.

Short Name : cataloglisteof

Link to using: Cannot link directly to this template

Checkout

All

Starting point of the checkout process. From this page, customers can select a shipping option, existing customers can log in, and new customers can register.

In order to checkout as a returning customer, they must provide their e-mail address and password.

Short Name : checkoutpro

Link to using: <ss:link source="\$templateSet.checkout"/>

Checkout Deadend

All

Message displayed if the customer's credit card does not meet your billing requirements.

Specifically, it will be displayed if your store only accepts credit orders from customers with a billing address from the store's home country; the customer has entered a different country than the store's home country; and there are no other payment options available for the customer (such as check, COD, etc.)

Short Name : checkout-deadend

Link to using: Cannot link directly to this template

Checkout New Customers

All

Gathers new customer information.

Short Name : checkout

Link to using: <ss:link source="\$templateSet.checkoutRegistration"/>

Closed

All

Message displayed if the store is closed.

Short Name : closed

Link to using: Cannot link directly to this template

County Tax

All

Allows customers to select the county in which they live, if the customer's zip/postal code spans more than one county (and therefore, more than one county tax zone).

Short Name : countytax

Link to using: Cannot link directly to this template

Credit

All

Credit detail.

Short Name : credit

Link to using: Cannot link directly to this template

Credit - Email

Pro

Confirms a customer credit by e-mail.

Short Name : credit-email

Link to using: Cannot link directly to this template

Customer History

Pro

Customer transaction history.

Short Name : history

Link to using: <ss:link source="\$templateSet.customerHistory"/>

Customer Properties

Pro

Displays current customer information, and allows the customer to update that information.

Short Name : customer

Link to using: <ss:link source="\$templateSet.customerProfile"/>

Customer Service

All

Customer service information and links.

Use this template to provide a link to the customer's order history and lost password pages. You can also allow a customer to look up an order's status via the tracking number. Remember to display the tracking number in the Order Confirm template.

Short Name : service

Link to using: <ss:link source="\$templateSet.customerService"/>

Customer Sign In

Pro

Log in form for previous customers who wish to view their account history.

All fields are required.

Short Name : signin

Link to using: Cannot link directly to this template

Customer Subscription

Pro

Information regarding the selected subscription.

Short Name : subscription

Link to using: <ss:link source="\$templateSet.subscribe"/>

Download List

Pro

Displays the list of items that are available for download.

Short Name : downloadlist

Link to using: <ss:link source="\$templateSet.downloadList"/>

Email A Friend

All

Allows a visitor to e-mail a link to your store, or to a product detail page, to a friend.

Short Name : emailfriend

Link to using: <ss:link source="\$templateSet.emailAFriend"/>

Email A Friend Email Content

All

The e-mail containing a link to your store sent by a visitor to a friend.

Short Name : emailfriend-email

Link to using: Cannot link directly to this template

Email A Friend Result

All

Displays the information that was entered on the Email a Friend template.

Short Name : emailfriendresult

Link to using: Cannot link directly to this template

Footer

All

Defines the overall page layout. This template works in conjunction with the Header template.

You can achieve many page layouts with the Header and Footer templates. These are used on every page of the store. All other templates are displayed within the area defined by these templates.

In addition, this template will *not* display on any template that begins with an <HTML> tag. This allows you to create custom designs on individual templates.

Short Name : footer

Link to using: Cannot link directly to this template

Frequently Asked Questions

All

Answers to frequently asked questions.

Use this template to improve the level of customer service to your customers. Also, if your business is getting the same question repeatedly from your customers, you should add it to this template.

If this template does not appear, you must add it by either using the Setup Wizard or Template Assistant.

Short Name : faq

Link to using: <ss:link source="\$templateSet.faq"/>

Generated Account Password - Email

All

E-mail that provides a customer with the password that StoreSense automatically generated for them.

This template will only be sent if a customer tries to log on or check out using a customer record that does not have a password. StoreSense will generate the password and send it to the customer.

Short Name : genpass-email

Link to using: Cannot link directly to this template

Header

All

Defines the overall page layout. This template works in conjunction with the Footer template.

You can achieve many page layouts with the Header and Footer templates. These are used on every page of the store. All other templates are displayed within the area defined by these templates.

In addition, this template will *not* display on any template that begins with an <HTML> tag. This allows you to create custom designs on individual templates.

Short Name : header

Link to using: Cannot link directly to this template

Invoice

All

Order invoice.

If you have upgraded from a prior version of StoreSense, and are using a shipper other than/in addition to the StoreSense Shipper, you will need to replace the line containing the shipping information with the `<ss:include template="selectshipping"/>` tag.

This template can also be viewed in Store Administration.

If you use the UPS shipper, StoreSense displays the UPS Trademark and accompanying disclaimer on each invoice and shipping related screen viewed by your customers. These elements are permanent and may not be modified or deleted. If these fields do not appear, add the If these fields do not appear, add the `<ss:include macro="ShipperCopyright"/>` to this template.

Short Name : invoice

Link to using: Cannot link directly to this template

Invoice - Email

All

Order invoice that can be e-mailed to a customer.

If you use the UPS shipper, StoreSense displays the UPS Trademark and accompanying disclaimer on each invoice and shipping related screen viewed by your customers. These elements are permanent and may not be modified or deleted. If these fields do not appear, add the If these fields do not appear, add the `<ss:include macro="ShipperCopyright"/>` to this template.

Short Name : invoice-email

Link to using: Cannot link directly to this template

Invoice - Print

All

Printable order invoice (does not contain the header or footer).

This template is used only in Store Administration to print an invoice.

If you use the UPS shipper, StoreSense displays the UPS Trademark and accompanying disclaimer on each invoice and shipping related screen viewed by your customers. These elements are permanent and may not be modified or deleted. If these fields do not appear, add the `<ss:include macro="ShipperCopyright" />` to this template.

Short Name : invoice-print

Link to using: Cannot link directly to this template

Lost Password

All

Verifies customer information so that password information may be e-mailed to them.

If the first and last name fields are present on this template, they will be required entry fields for the customer. If you delete the fields, the password will be sent to the customer based on the e-mail address alone.

Short Name : lostpassword

Link to using: `<ss:link source="$templateSet.lostPassword"/>`

Lost Password - Email

All

E-mail providing the user name and password to a customer who has lost their information.

Short Name : lostpass-email

Link to using: Cannot link directly to this template

Low Inventory - Email

Pro

E-mail that will be sent to the Store Buyer if Buyer Notifications are enabled, and inventory for a product is below the product's threshold.

Short Name : lowinv-email

Link to using: Cannot link directly to this template

Member Login

Pro

Login form for registered members.

A valid e-mail and password is required. Customers obtain passwords by either purchasing a product from the store, or by registering with the store.

All fields are required.

Short Name : memberlogin

Link to using: <ss:link source="\$templateSet.logon"/>

Member Login Error

Pro

Message displayed if member login information is incorrect.

Short Name : memberloginerror

Link to using: Cannot link directly to this template

Message

All

Miscellaneous messages that are presented to the customer.

These include when a required form field is not entered, when an order is cancelled, and when any other error condition occurs.

Short Name : message

Link to using: Cannot link directly to this template

Order Confirm

All

Order confirmation page.

Short Name : orderconfirm

Link to using: Cannot link directly to this template

Order Extensions

Pro

Gathers custom product information on a per product basis.

If this template is defined for a product in Store Administration, Product, it will be shown once for each product purchased. This includes once for each different product multiplied by the quantity ordered.

This template can also be viewed in Store Administration.

Short Name : orderext

Link to using: Cannot link directly to this template

Packing Slip

All

Packing slip for an order, displayed without the header or footer.

This template is used only in Store Administration - it is not visible in the storefront.

Short Name : packing

Link to using: Cannot link directly to this template

PayPal Purchase Token

All

Begins the PayPal payment process. This template is displayed within the Order Confirm template (at the location of the `<ss:include template="paypal-purchtoken" />` tag).

This template conforms to Pay Pal's payment processing requirements. Changes should not be made unless instructed otherwise by Pay Pal, or your Site Administrator.

Short Name : paypal-purchtoken

Link to using: Cannot link directly to this template

Policies

All

Store policies, such as shipping policies, returns and credits.

Short Name : policy

Link to using: `<ss:link source="$templateSet.storePolicies"/>`

Privacy Statement

All

Store privacy statement.

Privacy is a very important issue for customers shopping online. Some countries have strict privacy requirements for those business that wish to sell to their citizens. You are strongly encouraged to familiarize yourself with the privacy requirements of the countries in which you will do business.

The text shipped with StoreSense is only intended to serve as sample text.

If this template does not appear, you must add it by either using the Setup Wizard or Template Assistant.

Short Name : privacy

Link to using: `<ss:link source="$templateSet.privacyPolicy"/>`

Product Authorization

Pro

Message displayed if the customer is not authorized to purchase one or more products in their cart.

The authorization check occurs during checkout. Authorization is granted to a customer by selecting the **Authorization** check box in the Product Profile.

Short Name : productauth

Link to using: Cannot link directly to this template

Promotion Alert

All

Notifies the shopper if they are eligible for other promotions that would benefit them.

Short Name : promotion-alert

Link to using: Cannot link directly to this template

Purchase Order Limit

Pro

Message displayed if an order exceeds the customer's purchase order limit (defined in their Customer Profile).

You may want to include a link to your Service page or a specific way to contact the person for approving order limits.

Short Name : polimit

Link to using: Cannot link directly to this template

Recurring Bill Error - Email

Pro

Notifies staff (using the staff e-mail address), that an error occurred when billing a subscription.

Short Name : recureerror-email

Link to using: Cannot link directly to this template

Recurring Billing - Email

Pro

E-mail sent when a subscription is billed to the customer.

Short Name : recurbill-email

Link to using: Cannot link directly to this template

Registration

Pro

Begins the store registration process.

Short Name : register

Link to using: <ss:link source="\$templateSet.customerRegistration"/>

Registration Confirmation

Pro

Confirms that the customer has registered with the store.

Short Name : regconfirm

Link to using: Cannot link directly to this template

Sales Affiliate

Sales Manager

Registers a sales affiliate.

Sales Affiliates are other businesses, individuals or organizations with which you agree to pay a small percentage of the item's sale price if they refer a customer to your store.

Short Name : affiliate

Link to using: <ss:link source="\$templateSet.affiliate"/>

Sales Affiliate Confirm

Sales Manager

Confirms the registration of a sales affiliate.

Sales Affiliates are other businesses, individuals or organizations with which you agree to pay a small percentage of the item's sale price if they refer a customer to your store.

Short Name : affilconfirm

Link to using: Cannot link directly to this template

Sales Affiliate Email

Sales Manager

Confirms the registration of a sales affiliate by e-mail.

Short Name : affiliate-email

Link to using: Cannot link directly to this template

Sales Lead

Sales Manager

Gathers sales lead information from a customer.

Sales Leads are generated when a Product's Type field is set to "Lead".

Short Name : lead

Link to using: Cannot link directly to this template

Sales Lead Confirmation

Sales Manager

Confirms the sales lead.

Sales Leads are generated when a customer wishes to purchase a product that is not available online.

Short Name : leadconfirm

Link to using: Cannot link directly to this template

Select Shipping

All

Shipping quotes list, displayed within the Invoice template.

Short Name : selectshipping

Link to using: Cannot link directly to this template

Ship To

All

Collects shipping information if different from the customer's billing address.

This template is used if the Gift option is NOT selected on the Store Manager | Promotions page. Otherwise, the Ship To Gift template is used.

Short Name : shipto

Link to using: Cannot link directly to this template

Ship To Gift

Pro

Collects shipping and gift wrap information, if gift wrap is enabled for the store.

Gift Shipments must be enabled on the Store Manager | Promotions page for this template to appear.

The `<ss:value source="$invoice.message"/>` tag can be shown on the Invoice and Packing Slip.

Short Name : shiptogift

Link to using: Cannot link directly to this template

Shipping - Email

Pro

E-mail sent when an order is shipped.

Short Name : shipping-email

Link to using: Cannot link directly to this template

Store Locator

All

Information regarding the physical locations of your stores.

If this template does not appear, you must add it by either using the Setup Wizard or Template Assistant.

Short Name : locator

Link to using: `<ss:link source="$templateSet.storeLocations"/>`

Storefront

All

Store home page.

Short Name : storefront

Link to using: `<ss:link source="$templateSet.homePage"/>`

Subscription

Pro

Starting page in the store subscription process.

This template appears when the customer clicks the `<ss:link source="$templateSet.subscribe">Become a Member</ss:link>` tag in a store that has a subscription product based members only area.

Short Name : subscribe

Link to using: Cannot link directly to this template

Subscription Billing Option

Pro

Billing information for the selected subscription.

Short Name : recurbilling

Link to using: Cannot link directly to this template

Subscription Cancellation - Email

Pro

E-mail sent to customers when a subscription is cancelled.

Short Name : subscancel-email

Link to using: Cannot link directly to this template

Subscription Renewal - Email

Pro

E-mail sent to customers when a subscription is renewed.

Short Name : subsrenew-email

Link to using: Cannot link directly to this template

Supplier EDI

Supplier Connect

Notifies suppliers via download of orders that have been placed for their products.

This template contains Header (“H”) and Detail (“D”) records, which are required for proper downloading. The header record contains the customer’s number, invoice number, and order date. The detail records contain the customer’s number, invoice number, orderdate, product SKU, and product quantity. The resulting file is comma delimited with each string field enclosed in double quotes.

Short Name : supplier1

Link to using: Cannot link directly to this template

Supplier EDI 850

Supplier Connect

Notifies suppliers via ANSI ASC X12 download of orders that have been placed for their products.

The template is in the ANSI ASC X12 (American National Standards Institute Accredited Standards Committee X12, form 850 (Purchase Order) format. Changes should be made carefully, making sure that any changes conform to those standards.

By default, this template supports one “segment”. If you modify the template to support more than one segment, you will need to modify the parameter in the "`<ss:eval source=\"$detailCount.set($detailCount.multiply(1))\" />` tag to reflect the number of segments.

Short Name : supplier.x12.850

Link to using: Cannot link directly to this template

Supplier Email

Supplier Connect

Notifies suppliers via e-mail of orders that have been placed for their products.

Short Name : supplier3

Link to using: Cannot link directly to this template

Supplier Fax

Supplier Connect

Notifies suppliers by fax of orders that have been placed for their products.

Short Name : supplier2

Link to using: Cannot link directly to this template

Tax Region Select

All

Allows the customer to preview product pricing inclusive of applicable sales tax, and can be displayed within the Cart, Catalog Detail or Catalog List templates, at the location of the "`<ss:include source="taxregionselect" />` tag.

Your store must be configured to show prices inclusive of sales tax on the Sales Tax Preferences page in Store Administration before you can use this template.

Short Name : taxregionselect

Link to using: Cannot link directly to this template

Tracking

All

Package tracking information. Only available using a shipper other than/in addition to the StoreSense Shipper.

If you use the UPS shipper, StoreSense displays the UPS Trademark and accompanying disclaimer on each invoice and shipping related screen viewed by your customers. These elements are permanent and may not be modified or deleted. If these fields do not appear, add the If these fields do not appear, add the `<ss:include macro="ShipperCopyright"/>` to this template.

Short Name : tracking

Link to using: `<ss:link source="$templateSet.shipmentTracking"/>`

User Defined

All

User Defined templates are those templates that are imported/added with a type of "User Defined".

Short Name : userdefined

Link to using: Cannot link directly to this template

WorldPay Callback Processing Error

All

Message displayed when an error occurs for an order paid for with WorldPay.

This template will only be used if you have set up a Callback URL at your WorldPay account. If you have not done so, then the default page from WorldPay will be displayed instead. Tags that begin `<WP*>` are WorldPay tags and should not be edited. They may, however, be moved. In addition, WorldPay requires certain HTML restrictions as well. Instructions regarding these tags and restrictions are available from WorldPay.

Short Name : wpay-error

Link to using: Cannot link directly to this template

WorldPay Payment Cancellation

All

Message displayed when an order paid for with WorldPay has been cancelled.

Tags that begin `<WP*>` are WorldPay tags and should not be edited. They may, however, be moved. In addition, WorldPay requires certain HTML restrictions as well. Instructions regarding these tags and restrictions are available from WorldPay. This template will only be used if you have set up a Callback URL at your WorldPay account. If you have not done so, then the default page from WorldPay will be displayed instead.

Short Name : wpay-cancel

Link to using: Cannot link directly to this template

WorldPay Payment Confirmation

All

Message displayed when an order paid for with WorldPay has been confirmed.

Tags that begin <WP*> are WorldPay tags and should not be edited. They may, however, be moved. In addition, WorldPay requires certain HTML restrictions as well. Instructions regarding these tags and restrictions are available from WorldPay. This template will only be used if you have set up a Callback URL at your WorldPay account. If you have not done so, then the default page from WorldPay will be displayed instead.

Short Name : wpay-confirm

Link to using: Cannot link directly to this template

WorldPay Purchase Token

All

Begins the WorldPay payment process. This template is displayed within the Order Confirm template.

This template conforms to WorldPay's payment processing requirements changes should not be made unless instructed otherwise by WorldPay, or your Site Administrator.

Short Name : wpay-purchase

Link to using: Cannot link directly to this template



Appendix C: Tag Reference

This reference describes the SSML Tags available to use in the storefront, with samples.

The following is an example of how SSML tags are constructed:

<code><ss:edit</code>	<code>source="</code>	<code>\$customer.FirstName</code>	<code>"</code>	<code>size="20"</code>	<code>></code>
(1)	(2)	(4)	(3)	(4)	(6)

<code><ss:link</code>	<code>source="</code>	<code>\$templateSet.aboutUs</code>	<code>"</code>	<code>>About Us</code>	<code></ss:link></code>
(1)	(2)	(4)	(3)	(5)	(6)

- 1: `<ss:/>` defines this as a StoreSense tag.
- 2: The tag type. Generally indicates what this tag will resolve to on the storefront.
- 3: The Object Expression. Defines where the data is coming from, or the data to which input will be posted. Object expressions always start with the "\$" character.
- 4: Tag attributes that further define the tag.
- 5: The content to appear between the opening and closing tags. Not all tags have content.
- 6: The closing slash or tag.

<ss:break>

Allows you to "break" out of an `<ss:foreach>` loop.

Examples

To display only the first 10 top-level categories on the header, use:

```
<ss:foreach item="category" within="$catalog.categoriesToDepth(1)">
  <ss:link source="$category" /><br>
  <ss:if test="$counter.currentRow==10">
    <ss:break/>
  </ss:if>
</ss:foreach>
```

<ss:button>

Creates a button for a form (either as a standard form button, or as an image button).

Examples

To add a Submit button to a form, use:

```
<ss:button type="submit" value="Submit" />
```

...which resolves to:

```
<input type="submit" value="Submit">
```

To add a Submit button to a form that appears as an image, use:

```
<ss:button source="$templateSet.images['go.gif']" alt="Search" border="0"/>
```

...which resolves to:

```
<input type='image' src="/.../go.gif" border="0" alt="Search">
```

Attributes

Supports HTML attributes

source

Use this attribute to point to an image object expression when creating an image button.

type

The type of button. Only use this attribute to override the type attribute that StoreSense generates.

<ss:checkbox>

Creates a checkbox for a form.

Examples

To allow the shopper to add the gift wrap service to their order, use:

```
<ss:checkbox source="$invoice.giftWrapEnabled"/>
```

...which resolves to:

```
<input type="checkbox" name="$invoice.giftWrapEnabled" value="1"/>
```

Attributes

Supports HTML attributes

checked

Use this attribute to select the check box by default.

name

The form control name. Should only be used if the name is different than the source attribute.

shortcut

Shortcut attributes allow you to replace a large, complex piece of code with one word. All formatting and text are included within the shortcut. If you want to control formatting or text, replace the shortcut with the corresponding code (for assistance, try viewing the HTML source of the storefront page).

Valid Values

Value	Description
ChangeBilling	Creates a check box that allows the shopper to indicate that they wish to change their billing information.
ChangeShippingAddress	Creates a check box that allows the shopper to indicate they wish to ship the order to a different address.

source

The object expression in which the check box value will be stored.

value

The value to be applied if the check box is selected.

<ss:choose>

The <ss:choose> tag displays content from one among a number of possible alternatives. It consists of a sequence of <ss:when> elements, followed by an optional <ss:otherwise> element (which will be applied when *none* of the <ss:when/> conditions are met).

Examples

To apply special formatting to the “shipped” status on an invoice, use:

```
<ss:choose>
  <ss:when test="$detail.status == 'Shipped'">
    <b>This order has been shipped</b>
  </ss:when>
  <ss:otherwise>
    <ss:value source="$detail.status" />
  </ss:otherwise>
</ss:choose>
```

...which resolves to:

```
<b>This order has been shipped</b>
```

...if shipped. If the order has not been shipped, it displays the current order status (without bold formatting).

To display a list of categories and subcategories, use:

```
<ss:choose>
  <ss:when test="$category.TreeLevel == '2'">
    --<ss:value source="$category.name" /><br>
  </ss:when>
  <ss:when test="$category.TreeLevel == '3'">
    ----<ss:value source="$category.name" /><br>
  </ss:when>
  <ss:otherwise>
    <ss:value source="$category.name" /><br>
  </ss:otherwise>
</ss:choose>
```

...which resolves to:

```
Root Category 1<br>
--Subcategory 1<br>
--Subcategory 2<br>
----Subcategory 3<br>
Root Category 2<br>
...
```

This code would appear within an `<ss:foreach/>` tag that iterates through the list of categories. First, this code checks the level of the current category. If it is at level '2', it displays the category name with two dashes preceding it. If it is at level '3', it displays the category name with four dashes preceding it. Otherwise, it just displays the category name. Category levels correspond to the level of nesting. For example, a top-level category has a level of 1, any subcategories assigned to it have a level of 2, any subcategories assigned to the level 2 subcategories have a level of 3, and so on.

<ss:comment>

Contains comments that will be removed when the template is displayed but remain in the template source. Standard HTML comment tags (`<!-- -->`) *are not* removed when the template is displayed and are therefore accessible in the page source. If you wish to "hide" any content or code containing StoreSense tags, use this tag instead of HTML comments.

Examples

To display a note that cannot be seen when viewing page source, use:

```
<ss:comment>Note Text Here</ss:comment>
```

The code surrounded by the tags will not appear.

<ss:directive>

Communicates specific information to StoreSense.

Examples

To display custom keywords in the page's `<meta>` tag. The `<meta>` tag should be placed in the header, while the `<ss:directive/>` tag should be placed on the body template., use:

```
On the Header template:
<meta name="Keywords" content="$page.getDirective('page.meta.keywords') ">

On the storefront template:
<ss:directive name="page.meta.keywords" value="custom, keywords"/>
```

The value is set to '5'.

Attributes

name

The item to which information is being communicated

value

The information to be communicated

<ss:edit>

Creates a text input field for a form. If an HTML size attribute is not declared, the default size will be “10”. StoreSense adds a “maxlength” attribute, which defines the maximum number of characters allowed for this field.

Examples

To add a field to collect the customer’s first name, use:

```
<ss:edit source="$customer.firstName" size="20">
```

...which resolves to:

```
<input type="text" name="$customer.firstName" value="" maxlength="20" size="20">
```

Attributes

Supports HTML attributes

default

The default value to display in the edit field.

format

Defines the formatting characteristics for this tag when resolving the “source” attribute.

Valid Values

Value	Description
date.short	Date
date.medium	Date
date.full	Date
date.long	Date
time.short	Time
time.medium	Time
time.full	Time
time.long	Time
time24	Time
day	Day
month	Month
year	Year
percent	Percentage
%	Percentage
%N	Percentage, with the number of decimal places specified by ‘N’
truefalse	True or False
yesno	Yes or No
none	No formatting
money	Currency

maxlength

The maximum number of characters that may be entered. If this attribute is not specified, the value defined by StoreSense will be used.

name

The control name. Should only be used if the name is different than the source attribute.

source (Required)

The object expression in which the edit value will be stored.

value

The default text to appear in the field.

<ss:else>

See the <ss:if> tag for information.

<ss:emailheader>

Contains text to be used in an e-mail message header (such as the subject line).

Examples

To define a title to appear on an e-mail's subject line, use:

```
<ss:emailheader>
  Insert Title Text Here
</ss:emailheader>
```

...which resolves to:

```
Insert Title Text Here
```

... in the subject line of the e-mail message.

Attributes

type

Indicates if this is a text or HTML e-mail message.

Valid Values

Value	Description
text/plain	Indicates that this is a standard text e-mail message.
text/html	Indicates that this is an HTML format e-mail message. Not all e-mail clients support HTML e-mail. To provide both a text and HTML version of the e-mail, enter the text version at the very beginning, between embedded within <!-- and --> tags.

<ss:eval>

Executes the value in the source attribute, but does not display it.

Examples

When assigning a value to the math object, use:

```
<ss:eval source="$math.set(5)"/>
```

The value is set to '5'.

Attributes

source (Required)

The object expression that is to be executed

<ss:font>

Creates HTML font tags that use the values defined in the Setup Wizard and Template Assistant to format the storefront templates.

Examples

To apply all the heading font attributes, use:

```
<ss:font source="$font.heading">
```

...which resolves to:

```
<font face="arial,helvetica,sans-serif" color="black" size="4">
```

The tag resolves to the selections made using the Setup Wizard or Template Assistant.

To mix the settings selected during the Setup Wizard with custom font attributes, use:

```
<ss:font face="$font.paragraph" size="$font.paragraph" color="red">
```

...which resolves to:

```
<font face="arial,helvetica,sans-serif" size="2" color="red">
```

Attributes

Supports HTML attributes

source

The object expression containing the font property.

face

The object expression containing the font face property.

color

The object expression containing the font color property.

size

The object expression containing the font size property.

<ss:foreach>

Iterates over the items in a collection. For example, use this tag (in conjunction with others) to display a list of products. Or, you could use this tag to create a set of links to each of your product categories. In addition, this tag supports the <ss:sortby> tag on catalog list templates, which specifies the sort order of the list.

Examples

To display a list of products, with no sorting, use:

```
<ss:foreach item="listing" within="$catalog.results">
  <ss:foreach item="product" within="$listing.products">
    <p><ss:value source="$product.name" /></p>
  </ss:foreach>
</ss:foreach>
```

...which resolves to:

```
<p>Product 1</p>
<p>Product 2</p>
<p>...</p>
```

The first <ss:foreach/> creates the list of search results. Then, the second <ss:foreach> displays that list by product.

To display a list of products, grouped by category, use:

```
<ss:foreach item="listing" within="$catalog.results">
  <ss:sortby item="category" direction="asc" />

  <h1><ss:value source="$category.name" /></h1>

  <ss:foreach item="product" within="$listing.products">
    <ss:sortby item="product" direction="asc" />

    <p><ss:value source="$product.name" /></p>

  </ss:foreach>
</ss:foreach>
```

...which resolves to:

```
<h1>Category A Name</h1>
  <p>Product 1</p>
  <p>Product 2</p>
  <p>...</p>
<h1>Category B Name</h1>
  <p>Product 1</p>
  <p>Product 2</p>
  <p>...</p>
<h1>...</h1>
```

In this example, the first <ss:foreach> creates the list of products, then sorts them by category name. The next <ss:foreach> creates the list of products within each category, and sorts them by product name.

To display the prices and names for all of a product's attributes formatted as a bulleted list, use:

```
<ul>
  <ss:foreach item="attr" within="$product.attributes">
    <li><ss:value source="$attr.price"/>, <ss:value
source="$attr.description"/></li>
  </ss:foreach>
</ul>
```

...which resolves to:

```
<ul>
  <li>$85.95 Full-Size</li>
  <li>$19.95 Miniature</li>
</ul>
```

Attributes

item

Specifies the name to be used when referencing object expressions within the collection.

within

Specifies the collection or iterator from which items will be listed. For example, when creating a list of categories, the *category* is the “item”, and *CategorySearch* is the “collection”

<ss:form>

Creates a complete HTML form, as specified by the *shortcut* attribute.

Examples

To allow the customer to search for an order by entering a tracking number, use:

```
<ss:form shortcut="Tracking"/>
```

...which resolves to:

```
<form name='tracking' method='POST' action='...'>
  Tracking No: <input type='text' name='tracking' value='' size=12 maxlength=12>
  &nbsp;<input type='submit' value='Go!'/>
</form>
```

All formatting for the form is defined by the shortcut.

To create a form that allows users to update the quantity of product in their cart, use:

```
<ss:form shortcut="CartUpdate">
  <ss:edit name="qty" source="$detail.Quantity" size="4"/>
</ss:form>
```

...which resolves to:

```
<form name='cart' method='POST' action='...'>
  <input type='hidden' name='key' value='12/23/02 10:49:39 AM'>
  <input type='hidden' name='product_no' value='2'>
  <input type='text' name='qty' value='1' name='qty' size='4'/>
</form>
```

Attributes

Supports HTML attributes

shortcut (Required)

Shortcut attributes allow you to replace a large, complex piece of code with one word. All formatting and text are included within the shortcut. If you want to control formatting or text, replace the shortcut with the corresponding code (for assistance, try viewing the HTML source of the storefront page).

Valid Values

Value	Description
CartEnd	Inserts the HTML code that ends a form containing a list of items that can be selected and added to a cart. Must be used with a CartStart shortcut.
CartStart	Inserts the HTML code that begins a form containing a list of items that can be selected and added to a cart. Must be used with a CartEnd shortcut.
CartUpdate	Allows the customer to update the quantity of items in their cart.
ChangeSubscriptionBilling	Inserts the button to take the customer to change their subscription billing information.
ChangeSubscriptionShipTo	Inserts the button to take the customer to change their subscription shipping information.
Checkout	Inserts the button to take the customer to the Checkout New Customers template.
CheckoutPro	Inserts the button to take the customer to the Checkout template.
LanguageOverride	Inserts a list from which the customer can choose the language in which they want the store to appear.
Pay	Defines the form that allows the customer to pay an outstanding invoice.
PayPalPurchaseToken	Inserts the button to take the customer to complete a PayPal transaction.
SubmitSubscription	Allows the customer to submit changes to their subscriptions.
Tracking	Inserts a form that allows the customer to search for a specific order.

<ss:hidden>

Creates a hidden field for a form.

Examples

To include the current date in a form, without displaying it to the customer , use:

```
<ss:hidden source="$system.currentDate" name="date"/>
```

...which resolves to:

```
<input type="hidden" name="date" value="01/01/2003">
```

Attributes

name

The control name. Should only be used if the name is different than the source attribute.

format

Defines the formatting characteristics for this tag when resolving the “source” attribute.

Valid Values

Value	Description
date.short	Date
date.medium	Date
date.full	Date
date.long	Date
time.short	Time
time.medium	Time
time.full	Time
time.long	Time
time24	Time
day	Day
month	Month
year	Year
percent	Percentage
%	Percentage
%N	Percentage, with the number of decimal places specified by ‘N’
truefalse	True or False
yesno	Yes or No
none	No formatting
money	Currency

source (Required)

The object expression in which a value will be stored.

value

The default value associated with the field.

<ss:if>

Tests a single condition, described in the *test* attribute. If you need to test more than one condition, the <ss:choose> tag may be more appropriate. Also includes the <ss:else/> tag.

Examples

To display a selection list only if a product has colors defined for it, use:

```
<ss:if test="$product.hasColors()">
  Color: <ss:select source="$product.colors" name="color"/>
</ss:if>
```

...which resolves to:

```
Color: <select name="color">
```

```
<option value="Blue">Blue</option>
<option value="Green">Green</option>
</select>
```

...if the test condition is met. Otherwise, nothing is displayed.

To display the “Save Cart” link only if the customer is logged in *and* has items in their cart, use:

```
<ss:if test="$customer.isLoggedIn() AND $cart.details">
  <ss:link source="$templateSet.cartSave">Save Cart</ss:link>
</ss:if>
```

...which resolves to:

```
<a href='/click/Cart.bok?smode=save'>Save Cart</a>
```

...if the test conditions are both met. Otherwise, nothing will be displayed

To change the formatting of a category, based on its level, use:

```
<ss:if test="$category.treeLevel == '1'">
  <br><ss:value source="$category.name" /><br>
<ss:else/>
  --<ss:value source="$category.name" /><br>
</ss:if>
```

...which resolves to:

```
Root Category 1<br>
```

...if this is a top level category. Otherwise, it will resolve to “--Subcategory 1
”

To display the items a customer has added to their cart, use:

```
<ss:if test="$cart.details">
  <ss:foreach item="detail" within="$cart.details">
    <ss:value source="$detail.product.name" /><br>
  </ss:foreach>
<ss:else/>
  Empty
</ss:if>
```

...which resolves to:

```
Cutie Doll<br>
Fancy Girl Doll<br>
Dining Room Lithograph<br>
```

...if there are products in the cart. Otherwise, it would display “Empty”.

Attributes

test (Required)

Defines the test condition. Includes an Object Expression, operand, and test value. You can test for multiple conditions by using a single “AND” or “OR” statement. AND/OR must either be in all caps, or use the following symbols: “&&” (AND), “||” (OR). Valid operands are:

- ◆ > - Greater Than
- ◆ < - Less Than
- ◆ >= - Greater Than or Equal To
- ◆ <= - Less Than or Equal To

- ◆ = - Equals
- ◆ == - Equals
- ◆ != - Not Equal To
- ◆ <> - Not Equal To
- ◆ MATCHES - Supports wildcard at the end of the string (for example 'string*').

<ss:image>

Includes an image on the template.

Examples

To add a banner to your header template, use:

```
<ss:image source="$store.storeLogo" border="0" />
```

...which resolves to:

```

```

To display the product photo image on the Catalog List template, use:

```
<ss:image source="$product.photoImage" border="0" />
```

...which resolves to:

```

```

Attributes

Supports HTML attributes

source

The object expression that contains the image you wish to display.

<ss:include>

Inserts a template or file into the current template (at the location of this tag).

Examples

To display the Related Products template within the Catalog Detail template, use:

```
<ss:include template="catalogrelated" />
```

The code defined in the Related Products template will appear.

To display a static file on any template, use:

```
<ss:include source="$store.files['filename.ext']" />
```

The code defined in the filename.ext file will appear. This example assumes that you have uploaded filename.ext to your store content area.

To display the product price, conditionally displaying sale prices and saving where applicable, use:

```
<ss:include macro="productprice"/>
```

...which resolves to:

```
Product price, plus any sale information and formatting.
```

While the macro attribute “productprice” will determine if any sale information and formatting needs to be displayed, it will not allow you to apply your own formatting.

Attributes

source

The object expression containing the location of the file you wish to include.

template

The template or file to be included.

Valid Values

Value	Description
catalogrelated	Related Products template.
cartadd	Cart Add template.
paypal-purchtoken	PayPal Purchase Token template.
selectshipping	Select Shipping template.
taxregionselect	Tax Region Select template.
wpay-purchase	World Pay Purchase Token template.
filename	The file name of any user-defined templates.

macro

Code to be included at the location of this tag.

Valid Values

Value	Description
productprice	The product price, with sale price and savings information, if valid. The SSML equivalent of this macro is: <pre><ss:foreach item="productprice" within="\$product.finalPriceList"> <ss:choose> <ss:when test="\$counter.totalRows == 1"> <ss:value source="\$productprice"/> </ss:when> <ss:otherwise> <ss:choose> <ss:when test="\$counter.isLast"> <ss:value source="\$productprice"/> </ss:when> <ss:otherwise> <s><ss:value source="\$productprice"/></s> </ss:otherwise> </ss:choose> </ss:otherwise> </ss:choose> </ss:foreach></pre>

Value	Description
	<pre> </ss:choose> &nbsp; </ss:foreach> <ss:foreach item="promotion" within="\$product.activePromotions"> <ss:set name="detailText" value="\$product.promotionDetailText(\$promotion)"/> <ss:if test="\$detailText != null"> (<ss:value source="\$detailText"/>) </ss:if> </ss:foreach>
 </pre>
attributedetailprice	The attribute detail price, with sale price and savings information, if valid. The SSML equivalent of this macro is: <pre> <ss:if test="\$product.hasAttributes"> <ss:foreach item="attribute" within="\$product.attributes"> <ss:foreach item="price" within="\$attribute.finalPriceList"> <ss:choose> <ss:when test="\$counter.isLast"> <ss:value source="\$price"/> </ss:when> <ss:otherwise> <s><ss:value source="\$price"/></s> </ss:otherwise> </ss:choose> </ss:foreach> <ss:foreach item="promotion" within="\$product.activePromotions"> <ss:set name="detailText" value="\$product.promotionDetailText(\$promotion)"/> <ss:if test="\$detailText != null"> (<ss:value source="\$detailText"/>) </ss:if> </ss:foreach> <ss:value source="\$attribute.attributeText"/>
 </ss:foreach> </ss:if> </pre>
attributedetailfull	The attribute detail price, with sale price and savings information, if valid. The SSML equivalent of this macro is: <pre> <ss:if test="\$product.hasAttributes"> <ss:foreach item="attribute" within="\$product.attributes"> <ss:foreach item="price" within="\$attribute.finalPriceList"> <ss:choose> <ss:when test="\$counter.isLast"> <ss:value source="\$price"/> </ss:when> <ss:otherwise> <s><ss:value source="\$price"/></s> </ss:otherwise> </ss:choose> &nbsp; </ss:foreach> <ss:foreach item="promotion" within="\$product.activePromotions"> <ss:set name="detailText" value="\$product.promotionDetailText(\$promotion)"/> <ss:if test="\$detailText != null"> (<ss:value source="\$detailText"/>) </ss:if> </ss:foreach> <ss:value source="\$attribute.attributeText"/> <ss:if test="\$store.inventoryEnabled"> <ss:if test="\$attribute.quantity > 0"> &nbsp;<ss:value source="\$attribute.quantity"/> in stock <ss:else/> <ss:if test="\$store.backordersnotallowed"> &nbsp;<ss:value source="\$attribute.quantity"/> in stock </ss:if> </ss:if> </ss:if> </ss:foreach> </ss:if> </pre>

Value	Description
	<pre> <ss:else/> &nbsp; BACKORDERD </ss:if> </ss:if> </ss:if>
 </ss:foreach> </ss:if> </pre>
productpricex	See AttributeDetailFull for information regarding this macro.
ShipperCopyright	Information and formatting required by the shipper (such as UPS).

<ss:link>

Creates a link to a template or static page.

Examples

To create a link to the About Us template, use:

```
<ss:link source="$templateSet.aboutUs" />
```

...which resolves to:

```
<a href="../../../page.bok?template=about">About Us</a>
```

To create a link to the Members Only catalog, use:

```
<ss:link source="$templateSet.searchMembersOnly">
  Members Only Catalog
</ss:link>
```

...which resolves to:

```
<a href=/click/Catalog.bok?mo=1">
  Members Only Catalog
</a>
```

To create a link to a custom template, use:

```
<ss:link source="$templateSet.templates['shortname']. ">
  Insert Text Here
</ss:link>
```

...which resolves to:

```
<a href=/click/Page.bok?template=test">
  Insert Text Here
</a>
```

Attributes

Supports HTML attributes

hidevalue

If the link does not resolve correctly, using this attribute will prevent the link text from displaying on the page.

shortcut

Shortcut attributes allow you to replace a large, complex piece of code with one word. All formatting and text are included within the shortcut. If you want to control formatting or text, replace the shortcut with the corresponding code (for assistance, try viewing the HTML source of the storefront page).

Valid Values

Value	Description
AnchorLink	Creates a link to a reference on the current page.
AnchorName	Creates an <a name> reference.
BuyIt	Creates a link to the Catalog Detail template for the current product, with the buyit.gif image.
BuyItNow	Creates a link that adds the current product to the cart, with the buyitnow.gif image.
GoBack	Creates a link back to the previous template.
ListLongImagesCommon	On the catalog list, displays a “previous” image, “next” image, and a series of identical on/off images for each page, allowing the user to browse the search results. Requires the following images: nextpon.gif, nextpoff.gif, nextnon.gif, nextnoff.gif, next1on.gif, next1off.gif. << 1 1 1 1 >> << 1 1 1 1 >> << 1 1 1 1 >>
ListLongImagesUnique	On the catalog list, displays a “previous” image, “next” image, and a series of unique on/off images for each page, allowing the user to browse the search results. Requires the following images: nextpon.gif, nextpoff.gif, nextnon.gif, nextnoff.gif, nextNon.gif, nextNoff.gif (where N is a number for each image in the series, starting with 1). << 1 2 3 4 >> << 1 2 3 4 >> << 1 2 3 4 >>
ListLongText	On the catalog list, displays “Prev”, “Next”, and a series of text numbers for each page. 1 2 3 4 Next Prev 1 2 3 4 Next Prev 1 2 3 4
ListMediumImagesCommon	On the catalog list, displays a series of identical on/off images for each page. Requires the following images: next1on.gif, next1off.gif. 1 1 1 1 1 1 1 1 1 1 1 1
ListMediumImagesUnique	On the catalog list, displays a series of identical on/off images for each page. Requires the following images: nextNon.gif, nextNoff.gif (where N is a number for each image in the series, starting with 1). 1 2 3 4 1 2 3 4 1 2 3 4
ListMediumText	On the catalog list, displays a series of text numbers for each page. 1 2 3 4 1 2 3 4 1 2 3 4
ListNext	On the catalog list, displays “Next”. Next Next
ListNext#	On the catalog list, displays a “Next N”, where N represents the number of search results on the next page.

Value	Description
	Next 9 Next 9
ListPrev	On the catalog list, displays “Prev”. Prev Prev
ListPrev#	On the catalog list, displays a “Prev N”, where N represents the number of search results on the prior page. Prev 9 Prev 9
ListPrevNextImages	On the catalog list, displays “previous” and “next” images. Requires the following images: nextpon.gif, nextpoff.gif, nextnon.gif, nextnoff.gif << >> << >> << >>
ListPrevNextText	On the catalog list, displays “Prev” and “Next”. Next Prev Next Prev
ListPrevNextText#	On the catalog list, displays “Prev N” and “Next N”, where N represents the number of search results on the prior and next pages. Next 9 Prev 9 Next 9 Prev 9
Lead	Creates a link to display the Lead Product template, with the lead.jpg image.
PoweredBy	Inserts a link to a web site, with the “Powered By” logo.
ShipTrackingNumber	Inserts a link to the Tracking template.

source

The object expression containing the location to which you wish to link.

<ss:media>

Creates an <embed> tag for referencing multimedia files.

Examples

To insert a flash file, use:

```
<ss:media file="intuit.swf" width="550" height="400" type="flash"/>
```

...which resolves to:

```
<object classid='clsid:02BF25D5-8C17-4B23-BC80-D3488ABDDC6B'
  codebase='http://www.apple.com/qtactivex/qtplugin.cab'
  width='550' height='400'>
  <param name='SRC' value='http://url/catalog/Car.avi'>
  <embed src="intuit.swf" width="550" height="400"
    type="application/x-shockwave-flash">
  </embed>
</object>
```

Attributes

Supports HTML attributes

source (Required)

The object expression that references the media file you wish to embed.

type

The type of file. If this attribute is not present, the tag will attempt to resolve to the correct format using the file extension (for example, files with a *.SWF extension will use the Macromedia Flash player.)

Valid Values

Value	Description
image	Image file (such as *.gif, *.jpg or *.png)
flash	Flash player.
audio	QuickTime player
video	QuickTime player

width

The width of the media file. The use of this attribute is strongly encouraged.

height

The height of the media file. The use of this attribute is strongly encouraged.

<ss:otherwise>

See the <ss:choose> tag for information.

<ss:param>

Passes parameter values to a tag.

Examples

To link to the home page and include the affiliate's referral number, use:

```
<ss:link source="$templateSet.homePage">
  <ss:param name="affiliate_no" value="$affiliate.affiliateNumber"/>
</ss:link>
```

...which resolves to:

```
<a href="../../../StoreFront.bok?affiliate_no=1"></a>
```

Attributes

name (Required)

The name of the parameter

value (Required)

The value of the parameter, can be either a string or a valid object expression.

<ss:password>

Creates an HTML password input field (where the entry is displayed using asterisks for additional security).

Examples

To collect the customer's password, use:

```
<ss:password source="$customer.password"/>
```

...which resolves to:

```
<input type="password" name="$customer.password"
size="10" maxlength="15">
```

Attributes

Supports HTML attributes

source (Required)

The object expression in which the password will be stored.

value

The default value for the field.

<ss:radio>

Creates a “radio” button field.

Examples

To display a radio button for a shipping method, use:

```
<ss:radio name="shipme" source="$shipQuote.shipMethod"/>
```

...which resolves to:

```
<input type="radio" name="shipme" value="S"/>
```

Attributes

Supports HTML attributes

checked

Use this to select the radio button by default.

name

The control name. Should only be used if the name is different than the source attribute.

source

The object expression in which the radio button value will be stored.

value

The value to be applied if the radio button is selected.

<ss:select>

Creates a list from which a customer can select an option.

Examples

To create a list of available product colors, use:

```
<ss:select source="$product.colors" name="color"/>
```

...which resolves to:

```
<select name="color">
  <option value="Blue">Blue</option>
  <option value="Green">Green</option>
</select>
```

To create a drop-down list of all manufacturers that are currently assigned to products, adding an option to allow the visitor to search for *all* manufacturers, use:

```
<ss:select name="oem" source="$catalog.oemList">
  <option value='all' selected>ALL</option>
</ss:select>
```

...which resolves to:

```
<select name="oem">
<option value='all' selected>ALL</option>
<option value="Beauteous Makeup">Beauteous Makeup</option>
<option value="DollyLand">DollyLand</option>
</select>
```

To create a list of countries, that defaults to selecting the user's country, use:

```
<ss:select source="$system.Countries" name="$customer.Country"
selected="$customer.Country"/>
```

...which resolves to:

```
<select name="$customer.country">
  <option value="...">...</option>
  <option value="US" selected>United States</option>
```

```
<option value="...">...</option>
</select>
```

Attributes

Supports HTML attributes

name (Required)

The object expression in which the selection will be stored.

selected

The value that will be selected by default.

Valid Values

Value	Description
String	Text string
Object Expression	Valid object expression

source (Required)

The object expression containing the data to be displayed in the list.

<ss:set>

Creates an “object” for use on the current template.

Examples

To use the `$product.applyPromotion($promotion)` value repeatedly in code, use:

```
<ss:foreach item="promotion" within="$product.activePromotions">
  <ss:set name="newPrice" value="$product.applyPromotion($promotion)" />
  <ss:if test="$promotion.promotionApplied">
    <ss:if test="$newPrice != $product.finalPrice">
      </ss:if>
      <ss:value source="$newPrice" />
    </ss:if>
  </ss:foreach>
```

The `<ss:set/>` tag defines a new object = “`$newPrice`”, and sets its value to “`$product.applyPromotion($promotion)`”. Now, instead of having to type a lengthy object expression repeatedly, you can simply type “`$newPrice`”.

Attributes

name (Required)

The name of the new object.

value (Required)

The object expression that defines the new object.

<ss:sortby>

Determines the sort order of items in a collection specified by the <ss:foreach> tag. You can use multiple <ss:sortby> tags to specify multiple sort orders when used in <ss:foreach within="\$catalog.results">. When used in <ss:foreach within="\$catalog.categoryList()/categoryListWithMembersOnly()/categoriesToDepth()/categoriesToDepthWithMembersOnly()"> (for example, on the Header template), only <ss:sortby item="categorypriority"> may be used (the default for these lists is to sort by category name).

Examples

To sort a list by product name, use:

```
<ss:sortby item="product" />
```

To sort a list by category priority, then by category, and then by product name, use:

```
<ss:sortby item="categorypriority" direction="asc" />
<ss:sortby item="category" direction="asc" />
<ss:sortby item="product" direction="asc" />
```

Attributes

item (Required)

The item by which a list may be sorted.

Valid Values

Value	Description
category	Category
categorypriority	Category sort priority
oem	Manufacturer Name
price	Product Price
product	Product Name
sku	Product SKU
subcategory	Category

direction

Indicates the direction of the sort order.

Valid Values

Value	Description
asc	Ascending
ascending	Ascending
dsc	Descending
desc	Descending
descending	Descending

<ss:textarea>

Creates a large text box for user entry on a form. If the *rows* or *cols* HTML attributes are not specified, the default will be 4 and 10.

Examples

To collect a message to be sent with an “Email a Friend” referral, use:

```
<ss:textarea source="$email.message" rows="3" cols="40"/>
```

...which resolves to:

```
<textarea name=$email.message cols=40 rows=3 wrap=virtual></textarea>
```

Attributes

Supports HTML attributes

source (Required)

The object expression in which the textarea data will be stored.

<ss:value>

Displays the value of the object. If you are displaying a date object expression, be sure to use the *format* attribute - otherwise, the date will appear as YYYY-MM-DD HH.MM.SS.NNN.

Examples

To display the store name, use:

```
<ss:value source="$store.name"/>
```

...which resolves to:

```
Retroactive
```

To display the product name for every product in the user’s cart, and then limit the length of each name to the first 23 characters, use:

```
<ss:foreach item="detail" within="$cart.details">
  <ss:value source="$detail.product.name" maxlength="23"/><br>
</ss:foreach>
```

...which resolves to:

```
Cutie Doll<br>
Fancy Girl Doll<br>
Dining Room Lithogra...<br>
```

Attributes

source

The object expression containing the value you wish to display.

currency

Defines how currency values should be displayed. The “\$” value should only be used to override Locale formatting.

default

Indicates what should be displayed if no value is found.

format

Defines the formatting characteristics for this tag when resolving the “source” attribute.

Valid Values

Value	Description
date.short	Date
date.medium	Date
date.full	Date
date.long	Date
time.short	Time
time.medium	Time
time.full	Time
time.long	Time
time24	Time
day	Day
month	Month
year	Year
percent	Percentage
%	Percentage
%N	Percentage, with the number of decimal places specified by ‘N’
truefalse	True or False
yesno	Yes or No
none	No formatting
money	Currency

maxlength

The maximum number of characters that will be displayed for a value. If this attribute is not specified, the value defined by StoreSense will be used.

escape

If the value being returned by the source attribute contains special characters, use this attribute to indicate how the characters should be escaped. This attribute can be useful when using SSML tags within JavaScript.

Valid Values

Value	Description
javastring	Escapes characters using JavaScript style escaping.
quoted	Escapes characters using quotation marks.
encode	Escapes characters by encoding them.

<ss:when>

Used with <ss:choose> to define a series of test conditions. Test conditions are evaluated in the order in which the <ss:when> tags appear in the code. To provide an alternative if none of the “when” conditions are met, include an <ss:otherwise> tag. See <ss:choose> for more information and examples.

Examples

To apply special formatting to the “shipped” status on an invoice, use:

```
<ss:choose>
  <ss:when test="$detail.status == 'Shipped'">
    <b>This order has been shipped</b>
  </ss:when>
  <ss:otherwise>
    <ss:value source="$detail.status" />
  </ss:otherwise>
</ss:choose>
```

...which resolves to:

```
<b>This order has been shipped</b>
```

...if shipped. If the order has not been shipped, it displays the current order status (without bold formatting).

Attributes

test (Required)

Defines the test condition. Includes an object expression, operand, and test value. You can test for multiple conditions by using a single “AND” or “OR” statement. AND/OR must either be in all caps, or use the following symbols: “&&” (AND), “||” (OR). Valid operands are:

- ◆ > - Greater Than
- ◆ < - Less Than
- ◆ >= - Greater Than or Equal To
- ◆ <= - Less Than or Equal To
- ◆ = - Equals
- ◆ == - Equals
- ◆ != - Not Equal To
- ◆ <> - Not Equal To
- ◆ MATCHES - Supports wildcard at the end of the string (for example ‘string*’).



Appendix D: Tag Mappings

This reference lists all tags used in previous version of StoreSense, and their new equivalents. This list includes all tags for all store types and modules.

</loop>	<code></ss:foreach></code>
<::exclude>	<code></ss:if></code>
<::about>	<code><ss:link source="\$templateSet.aboutUs">About Us</ss:link></code>
<::cart>	<code><ss:link source="\$templateSet.cart">Shopping Cart</ss:link></code>
<::cartcontents>	1: Standard usage <code><ss:if test="\$cart.details"> <ss:foreach item="detail" within="\$cart.details"> <ss:value source="\$detail.product.name"/>
 </ss:foreach> <ss:else/> Empty </ss:if></code> 2: When the "maxlength" attribute is used <code><ss:foreach item="detail" within="\$cart.details"> <ss:value source="\$detail.product.name" maxlength="Integer"/>
 </ss:foreach></code>
<::catalog>	<code><ss:link source="\$templateSet.catalog">Catalog</ss:link></code>
<::catalogmbr>	1: Text Only <code><ss:link source="\$templateSet.searchMembersOnly"> Members Only Catalog </ss:link></code> 2: Image <code><ss:link source="\$templateSet.searchMembersOnly"> <ss:image source="\$templateSet.images['members.gif']" border="0" align="bottom"/> </ss:link></code>
<::category=CATEGORY>	<code>\$url.searchlink('category=CATEGORY')</code>
<::checkout>	<code><ss:link source="\$templateSet.checkout">Checkout</ss:link></code>
<::customtext type="TYPE">	<code><ss:value source="templateSet.customText['TYPE']"/></code>

<::date>	<code><ss:value source="\$system.currentDate" format="date.short" /></code>
<::datelong>	<code><ss:value source="\$system.currentDate" format="date.long" /></code>
<::day>	<code><ss:value source="\$system.currentDate" format="day" /></code>
<::daysto>	<code><ss:value source="\$system.daysTill('DATE') " /></code>
<::daystochristmas>	<code><ss:value source="\$system.daysTill('Christmas') " /></code>
<::download>	<code><ss:link source="\$templateSet.downloadList"> Download Files </ss:link></code>
<::else>	<code><ss:else/></code>
<::emptycart>	<code><ss:link source="\$templateSet.cartEmpty"> Empty Cart </ss:link></code>
<::endif>	<code></ss:if></code>
<::endloop>	<code></ss:foreach></code>
<::exclude>	<code><ss:if test="\$store.storeType != 'STORETYPE'"></code>
<::featuredproduct name="OPTION">	<code><ss:value source="\$store.featuredProduct.OPTION" /></code>
<::fontcolor type="FONTTYPE">	<code><ss:font color="\$font.FONTTYPE"></code>
<::fontface type="FONTTYPE">	<code><ss:font face="\$font.FONTTYPE"></code>
<::fontsize type="FONTTYPE">	<code><ss:font size="\$font.FONTTYPE"></code>
<::goback>	<code><ss:link shortcut="goback" /></code>
<::if <::test=CONDITION>>	<code><ss:if test="CONDITION"></code>
<::if <:hasdownloads>>	<code><ss:if test="\$invoice.hasDownloads"></code>
<::if <:isbrowserversion type="VERSION">>	<code>\$browser.VERSION</code>
<::if <:isfeaturedproduct>>	<code><ss:if test="\$store.featuredProduct"></code>
<::if <:ismember>>	<code><ss:if test="\$customer.isLoggedIn() "></code>

<::if <:ispaymentmethod>>	<code><ss:if test="\$invoice.paymentMethod == PAYMENTMETHOD"></code>
<::if <:ispaymentmethodworldpay>>	<code><ss:if test="\$invoice.paymentMethodCode == WorldPay"></code>
<::if <:issubscription>>	<code><ss:if test="\$product.subscriptionEnabled"></code>
<::if <:issubscriptionrequired>>	<code><ss:if test="\$store.membersOnlySubscriptionRequired"></code>
<::if <:isworldpayfixcontact>>	<code><ss:if test="\$worldpay.isFixContact"></code>
<::if <:isworldpaytestmode>>	<code><ss:if test="\$worldpay.isTestMode"></code>
<::include file=FILE>	<code><ss:include source="\$store.files['FILE']"/></code>
<::include template="TEMPLATE">	<code><ss:include template="TEMPLATE"/></code>
<::join>	1: Image <code><ss:image source="\$templateSet.images['join.gif']" border="0" align="bottom"/></code> 2: Text <code><ss:link source="\$templateSet.subscribe">Join Now</ss:link></code>
<::languageoverride>	<code><ss:form shortcut="LanguageOverride"/></code>
<::logoff>	<code><ss:link source="\$templateSet.logoff">Logoff</ss:link></code>
<::logon>	<code><ss:link source="\$templateSet.logon">Logon</ss:link></code>
<::loop=category>	<code><ss:foreach item="category" within="\$catalog.categoriesToDepth(4)"></code>
<::member>	<code><ss:value source="\$customer.fullName"/></code>
<::membercompany>	<code><ss:value source="\$customer.company"/></code>
<::memberfirst>	<code><ss:value source="\$customer.firstName"/></code>
<::memberlast>	<code><ss:value source="\$customer.lastName"/></code>
<::month>	<code><ss:value source="\$system.currentDate" format="month"/></code>
<::oem=MANUFACTURER>	<code>\$url.searchLink('oem=MANUFACTURER')</code>
<::policy>	<code><ss:link source="\$templateSet.storePolicies">Policies</ss:link></code>

<::product=PRODUCT>	<code>\$url.searchLink('product=PRODUCT')</code>
<::register>	1: Text <pre><ss:image source="\$templateSet.images['register.gif']" border="0" align="bottom"/></pre> 2: Image <pre><ss:link source="\$templateSet.customerRegistration"> Register </ss:link></pre>
<::service>	<pre><ss:link source="\$templateSet.customerService"> Customer Service </ss:link></pre>
<::servicehistory>	<pre><ss:link source="\$templateSet.customerHistory"> Account History </ss:link></pre>
<::servicelost>	<pre><ss:link source="\$templateSet.lostPassword"> Lost Passwords </ss:link></pre>
<::store>	<pre><ss:link source="\$templateSet.homePage">Home</ss:link></pre>
<::storebillboard>	1: Link <pre><ss:link shortcut="\$store"> <ss:image source="\$store.getBillboard()" alt="\$store.name" border="0"/> </ss:link></pre> 2: Image only <pre><ss:image source="\$store.billboard"/></pre>
<::storecity>	<pre><ss:value source="\$store.city"/></pre>
<::storecountry>	<pre><ss:value source="\$store.country"/></pre>
<::storedescription>	<pre><ss:value source="\$store.description"/></pre>
<::storeemail>	<pre><ss:value source="\$store.email"/></pre>
<::storefax>	<pre><ss:value source="\$store.fax"/></pre>
<::storeimages>	<code>\$store.Images</code>
<::storelogo>	<pre><ss:if test="\$store.isFeatureEnabled('storeLogo')"> <ss:image source="\$store.storeLogo"/> </ss:if></pre>
<::storename>	<pre><ss:value source="\$store.name"/></pre>

<::storepage=FILE>	<code>\$url.pageFile('FILE')</code>
<::storepage>FILE	<code>\$url.pageFile('FILE')</code>
<::storepath>	<code>\$url.storePath</code>
<::storephone>	<code><ss:value source="\$store.phone"/></code>
<::storescript=SCRIPT>	<code>\$url.buildUrl('SCRIPT')</code>
<::storescriptprefix>	<code>\$system.prefix</code>
<::storescripts>	<code>\$url.basePath</code>
<::storescriptsuffix>	<code>\$system.suffix</code>
<::storestate>	<code><ss:value source="\$store.state"/></code>
<::storestreet>	<code><ss:value source="\$store.street"/></code>
<::storetemplate=TEMPLATE>	<code>\$url.buildUrlWithQueryString('Page', 'template=TEMPLATE')</code>
<::storeurl>	<code><ss:value source="\$store.url"/></code>
<::storezip>	<code><ss:value source="\$store.postalCode"/></code>
<::time24>	<code><ss:value source="\$system.currentDate" format="time24"/></code>
<::time>	<code><ss:value source="\$system.currentDate" format="time.short"/></code>
<::year>	<code><ss:value source="\$system.currentDate" format="year"/></code>
<:addtocart image>	<pre> <ss:if test="\$product.type == ProductType.LEAD"> <ss:link shortcut="lead"/> <ss:else/> <ss:include template="cartadd"/> </ss:if> </pre>
<:addtocart template=TEMPLATE>	<pre> <ss:if test="\$product.type == ProductType.LEAD"> <ss:link shortcut="lead"/> <ss:else/> <ss:include template="TEMPLATE"/> </ss:if> </pre>
<:addtocart>	<pre> <ss:if test="\$product.type == ProductType.LEAD"> <ss:link shortcut="lead"/> <ss:else/> </pre>

```
<ss:include template="cartadd"/>
</ss:if>
```

<:affiliate>

1: Form Field

```
<ss:edit source="$affiliate.name" size="40"/>
```

2: Affiliate Templates

```
<ss:value source="$affiliate.name"/>
```

3: Invoice Templates

```
<ss:value source="$invoice.affiliate.name"/>
```

<:affiliatecity>

1: Form Field

```
<ss:edit source="$affiliate.city" size="20"/>
```

2: Display Value

```
<ss:value source="$affiliate.city"/>
```

<:affiliatecontact>

1: Form Field

```
<ss:edit source="$affiliate.contact" size="40"/>
```

2: Display Value

```
<ss:value source="$affiliate.contact"/>
```

<:affiliatecountry>

1: Form Field

```
<ss:edit source="$affiliate.country" size="40"/>
```

2: Display Value

```
<ss:value source="$affiliate.country"/>
```

<:affiliateemail>

1: Form Field

```
<ss:edit source="$affiliate.email" size="40"/>
```

2: Display Value

```
<ss:value source="$affiliate.email"/>
```

<:affiliatefax>

1: Form Field

```
<ss:edit source="$affiliate.fax" size="20"/>
```

2: Display Value

```
<ss:value source="$affiliate.fax"/>
```

<:affiliateid>

```
<ss:value source="$invoice.affiliate.affiliateNumber"/>
```

<:affiliateno>

```
<ss:value source="$invoice.affiliate.affiliateNumber"/>
```

<:affiliatephone>

1: Form Field

```
<ss:edit source="$affiliate.phone" size="20"/>
```

2: Display Value

```
<ss:value source="$affiliate.phone"/>
```

<:affiliateresale>**1: Display Value**

```
<ss:value source="$affiliate.resale"/>
```

2: Form Field

```
<ss:edit source="$affiliate.resale" size="20"/>
```

<:affiliatestate>**1: Form Field**

```
<ss:edit source="$affiliate.state" size="10"/>
```

2: Display Value

```
<ss:value source="$affiliate.state"/>
```

<:affiliatestreet1>**1: Form Field**

```
<ss:edit source="$affiliate.street1" size="30"/>
```

2: Display Value

```
<ss:value source="$affiliate.street1"/>
```

<:affiliatestreet2>**1: Form Field**

```
<ss:edit source="$affiliate.street2" size="30"/>
```

2: Display Value

```
<ss:value source="$affiliate.street2"/>
```

<:affiliatestreet>**1: Form Field**

```
<ss:edit source="$affiliate.street" size="30"/>
```

2: Display Value

```
<ss:value source="$affiliate.street"/>
```

<:affiliateurl>**1: Form Field**

```
<ss:edit source="$affiliate.url" size="40"/>
```

2: Display Value

```
<ss:value source="$affiliate.url"/>
```

<:affiliatezip>**1: Form Field**

```
<ss:edit source="$affiliate.postalCode" size="10"/>
```

2: Display Value

```
<ss:value source="$affiliate.postalCode"/>
```

<:billingoption>

```
<ss:foreach item="paymentMethod"
  within="$store.activePaymentMethods
  ($cart.hasSubscriptionProducts())">
  <tr>
  <td colspan="2">
    <ss:radio name="PaymentMethodCode"
      value="$paymentMethod.paymentMethodCode"
      checked="$paymentMethod.isBillingDefault()" />
    <ss:value source="$paymentMethod.name" /></td>
  </tr>
  <tr>
  <td width="20"></td>
```

```

<td>
  <table border="0" cellpadding="2" cellspacing="2">
    <ss:foreach item="paymentParameter"
      within="$paymentMethod.paymentParameters($browser)">
      <ss:if test="$paymentMethod.name == 'Credit Card'">
        <ss:if test="$paymentParameter.name == 'CreditCardExpMM'">
          <tr align="left" valign="top">
            <td><ss:value source="$paymentParameter.displayText"/></td>
            <td><ss:value source="$paymentParameter.displayFormField()"/>
          </tr>
        <ss:else/>
          <ss:if test="$paymentParameter.name == 'CreditCardExpYY'">
            <ss:value source="$paymentParameter.displayFormField()"/>
          </td>
          </tr>
        <ss:else/>
          <tr align="left" valign="top">
            <td><ss:value source="$paymentParameter.displayText"/></td>
            <td><ss:value source="$paymentParameter.
              displayFormField()"/></td>
          </tr>
        </ss:if>
      </ss:if>
    </ss:foreach>
  </table>
</td>
</tr>
</ss:foreach>

```

<:billnew>

<ss:checkbox shortcut="ChangeBilling"/>

<:buyit>

<ss:link shortcut="BuyIt"/>

<:buyitnow>

<ss:link shortcut="BuyItNow"/>

<:cancel>

<input type="submit" name="cancel" value="Cancel">

<:canceldate>

<ss:value source="\$subscription.cancelDate" format="date.short"/>

<:cartcheckbox>

```

<ss:if test="$product.attributes">
<ss:else/>
  <ss:checkbox name="cartchk" value="$product.getUrl()"/>
</ss:if>

```

<:cartcolor>

```

<ss:if test="$product.hasColors()">
  <ss:select source="$product.colors" name="color"/>
</ss:if>

```

<:cartcolortitle>

<ss:if test="\$product.hasColors()">Color</ss:if>

<:cartcustom1>

```

<ss:if test="$product.customService1Title">
  <input type="text" name="custom1" maxlength="50" size="20"/>
</ss:if>

```

<:cartcustom1fee>

```
<ss:if test="$product.customService1Title">
  <ss:value source="$product.customService1Fee"/>
</ss:if>
```

<:cartcustom1title>

```
<ss:if test="$product.customService2Title">
  <ss:value source="$product.customService1Title"/>
</ss:if>
```

<:cartcustom2>

```
<ss:if test="$product.customService2Title">
  <input type="text" name="custom2" maxlength="50" size="20"/>
</ss:if>
```

<:cartcustom2fee>

```
<ss:if test="$product.customService2Title">
  <ss:value source="$product.customService2Fee"/>
</ss:if>
```

<:cartcustom2title>

```
<ss:if test="$product.customService2Title">
  <ss:value source="$product.customService2Title"/>
</ss:if>
```

<:cartend>

```
</ss:form>
```

<:cartname>

1: Form Field

```
<input type="text" name="cartname" size="30" maxlength="30">
```

2: Display Value

```
<ss:value source="$cart.name"/>
```

<:cartqty>

```
<input type="text" name="qty" value="1" size="3">
```

<:cartreset>

```
<input type="reset" value="Clear">
```

<:cartsave image>

```
<ss:if test="$customer.isLoggedIn() AND $cart.details">
  <ss:link source="$templateSet.cartSave">
    <ss:image source="$store.images['cartsave.gif']" border="0" align="bottom"/>
  </ss:link>
</ss:if>
```

<:cartsave>

```
<ss:if test="$customer.isLoggedIn() AND $cart.details">
  <ss:link source="$templateSet.cartSave">
    Save Cart
  </ss:link>
</ss:if>
```

<:cartselect image>

```
<ss:if test="$customer.hasSavedCarts">
  <ss:link source="$templateSet.cartSelect">
    <ss:image source="$store.images['cartsel.gif']"
      border="0" align="bottom"/>
  </ss:link>
</ss:if>
```

<:cartselect>

1: Radio Button

```
<ss:radio name="cartid" value="$cart.cartNumber"/>
```

2: Link

```
<ss:if test="$customer.hasSavedCarts">
  <ss:link source="$templateSet.cartSelect">
    Select Cart
  </ss:link>
</ss:if>
```

<:cartsize>

```
<ss:if test="$product.hasSizes()">
  <ss:select source="$product.sizes" name="size"/>
</ss:if>
```

<:cartsizetitle>

```
<ss:if test="$product.hasSizes()">Size</ss:if>
```

<:cartstart>

```
<ss:form shortcut="CartStart">
```

<:cartsubmit>

```
<input type="submit" value="Add To Shopping Cart">
```

<:category nosub>

```
<ss:select name="category" source="$catalog.categoriesToDepth(0)">
  <option value="all" selected>All</option>
</ss:select>
```

<:category>

```
<ss:value source="$listing.name"/>
```

<:category>

1: Storefront template, to display a list of all categories in the store

```
<ss:link source="$category">
  <ss:if test="$category.treeLevel == "1"">
    <ss:value source="$category.name"/>
  <ss:else/>
    <ss:value source="$category.name"/>
  </ss:if>
</ss:link>
```

2: Catalog templates, to display a list of categories

```
<ss:select name="category" source="$catalog.categoryList">
  <option value="all" selected>All</option>
</ss:select>
```

3: Within a <ss:foreach item="listing"> tag, displays the category name

```
<ss:value source="$listing.name"/>
```

<:categorydesc>

```
<ss:value source="$category.description"/>
```

<:categoryphoto ALIGN>

```
<ss:image source="$listing.category.photoImage"
  border="0" align="ALIGN"/>
```

<:categoryphoto>

```
<ss:image source="$listing.category.photoImage" border="0"/>
```

<:change name=chgbilling>

```
<ss:form shortcut="ChangeSubscriptionBilling"/>
```

<:change name=chgshipto>

```
<ss:form shortcut="ChangeSubscriptionShipTo"/>
```

<:change>

```
<ss:button source="$templateSet.images['change.gif']" border=0/>
```

<:checkout>

```
<ss:form shortcut="Checkout" />
```

<:checkoutpro>

```
<ss:form shortcut="CheckoutPro" />
```

<:city>**1: Display Value, customer templates**

```
<ss:value source="$customer.city" />
```

2: Form Field, customer templates

```
<ss:edit source="$customer.city" size="20" />
```

3: Form Field, invoice templates

```
<ss:edit source="$address.city" size="20" />
```

4: Display Value, lead templates

```
<ss:value source="$lead.City" />
```

5: Form Field, lead templates

```
<ss:edit source="$lead.City" size="20" />
```

<:company>**1: Display Value, customer templates**

```
<ss:value source="$customer.company" />
```

2: Form Field, customer templates

```
<ss:edit source="$customer.company" size="30" />
```

3: Form Field, lead templates

```
<ss:edit source="$lead.company" size="30" />
```

4: Display Value, lead templates

```
<ss:value source="$lead.company" />
```

5: Display Value, credit templates

```
<ss:value source="$credit.customer.company" />
```

<:country>**1: Display Value, customer templates**

```
<ss:value source="$customer.country" />
```

2: Form Field, lead templates

```
<ss:edit source="lead.country" size="30" />
```

3: Display Value, lead templates

```
<ss:value source="lead.country" />
```

4: Form Field, customer templates

```
<ss:edit source="$customer.country" size="30" />
```

5: Form Field, shipping templates

```
<ss:edit source="address.country" size="30" />
```

<:countrylist>**1: Lead templates**

```
<ss:select source="$system.countries" name="$lead.country"
selected="$store.country" />
```

2: Invoice templates

```
<ss:select source="$system.countries" name="$address.country"
selected="$address.country" />
```

3: Customer templates

```
<ss:select source="$system.countries" name="$customer.country"
```

<code>selected="\$store.country"/></code>
<:credit>
<code><ss:value source="\$customer.storeCredit"/></code>
<:creditdate>
<code><ss:value source="\$credit.enterDate" format="date.short"/></code>
<:creditmethod>
<code><ss:value source="\$credit.paymentMethod"/></code>
<:creditno>
<code><ss:value source="\$credit.creditNumber"/></code>
<:creditnote>
<code><ss:value source="\$credit.note"/></code>
<:creditsalestax>
<code><ss:value source="\$credit.tax"/></code>
<:creditshipping>
<code><ss:value source="\$credit.shipping"/></code>
<:creditsubtotal>
<code><ss:value source="\$credit.subtotal"/></code>
<:credittotal>
<code><ss:value source="\$credit.total"/></code>
<:currentextension>
<code><ss:value source="\$extension.currentIndex"/></code>
<:custcity>
1: Display Value, subscription templates <code><ss:value source="\$subscription.billToCity"/></code>
2: Display Value, supplier templates <code><ss:value source="\$edi.customer.city"/></code>
3: Display Value, customer templates <code><ss:value source="\$customer.city"/></code>
4: Display Value, credit templates <code><ss:value source="\$credit.customer.city"/></code>
5: Display Value, invoice templates <code><ss:value source="\$invoice.billToCity"/></code>
<:custcompany>
1: Display Value, supplier templates <code><ss:value source="\$edi.customer.company"/></code>
2: Form Field, customer templates <code><ss:edit source="\$customer.company" size="30"/></code>
3: Display Value, customer templates <code><ss:value source="\$customer.company"/></code>
4: Display Value, invoice templates <code><ss:value source="\$invoice.customer.company"/></code>
<:custcountry option=full>
1: Display Value, customer templates <code><ss:value source="\$customer.countryName"/></code>

- 2: Display Value, subscription templates
`<ss:value source="$subscription.billToCountry"/>`
- 3: Display Value, credit templates
`<ss:value source="$credit.customer.country"/>`
- 4: Display Value, invoice templates
`<ss:value source="$invoice.shipToCountry"/>`

<:custcountry>

- 1: Display Value, subscription templates
`<ss:value source="$subscription.billToCountry"/>`
- 2: Display Value, supplier templates
`<ss:value source="$edi.customer.country"/>`
- 3: Display Value, customer templates
`<ss:value source="$customer.country"/>`

<:custcredit>

`<ss:value source="$customer.storeCredit"/>`

<:custcreditcard>

`<ss:value source="$customer.creditCardCode"/>`

<:custcreditcardexp>

`<ss:value source="$customer.creditCardExpiration"/>`

<:custcreditcardnumber>

`<ss:value source="$customer.creditCardNumber"/>`

<:custcustom1>

- 1: Display Value, customer templates
`<ss:value source="$customer.customText1"/>`
- 2: Form Field, customer templates
`<ss:edit source="$customer.customText1" size="30"/>`
- 3: Display Value, supplier templates
`<ss:value source="$edi.customer.customText1"/>`

<:custcustom2>

- 1: Display Value, customer templates
`<ss:value source="$customer.customText2"/>`
- 2: Form Field, customer templates
`<ss:edit source="$customer.customText2" size="30"/>`
- 3: Display Value, supplier templates
`<ss:value source="$edi.customer.customText2"/>`

<:custcustom3>

- 1: Display Value, customer templates
`<ss:value source="$customer.customText3"/>`
- 2: Form Field, customer templates
`<ss:edit source="$customer.customText3" size="30"/>`
- 3: Display Value, supplier templates
`<ss:value source="$edi.customer.customText3"/>`

<:custcustom4>

- 1: Display Value, customer templates
`<ss:value source="$customer.customText4"/>`

2: Form Field, customer templates

```
<ss:edit source="$customer.customText4" size="30" />
```

3: Display Value, supplier templates

```
<ss:value source="$edi.customer.customText4" />
```

<:custcustom5>

1: Display Value, customer templates

```
<ss:value source="$customer.customText5" />
```

2: Form Field, customer templates

```
<ss:edit source="$customer.customText5" size="30" />
```

3: Display Value, supplier templates

```
<ss:value source="$edi.customer.customText5" />
```

<:custcustom6>

1: Display Value, customer templates

```
<ss:value source="$customer.custom6" />
```

2: Form Field, customer templates

```
<ss:edit source="$customer.customText6" size="30" />
```

3: Display Value, supplier templates

```
<ss:value source="$edi.customer.customText6" />
```

<:custcustom7>

1: Display Value, customer templates

```
<ss:value source="$customer.custom7" />
```

2: Form Field, customer templates

```
<ss:edit source="$customer.customText7" size="30" />
```

3: Display Value, supplier templates

```
<ss:value source="$edi.customer.customText7" />
```

<:custcustom8>

1: Display Value, customer templates

```
<ss:value source="$customer.custom8" />
```

2: Form Field, customer templates

```
<ss:edit source="$customer.customText8" size="30" />
```

3: Display Value, supplier templates

```
<ss:value source="$edi.customer.customText8" />
```

<:custemail2>

```
<ss:value source="$customer.email2" />
```

<:custemail>

1: Display Value, customer templates

```
<ss:value source="$customer.email" />
```

2: Form field, e-mail a friend templates

```
<ss:edit source="$email.senderEmail" />
```

<:custfax2>

```
<ss:value source="$customer.workFax" />
```

<:custfax>

```
<ss:value source="$customer.homeFax" />
```

<:custfirstname>

```
<ss:value source="$customer.firstName" />
```

<:custlastname>

```
<ss:value source="$customer.lastName"/>
```

<:custname>**1: Form Field, e-mail a friend templates**

```
<ss:edit source="$email.senderName"/>
```

2: Display Value, customer templates

```
<ss:value source="$credit.customer.fullName"/>
```

3: Link, history templates

```
<ss:link source="$customer">
  <ss:value source="$customer.fullName"/>
</ss:link>
```

4: Display Value, customer templates

```
<ss:value source="$customer.fullName"/>
```

5: Display Value, subscription templates

```
<ss:value source="$subscription.customer.fullname"/>
```

6: Display Value, supplier templates

```
<ss:value source="$edi.customer.fullName"/>
```

7: Display value, credit templates

```
<ss:value source="$credit.customer.fullName"/>
```

8: Display value, invoice templates

```
<ss:value source="$invoice.customer.fullName"/>
```

<:custno>**1: Display Value, customer templates**

```
<ss:value source="$customer.customerNumberFull"/>
```

2: Display Value, supplier templates

```
<ss:value source="$edi.customer.customerNumberFull"/>
```

<:custnotify>

```
<ss:value source="$customer.notify" format="yesno"/>
```

<:custom1>

```
<ss:edit name="custom1" source="$catalog.custom1" size="30"/>
```

<:custom2>

```
<ss:edit name="custom2" source="$catalog.custom2" size="30"/>
```

<:custom3>

```
<ss:edit name="custom3" source="$catalog.custom3" size="30"/>
```

<:custom4>

```
<ss:edit name="custom4" source="$catalog.custom4" size="30"/>
```

<:custpager>

```
<ss:value source="$customer.pager"/>
```

<:custphone2>**1: Display Value, customer templates**

```
<ss:value source="$customer.workPhone"/>
```

2: Display Value, invoice templates

```
<ss:value source="$invoice.customer.workPhone"/>
```

<:custphone3>

1: Display Value, customer templates

```
<ss:value source="$customer.mobilePhone"/>
```

2: Display Value, invoice templates

```
<ss:value source="$invoice.customer.mobilePhone"/>
```

<:custphone>

1: Display Value, customer templates

```
<ss:value source="$customer.homePhone"/>
```

2: Display Value, invoice templates

```
<ss:value source="$invoice.customer.homePhone"/>
```

<:custresale>

```
<ss:value source="$customer.resale"/>
```

<:custshipcity>

```
<ss:value source="$invoice.shipToCity"/>
```

<:custshipcountry>

```
<ss:value source="$invoice.shipToCountry"/>
```

<:custshipname>

```
<ss:value source="$invoice.recipient"/>
```

<:custshipstate>

```
<ss:value source="$invoice.shipToState"/>
```

<:custshipstreet>

```
<ss:value source="$invoice.shipToStreet"/>
```

<:custshipzip>

```
<ss:value source="$invoice.shipToPostalCode"/>
```

<:custsince>

```
<ss:value source="$customer.customerSince" format="yyyy"/>
```

<:custstate>

1: Display Value, supplier templates

```
<ss:value source="$edi.customer.state"/>
```

2: Display Value, customer templates

```
<ss:value source="$customer.state"/>
```

3: Display Value, subscription templates

```
<ss:value source="$subscription.billToState"/>
```

4: Display value, credit templates

```
<ss:value source="$credit.customer.state"/>
```

5: Display value, invoice templates

```
<ss:value source="$invoice.billToState"/>
```

<:custstreet1>

1: Display Value, customer templates

```
<ss:value source="$customer.street"/>
```

2: Display Value, credit templates

```
<ss:value source="$credit.customer.street1"/>
```

<:custstreet2>**1: Display Value, supplier templates**

```
<ss:value source="$edi.customer.street2"/>
```

2: Display Value, customer templates

```
<ss:value source="$customer.street2"/>
```

3: Display value, credit templates

```
<ss:value source="$credit.customer.street2"/>
```

4: Display value, subscription templates

```
<ss:value source="$subscription.billToStreet2"/>
```

<:custstreet>**1: Subscription templates**

```
<ss:value source="$subscription.billToStreet"/>
<ss:if test="$subscription.billToStreet2">
  <br><ss:value source="$subscription.street2"/>
</ss:if>
```

2: Supplier templates

```
<ss:value source="$edi.customer.street"/>
<ss:if test="$edi.customer.street">
  <br><ss:value source="$edi.customer.street"/>
</ss:if>
```

3: Customer templates

```
<ss:value source="$customer.street"/>
<ss:if test="$customer.street2">
  <br><ss:value source="$customer.street2"/>
</ss:if>
```

4: Credit templates

```
<ss:value source="$credit.customer.street"/>
<ss:if test="$credit.customer.street2">
  <br><ss:value source="$credit.customer.street2"/>
</ss:if>
```

5: Invoice templates

```
<ss:value source="$invoice.billToStreet"/>
<ss:if test="$invoice.billToStreet2">
  <br><ss:value source="$invoice.billToStreet2"/>
</ss:if>
```

<:custtext1>

```
<ss:edit source="$extension.extensionText1" size="40"/>
```

<:custtext2>

```
<ss:edit source="$extension.extensionText2" size="40"/>
```

<:custtext3>

```
<ss:edit source="$extension.extensionText3" size="40"/>
```

<:custtext4>

```
<ss:edit source="$extension.extensionText4" size="40"/>
```

<:custtext5>

```
<ss:edit source="$extension.extensionText5" size="40"/>
```

<:custtext6>

```
<ss:edit source="$extension.extensionText6" size="40"/>
```

<:custtext7>

```
<ss:edit source="$extension.extensionText7" size="40"/>
```

<:custtext8>	<code><ss:edit source="\$extension.extensionText8" size="40"/></code>
<:custtext9>	<code><ss:edit source="\$extension.extensionText9" size="40"/></code>
<:custzip>	1: Display Value, customer templates <code><ss:value source="\$customer.postalCode"/></code> 2: Display Value, supplier templates <code><ss:value source="\$edi.customer.postalCode"/></code> 3: Display Value, subscription templates <code><ss:value source="\$subscription.billToPostalCode"/></code> 4: Display value, credit templates <code><ss:value source="\$credit.customer.postalCode"/></code> 5: Display value, invoice templates <code><ss:value source="\$invoice.billToPostalCode"/></code>
<:delete>	<code><input type="submit" name="delete" value="Delete"></code>
<:edidate>	<code><ss:value source="\$ediSet.date"/></code>
<:edigscontrolno>	<code><ss:value source="\$ediSet.ediGsControlNo"/></code>
<:edigscount>	<code><ss:value source="\$ediSet.ediGsCount"/></code>
<:ediisacontrolno>	<code><ss:value source="\$ediSet.ediIsaControlNo"/></code>
<:edipo1count>	<code><ss:value source="\$edi.ediPolCount"/></code>
<:edipo1lineno>	<code><ss:value source="\$detail.ediPolLineNo"/></code>
<:edistcontrolno>	<code><ss:value source="\$edi.ediStControlNo"/></code>
<:edistcount>	<code><ss:value source="\$edi.ediStCount"/></code>
<:editime>	<code><ss:value source="\$ediSet.time"/></code>
<:email2>	<code><ss:edit source="\$customer.email2" size="40"/></code>
<:email>	1: Display Value, customer templates <code><ss:value source="\$customer.email"/></code> 2: Form Field, customer templates <code><ss:edit source="\$customer.email" size="40"/></code> 3: Form Field, lead templates <code><ss:edit source="\$lead.email" size="40"/></code>

4: Display Value, lead templates

```
<ss:value source="$lead.email"/>
```

5: Form Field, lost password and login templates

```
<ss:edit source="$customerRegistry.email"
value="$customer.email" size="40"/>
```

<:emailafriend>

```
<ss:link source="$templateSet.emailAFriend">
Email A Friend
</ss:link>
```

<:emailcc>

```
<ss:edit source="$customer.email2" size="40"/>
```

<:emptycart>

```
<ss:if test="$cart.details">
<ss:link source="$templateSet.cartEmpty">EmptyCart</ss:link>
</ss:if>
```

<:endloop>

```
</ss:foreach>
```

<:errmsg>

```
<ss:value source="$message"/>
```

<:expirationdate>

```
<ss:value source="$subscription.expirationDate"
format="date.short" default="&nbsp;"/>
```

<:fax2>

1: Display Value, customer templates

```
<ss:value source="$customer.workFax"/>
```

2: Form Field, customer templates

```
<ss:edit source="$customer.workFax" size="20"/>
```

<:fax>

1: Display Value, customer templates

```
<ss:value source="$customer.homeFax"/>
```

2: Form Field, customer templates

```
<ss:edit source="$customer.homeFax" size="20"/>
```

3: Display Value, lead templates

```
<ss:edit source="$lead.fax"/>
```

4: Form Field, lead templates

```
<ss:edit source="$lead.fax" size="20"/>
```

<:filelink>

```
<ss:link source="$detail">
<ss:value source="$detail.downloadFileName">
</ss:link>
```

<:firstname>

1: Display Value, lead templates

```
<ss:value source="$lead.firstName"/>
```

2: Form Field, lead templates

```
<ss:edit source="$lead.firstName" size="20"/>
```

3: Display Value, customer templates

```
<ss:value source="$customer.firstName"/>
```

4: Form Field, lost password and login templates

```
<ss:edit source="$customerRegistry.firstName" size="20"/>
```

5: Form Field, customer templates

```
<ss:edit source="$customer.firstName" size="20"/>
```

<:footer>

```
<ss:include template="footer"/>
```

<:giftwrap>

```
<ss:if test="$store.gift && $store.wrap">
  <ss:checkbox source="$address.giftWrapEnabled"/>
  For an additional <ss:value source="$store.wrapPrice"/>
  we can gift wrap your package.
</ss:if>
```

<:groupbycategory>

```
<ss:sortby item="category"/>
```

<:groupbyoem>

```
<ss:sortby item="oem"/>
```

<:groupbyproduct>

```
<ss:sortby item="product"/>
```

<:groupbysubcategory>

```
<ss:sortby item="category"/>
```

<:header>

```
<ss:include template="header"/>
```

<:if <:isaffiliatepresent>>

```
<ss:if test="$invoice.affiliate">
```

<:if <:ischeckout>>

```
<ss:if test="$invoice.invoiceNumber == 0">
```

<:if <:ispromodiscountpresent>>

```
<ss:if test="$invoice.promotionDiscount">
```

<:if <:isshippingamount>>

```
<ss:if test="$invoice.hasShippingQuotes()">
```

<:if <:issubscription>>

1: Test to see if subscriptions are enabled for the store

```
<ss:if test="$product.subscriptionEnabled">
```

2: Tests to see if the customer has active subscriptions

```
<ss:if test="$customer.hasActiveSubscriptions">
```

<:invoiceno>

1: Credit templates

```
<ss:value source="$credit.invoiceNumber" default="N/A"/>
```

2: Invoice templates

```
<ss:value source="$invoice.invoiceNumber"/>
<ss:if test="$supplier.suffixInd">
  <ss:value source="$supplier.suffix"/>
</ss:if>
```

3: Supplier templates

```
<ss:value source="$edi.invoiceNumber"/>
```

<:keyword>**1: Form Field**

```
<ss:edit source="$catalog.keyword" size="30"/>
```

2: Display Value

```
<ss:value source="$catalog.keyword"/>
```

<:keywordoption>

```
<input type="radio" name="keywordoption" value="AND" checked>And  
<input type="radio" name="keywordoption" value="OR">Or
```

<:lastbilldate>

```
<ss:value source="$subscription.lastInvoiceDate" format="date.short"/>
```

<:lastinvoicedate>

```
<ss:value source="$subscription.lastInvoiceDate" format="date.short"/>
```

<:lastname>**1: Display Value, customer templates**

```
<ss:value source="$customer.lastName"/>
```

2: Form Field, customer templates

```
<ss:edit source="$customer.lastName" size="20"/>
```

3: Display Value, lead templates

```
<ss:value source="$lead.lastName"/>
```

4: Form Field, lead templates

```
<ss:edit source="$lead.lastName" size="20"/>
```

5: Form Field, lost password and login templates

```
<ss:edit source="$customerRegistry.lastName" size="20"/>
```

<:lastrenewdate>

```
<ss:value source="$subscription.lastRenewDate" format="date.short"/>
```

<:lead>

```
<ss:link source="$templateSet.lead">  
  <ss:image source="$templateSet.images['lead.gif']"  
    border="0" align="bottom"/>  
</ss:link>
```

<:leadcheck1>**1: Display Value**

```
<ss:value source="$lead.CustomCheckbox1" format="yesno"/>
```

2: Form Field

```
<ss:checkbox source="$lead.customCheckbox1"/>
```

<:leadcheck2>**1: Display Value**

```
<ss:value source="$lead.CustomCheckbox2" format="yesno"/>
```

2: Form Field

```
<ss:checkbox source="$lead.customCheckbox2"/>
```

<:leadcheck3>**1: Display Value**

```
<ss:value source="$lead.CustomCheckbox3" format="yesno"/>
```

2: Form Field

```
<ss:checkbox source="$lead.customCheckbox3"/>
```

<:leadcheck4>**1: Display Value**

```
<ss:value source="$lead.CustomCheckbox4" format="yesno"/>
```

2: Form Field

```
<ss:checkbox source="$lead.customCheckbox4"/>
```

<:leadcheck5>**1: Display Value**

```
<ss:value source="$lead.CustomCheckbox5" format="yesno"/>
```

2: Form Field

```
<ss:checkbox source="$lead.customCheckbox5"/>
```

<:leadcheck6>**1: Display Value**

```
<ss:value source="$lead.CustomCheckbox6" format="yesno"/>
```

2: Form Field

```
<ss:checkbox source="$lead.customCheckbox6"/>
```

<:leadcustom1>**1: Display Value**

```
<ss:value source="$lead.CustomText1"/>
```

2: Form Field

```
<ss:edit source="$lead.customText1"/>
```

<:leadcustom2>**1: Display Value**

```
<ss:value source="$lead.CustomText2"/>
```

2: Form Field

```
<ss:edit source="$lead.customText2"/>
```

<:leadcustom3>**1: Display Value**

```
<ss:value source="$lead.CustomText3"/>
```

2: Form Field

```
<ss:edit source="$lead.customText3"/>
```

<:leadcustom4>**1: Display Value**

```
<ss:value source="$lead.CustomText4"/>
```

2: Form Field

```
<ss:edit source="$lead.customText4"/>
```

<:leadcustom5>**1: Display Value**

```
<ss:value source="$lead.CustomText5"/>
```

2: Form Field

```
<ss:edit source="$lead.customText5"/>
```

<:leadcustom6>**1: Display Value**

```
<ss:value source="$lead.CustomText6"/>
```

2: Form Field

```
<ss:textarea source="$lead.customText6" rows="4" cols="35"/>
```

<:loop=edi>

```
<ss:foreach item="edi" within="$ediset.ediRecords">
```

<:loop=product>

```
<ss:foreach item="detail" within="$invoice.details">
```

<:loop=subcategory>

```
<ss:foreach item="category"
  within="$system.emptyCategoryList">
```

<:loop=subscription>

```
<ss:foreach item="subscription" within="$customer.subscriptions">
```

<:loop>

1: Creates a list of customer invoices

```
<ss:foreach item="detail" within="$customer.accountHistory">
```

2: Creates a list of downloadable files

```
<ss:foreach item="invoice" within="$customer.downloadFiles">
```

3: Creates a list of search results

```
<ss:foreach item="listing" within="$catalog.results">
```

4: Creates a list of details for each item on an invoice

```
<ss:foreach item="detail" within="$invoice.details">
```

5: Creates a list of stores purchased by a customer

```
<ss:foreach item="store" within="$store.catalog.stores">
```

6: Creates a list of details for each item in a cart

```
<ss:foreach item="detail" within="$cart.details">
```

7: Creates a list of store classifications

```
<ss:foreach item="classification"
  within="$store.classifications">
```

8: Creates a list of saved carts

```
<ss:foreach item="cart" within="$customer.savedCarts">
```

9: Creates a list of relateds

```
<ss:foreach item="product" within="$product.relatedProducts">
```

10: Creates a list of items for which the shopper does not have authorization to purchase

```
<ss:foreach item="product" within="$cart.unauthorizedProducts">
```

11: Creates a list of details for each item on a supplier template

```
<ss:foreach item="detail" within="$edi.details">
```

12: Creates a list of items that are low on inventory

```
<ss:foreach item="product" within="$warehouse.lowInventoryProducts">
```

13: Creates a list of credits

```
<ss:foreach item="detail" within="$credit.details">
```

<:mallclassificationdescription>

```
<ss:value source="$classification.description" default="&nbsp;"/>
```

<:mallclassificationlink>

```
<ss:link source="$classification">
  <ss:value source="$classification.name"/>
</ss:link>
```

<:mallclassificationselect>

```
<ss:select name="classificationid" source="$store.classifications">
  <option value="" selected></option>
```

	</ss:select>
<:message>	
1: Form Field	<ss:edit source="\$address.message" size="40"/>
2: Display Value	<ss:value source="\$message"/>
<:messagearea>	
1: E-mail a friend templates	<ss:textarea source="\$email.message" rows="3" cols="40"/>
2: Shipping templates	<ss:textarea source="\$address.message" cols="40" rows="3" wrap="virtual"/>
<:name>	
	<ss:link source="\$templateSet.customerProfile"> <ss:value source="\$customer.firstName"/> <ss:value source="\$customer.lastName"/> </ss:link>
<:next #>	
	<ss:link shortcut="ListNext#"/>
<:next image#>	
	<ss:link shortcut="LongImagesUnique"/>
<:next image>	
	<ss:link shortcut="ListPrevNextImages"/>
<:next pages>	
	<ss:link shortcut="ListNext#"/>
<:next>	
	<ss:link shortcut="ListNext"/>
<:nextbilldate>	
	<ss:value source="\$subscription.nextBillDate" format="date.short"/>
<:noshippingamount>	
	<ss:value source="\$invoice.shipping"/>
<:notify>	
	<ss:value source="\$customer.notify" format="yesno"/> <ss:checkbox source="\$customer.notify"/>
<:oem>	
1: Display Value, catalog list templates	<ss:if test="\$store.name AND \$listing.oem"> <ss:value source="\$listing.oem.name"/> <ss:else/> <ss:value source="\$product.oem.name"/> </ss:if>
2: Form Field, catalog templates	<ss:edit source="\$catalog.oem" size="30"/>
3: Display Value, catalog detail templates	<ss:value source="\$product.oem"/>
<:oemlist>	
	<ss:select name="oem" source="\$catalog.oemList"> <option value="all" selected>All</option> </ss:select>

<:oemlogo link>

```
<a href="http://$product.oem.url">
  <ss:image source="$product.oem.logo" border="0" />
</a>
```

<:oemlogo>

```
<ss:image source="$product.oem.logo" />
```

<:oemurl link>

```
<a href="http://$product.oem.url">
  <ss:value source="$product.oem.url" />
</a>
```

<:oemurl>

```
<ss:value source="$product.oem.url" />
```

<:orderamount>**1: Apply special formatting if this is a credit amount**

```
<ss:if test="$detail.credit">
  <font color="red">
    <ss:value source="$detail.transactionAmount" />
  </font>
<ss:else/>
  <ss:value source="$detail.transactionAmount" />
<ss:if>
```

2: No special formatting

```
<ss:value source="$detail.transactionAmount" />
```

<:ordercredit>

```
<ss:value source="$invoice.credit" />
```

<:orderdate>**1: Link, invoice templates**

```
<ss:link source="$detail">
  <ss:value source="$detail.date" format="date.short" />
</ss:link>
```

2: Value, invoice templates

```
<ss:value source="$invoice.enterDate" format="date.short" />
```

3: Value, supplier templates

```
<ss:value source="$edi.enterDate" format="yyyyMMdd" />
```

<:ordergift>

```
<ss:if test="$invoice.message">
  Message:
  <ss:value source="$invoice.message" />
</ss:if>
```

<:ordermethod>**1: Billing templates**

```
<ss:value source="$paymentMethod.PaymentMethod"
  default="&nbsp;" />
```

2: Customer templates

```
<ss:value source="$customer.PaymentMethod" />
```

3: Invoice templates

```
<ss:value source="$invoice.paymentMethod" />
```

<:orderno>

```
<ss:value source="$detail.itemNumber" default="&nbsp;" />
```

<:ordernote>	<code><ss:value source="\$invoice.orderNote"/></code>
<:orderpono>	<code><ss:value source="\$invoice.purchaseOrderNumber"/></code>
<:orderpromo>	<code><ss:value source="\$invoice.promotionCode"/></code>
<:orderpromodiscount>	<code><ss:value source="\$invoice.promotionDiscount"/></code>
<:orderpromotext>	<code><ss:foreach item="promotion" within="\$invoice.promotions"></code> <code> <ss:value source="\$promotion.invoiceText"/></code> <code>
</code> <code></ss:if></code>
<:ordersalestax>	<code><ss:value source="\$invoice.tax"/></code>
<:ordershipdate>	<code><ss:value source="\$invoice.shipDate" default="&nbsp;"</code> <code> format="date.short"/></code>
<:ordershipmethod>	<code><ss:value source="\$invoice.shipMethodDescription"</code> <code> default="&nbsp;" /></code>
<:ordershipping>	<code><ss:value source="\$invoice.shipping"/></code>
<:ordershiptrack>	<code><ss:value source="\$invoice.shipTrackNumber.number"/></code>
<:orderstatus>	1: History and tracking templates <code><ss:value source="\$detail.status" default="&nbsp;" /></code> 2: Invoice templates <code><ss:value source="\$invoice.status"/></code>
<:ordersubtotal>	<code><ss:value source="\$invoice.subtotal"/></code>
<:ordertotal>	1: Subscription templates <code><ss:value source="\$subscription.price"/></code> 2: Invoice templates <code><ss:value source="\$invoice.total"/></code>
<:ordertotaldue>	1: Invoice templates <code><ss:value source="\$invoice.totalDue"/></code> 2: Subscription templates <code><ss:value source="\$subscription.price"/></code>
<:ordertrackingno>	<code><ss:value source="\$invoice.tracking" default="&nbsp;" /></code>
<:ordertype>	<code><ss:value source="\$detail.activity"/></code>

<:page=FILENAME.EXT+LINK TEXT>

```
<ss:link source="$store.files['FILENAME.EXT']">LINK TEXT</ss:link>
```

<:pager>

- 1: Display Value, customer templates

```
<ss:value source="$customer.pager" />
```

- 2: Form Field, customer templates

```
<ss:edit source="$customer.pager" size="20" />
```

<:password2>

```
<input type="password" name="confirm" size="10">
```

<:password>

- 1: Display Value

```
<ss:value source="$customer.password" />
```

- 2: Form Field

```
<ss:password source="$customerRegistry.password" size="10" />
```

<:paymentmethod>

```
<ss:value source="$subscription.paymentMethod" />
```

<:phone2>

- 1: Display Value, customer templates

```
<ss:value source="$customer.workPhone" />
```

- 2: Form Field, customer templates

```
<ss:edit source="$customer.workPhone" size="20" />
```

<:phone3>

- 1: Display Value, customer templates

```
<ss:value source="$customer.mobilePhone" />
```

- 2: Form Field, customer templates

```
<ss:edit source="$customer.mobilePhone" size="20" />
```

<:phone>

- 1: Display Value, customer templates

```
<ss:value source="$customer.homePhone" />
```

- 2: Form Field, customer templates

```
<ss:edit source="$customer.homePhone" size="20" />
```

- 3: Display Value, lead templates

```
<ss:value source="$lead.phone" />
```

- 4: Form Field, lead templates

```
<ss:edit source="$lead.phone" size="20" />
```

<:polimit>

```
<ss:edit source="$customer.orderLimit" />
```

<:product alink>

```
<ss:link shortcut="AnchorLink" name="$product.productNumber">  
  <ss:value source="$product.name" />  
</ss:link>
```

<:product aname>

```
<ss:link shortcut="AnchorName" name="$product.productNumber" />
```

<:product nolink>**1: Display Value**

```
<ss:value source="$product.name" />
```

2: Display Value

```
<ss:value source="$detail.product.name" />
```

<:product>**1: Display Value, catalog detail templates**

```
<ss:value source="$product.name" />
```

2: Link, catalog list templates

```
<ss:link source="$product">
  <ss:value source="$product.name" />
</ss:link>
```

3: Form Field, catalog templates

```
<ss:edit source="$catalog.product" size="30" />
```

4: Display Value, invoice templates

```
<ss:value source="$detail.product.name" />
```

5: Display Value, subscription templates

```
<ss:value source="$subscription.product.name" />
```

<:productattr>**1: Invoice templates**

```
<ss:value source="$detail.attributeText" />
```

2: Subscription templates

```
<ss:value source="$subscription.attributeText" />
```

<:productbackorder>

```
<ss:if test="$detail.isbackordered"><b>Backordered</b></ss:if>
```

<:productbrief>

```
<ss:value source="$product.brief" />
```

<:productcost>**1: Catalog detail templates**

```
<ss:value source="$product.cost" />
```

2: Invoice templates

```
<ss:value source="$detail.product.cost" />
```

<:productcustom1>

```
<ss:value source="$product.customText1" />
```

<:productcustom2>

```
<ss:value source="$product.customText2" />
```

<:productcustom3>

```
<ss:value source="$product.customText3" />
```

<:productcustom4>

```
<ss:value source="$product.customText4" />
```

<:productcustom5>

```
<ss:value source="$product.customText5" />
```

<:productcustom6>

```
<ss:value source="$product.customText6" />
```

<:productdesc>	<code><ss:value source="\$product.description"/></code>
<:productno>	1: Subscription Templates <code><ss:value source="\$subscription.product.productNumber"/></code> 2: Catalog list templates <code><ss:value source="\$detail.product.productNumber"/></code> 3: Catalog detail templates <code><ss:value source="\$product.productNumber"/></code>
<:productoem>	<code><ss:value source="\$product.oem"/></code>
<:productshipping>	<code><ss:value source="\$product.shippingCost"/></code>
<:productphoto LEFT/RIGHT>	<code><ss:image source="\$product.photoImage" align="LEFT/RIGHT" border="0"/></code>
<:productphoto>	<code><ss:image source="\$product.photoImage" border="0"/></code>
<:productphotocart>	<code><ss:image source="\$catalog.images[\$detail.product.thumbnailImageName]" width="25" height="25"/></code>
<:productprice>	1: Catalog detail templates <code><ss:include macro="product.price"/></code> 2: Catalog list, invoice templates <code><ss:include macro="product.price" "/></code> 3: Subscription templates <code><ss:value source="\$subscription.price"/></code>
<:productpriceretail>	<code><ss:value source="\$detail.product.retailPrice"/></code>
<:productpricesale>	<code><ss:value source="\$product.finalPrice"/></code>
<:productpricesavings format=percent fractiondigits=XX>	<code><ss:value source="\$product.priceSavingsPercent" format="%XX"/></code>
<:productpricesavings format=percent>	<code><ss:value source="\$product.priceSavingsPercent" format="percent"/></code>
<:productpricesavings>	1: Catalog detail templates <code><ss:value source="\$product.priceSavingsCurrency"/></code> 2: Invoice, catalog list templates <code><ss:value source="\$detail.product.priceSavingsCurrency"/></code>
<:productpricex>	<code><ss:include macro="attributedetailfull"/></code>
<:productqty>	<code><ss:value source="\$detail.quantity"/></code>

<:productquantity>**1: Catalog list, invoice templates**

```
<ss:value source="$detail.quantity"/>
```

2: Subscription templates

```
<ss:value source="$subscription.quantity"/>
```

3: Catalog detail templates

```
<ss:value source="$product.quantityInStock"/>
```

4: Form field, cart template

```
<ss:edit source="$detail.quantity" name="qty" size="4"/>
```

<:productretail>

```
<ss:value source="$product.retailPrice"/>
```

<:productsaleend>

```
<ss:if test="$product.onSale">
```

```
  Sale Ends: <ss:value source="$product.saleEndDate"  
    format="date.short"/>
```

```
</ss:if>
```

<:productsku>**1: Catalog list templates**

```
<ss:value source="$detail.sku"/>
```

2: Invoice templates

```
<ss:value source="$detail.productInfo.sku"/>
```

3: Catalog detail templates

```
<ss:value source="$product.sku"/>
```

<:productstock>**1: Catalog list templates**

```
<ss:if test="$detail.product.quantityInStock">  
  In Stock  
<ss:else/>  
  Backordered  
</ss:if>
```

2: Catalog detail templates

```
<ss:if test="$product.quantityInStock">  
  In Stock  
<ss:else/>  
  Backordered  
</ss:if>
```

<:productstockattr>

```
<ss:foreach item="attr" within="$product.attributes">  
  <ss:value source="$attr.size"/>  
  (<ss:value source="$attr.quantityInStock"/>),  
</ss:foreach>
```

<:productstockback>

```
<ss:if test="$product.quantityInStock < 0">Backordered</ss:if>
```

<:productstockimage>

```
<ss:if test="$product.quantityInStock">  
  <ss:image source="$templateSet.images['instock.gif']"  
    border="0"/>  
</ss:if>
```

<:productstockqty>

```
<ss:value source="$product.quantityInStock"/>
```

<:productstockyesno>

```
<ss:value source="$product.quantityInStock" format="yesno" />
```

<:productsurcharge>

```
<ss:value source="$product.surcharge" />
```

<:productsurchargeTitle>

```
<ss:value source="$product.surchargeTitle" />
```

<:productthumbnail link>

```
<ss:link source="$product.photoImage">
  <ss:image source="$catalog.images[$product.thumbnailImageName]"
    border="0" />
</ss:link>
```

<:productthumbnail nolink>

```
<ss:image source="$catalog.images[$product.thumbnailImageName]"
  border="0" />
```

<:productthumbnail>

```
<ss:link source="$product">
  <ss:image source="$product.thumbnailImage" border="0" />
</ss:link>
```

<:producttotal>

```
<ss:value source="$detail.total" />
```

<:productunit>**1: Catalog detail templates**

```
<ss:value source="$product.unitOfMeasure" />
```

2: Invoice, catalog list templates

```
<ss:value source="$product.unitOfMeasure" />
```

<:productweight>**1: Invoice, catalog list templates**

```
<ss:value source="$detail.product.weightText" />
```

2: Catalog detail templates

```
<ss:value source="$product.weight" />
```

<:promo>

```
<input type="text" name="promo" value="" size="10" />
```

<:purchasedate>**1: Link**

```
<ss:link source="$url.buildUrl('CustomerSubscription')">
  <ss:param name="subscription"
    value="$subscription.subscriptionNumber" />
  <ss:value source="$subscription.purchaseDate"
    format="date.short" />
</ss:link>
```

2: Display Value

```
<ss:value source="$subscription.purchaseDate" format="date.short" />
```

<:recipient>**1: Invoice templates**

```
<ss:edit source="$address.recipient" size="40" />
```

2: E-mail a friend templates

```
<ss:edit source="$email.recipientName" />
```

<:recipientemail>**1: E-mail a friend templates**

```
<ss:edit source="$email.recipientEmail"/>
```

2: Invoice templates

```
<ss:edit source="$email.recipientEmail"/>
```

<:referrer>

```
<ss:link source="$templateSet.searchResults">
  Return To Catalog
</ss:link>
```

<:related>

```
<ss:if test="$product.hasRelateds">
  <ss:include template="catalogrelated"/>
</ss:if>
```

<:resale>**1: Display Value, customer templates**

```
<ss:value source="$customer.resale"/>
```

2: Form Field, customer templates

```
<ss:edit source="$customer.resale" size="20"/>
```

<:resaleind>

```
<ss:checkbox source="resale"/>
```

<:reset>

```
<input type="reset" value="Clear">
```

<:searchend>

```
<ss:value source="$catalog.end"/>
```

<:searchstart>

```
<ss:value source="$catalog.start"/>
```

<:searchstorecriteria>

```
<ss:value source="$storecatalog.searchCriteria" default="All"/>
```

<:searchstorename>

```
<ss:link shortcut="MallStoreLink">
  <ss:value source="$store.name"/>
</ss:link>
```

<:searchtext>

```
<ss:value source="$catalog.searchtext" default="All"/>
```

<:searchtotal>

```
<ss:value source="$catalog.total"/>
```

<:selectshipping>

```
<ss:include template="selectshipping"/>
```

<:shipmethod>

```
<ss:value source="$invoice.shipMethodDescription"/>
```

<:shipnextday>

```
<input type="radio" name="shipme" value="N"/>
<ss:value source="$store.shipping.getServiceDescription('N')"/>
```

<:shipnextdayrate>

```
<ss:value source="$cart.getShipping('N')"/>
```

<:shipping>

```
<ss:value source="$cart.shipping" default="0"/>
```

<:shippingquotes>

```
<ss:foreach item="quote" within="$invoice.shippingQuotes">
  <ss:if test="$invoice.shipMethod == $quote.shipMethod">
    <option value="$quote.value" selected>
  <ss:else/>
    <option value="$quote.value">
      <ss:value source="$quote.name"/>:
      <ss:value source="$quote.amount"/>
    </option>
  </option>
```

<:shipstandard>

```
<input type="radio" name="shipme" value="S" checked/>
<ss:value source="$store.shipping.getServiceDescription('S')"/>
```

<:shipstandardrate>

```
<ss:value source="$cart.getShipping('S')"/>
```

<:shipto>

```
<ss:checkbox shortcut="ChangeShippingAddress"/>
```

<:shiptocity>**1: Display Value**

```
<ss:value source="$invoice.shipToCity"/>
```

2: Display Value, subscription templates

```
<ss:value source="$subscription.shipToCity"/>
```

<:shiptocompany>**1: Display Value**

```
<ss:value source="$invoice.company"/>
```

2:

```
<ss:value source="$invoice.company" default="&nbsp;"/>
```

<:shiptocountry option=full>**1: Invoice templates**

```
<ss:value source="$invoice.shipToCountry.name"/>
```

2: Subscription templates

```
<ss:value source="$subscription.shipToCountry.name"/>
```

<:shiptocountry>**1: Display Value, subscription templates**

```
<ss:value source="$subscription.shipToCountry"/>
```

2: Display Value

```
<ss:value source="$invoice.shipToCountry"/>
```

<:shiptoname>**1: Subscription templates**

```
<ss:value source="$subscription.recipient"/>
```

2: Invoice templates

```
<ss:value source="$invoice.recipient"/>
```

<:shiptophone>**1: Subscription templates**

```
<ss:value source="$subscription.phone"/>
```

2: Invoice templates

```
<ss:value source="$invoice.phone"/>
```

<:shiptostate>**1: Subscription templates**

```
<ss:value source="$subscription.shipToState" />
```

2: Invoice templates

```
<ss:value source="$invoice.shipToState" />
```

<:shiptostreet2>

```
<ss:value source="$invoice.shipToStreet2" />
```

<:shiptostreet>**1: Subscription templates**

```
<ss:value source="$subscription.shipToStreet" />
<ss:if test="$subscription.shipToStreet2">
  <br/><ss:value source="$subscription.shipToStreet2" />
</ss:if>
```

2: Invoice templates

```
<ss:value source="$invoice.shipToStreet" />
<ss:if test="$invoice.street2">
  <br/><ss:value source="$invoice.street2" />
</ss:if>
```

<:shiptozip>**1: Display Value, subscription templates**

```
<ss:value source="$subscription.shipToPostalCode" />
```

2: Display Value

```
<ss:value source="$invoice.shipToPostalCode" />
```

<:shiptwoday>

```
<input type="radio" name="shipme" value="2" />
<ss:value source="$store.shipping.getServiceDescription('2')"/>
```

<:shiptwodayrate>

```
<ss:value source="$cart.getShipping('2')"/>
```

<:sortorder primary="ITEM">

```
<ss:sortby item="ITEM" />
```

<:sortorder secondary="ITEM">

```
<ss:sortby item="ITEM" />
```

<:sortorder tertiary="ITEM">

```
<ss:sortby item="ITEM" />
```

<:state>**1: Form Field, invoice templates**

```
<ss:edit source="$address.state" size="10" />
```

2: Form Field, lead templates

```
<ss:edit source="$lead.state" size="10" />
```

3: Display Value, lead templates

```
<ss:edit source="$lead.state" />
```

4: Display Value, customer templates

```
<ss:value source="$customer.state" />
```

5: Form Field, customer templates

```
<ss:edit source="$customer.state" size="10" />
```

<:statelist country="COUNTRY" default="DEFAULT">

1: Customer, Checkout templates

```
<ss:select source="$system.states('COUNTRY') "
  name="$customer.state" selected="DEFAULT" />
```

2: Ship To templates

```
<ss:select source="$system.states('COUNTRY') "
  name="$address.state" selected="DEFAULT" />
```

3: Lead templates.

```
<ss:select source="$system.states('COUNTRY') "
  name="$lead.state" selected="DEFAULT" />
```

<:statelist country="COUNTRY">

1: Customer, Checkout templates

```
<ss:select source="$system.states('COUNTRY') "
  name="$customer.state" />
```

2: Ship To templates

```
<ss:select source="$system.states('COUNTRY') "
  name="$address.state" />
```

3: Lead templates.

```
<ss:select source="$system.states('COUNTRY') "
  name="$lead.state" />
```

<:status>

```
<ss:value source="$subscription.status" />
```

<:storelist>

```
<ss:select source="$customer.stores()" name="custom1" />
<option value="Select Store" selected>--Select Store--</option>
</ss:select>
```

<:storesaletext>

```
<ss:foreach item="promotion"
  within="$store.activeStorePromotions">
  <ss:if test="$store.isValidPromotion($promotion)">
    <ss:value source="$promotion.invoiceText" /><br>
  </ss:if>
</ss:foreach>
```

<:street1>

1: Display Value, customer templates

```
<ss:value source="$customer.street" />
```

2: Form Field, customer templates

```
<ss:edit source="$customer.street" size="30" />
```

3: Form Field, lead templates

```
<ss:edit source="$lead.street" size="30" />
```

4: Form Field, invoice templates

```
<ss:edit source="$address.street" size="30" />
```

<:street2>

1: Display Value, customer templates

```
<ss:value source="$customer.street2" />
```

2: Form Field, customer templates

```
<ss:edit source="$customer.street2" size="30" />
```

3: Form Field, invoice templates

```
<ss:edit source="$address.street2" size="30" />
```

4: Display Value, lead templates

```
<ss:value source="$lead.street2"/>
```

5: Form Field, lead templates

```
<ss:edit source="$lead.street2" size="30"/>
```

<:street>

1: Display Value, lead templates

```
<ss:value source="$lead.Street"/>
```

2: Display Value, customer templates

```
<ss:value source="$customer.street"/>
```

3: Form Field, customer templates

```
<ss:edit source="$customer.street" size="30"/>
```

<:subcategory>

1: Display Value

```
<ss:value source="$listing.name"/>
```

2: Link

```
<ss:link source="$url.buildUrl('Search')">
  <ss:param name="category" value="$category.parentChainDisplay"/>
  <ss:value source="$category.name"/>
</ss:link>
```

<:subject>

```
<ss:emailheader>
  Enter Text Here
</ss:emailheader>
```

<:submit>

1: Subscription templates

```
<ss:form shortcut="SubmitSubscription"/>
```

2: All other templates

```
<input type="submit" name="submit" value="Submit"/>
```

3: Catalog Detail template

```
<input type="submit" name="submit" value="Add to Shopping Cart"/>
```

<:subscribe>

```
<input type="submit" value="Subscribe"/>
```

<:subscriptionbillnote>

```
<ss:foreach item="subscription" within="$invoice.subscription"/>
  <p><ss:value source="$subscription.billNote"/></p>
</ss:foreach>
```

<:subscriptionno>

```
<ss:value source="$subscription.subscriptionNumber"/>
```

<:subscriptionperiod>

1: Subscription templates

```
<ss:value source="$subscription.subscriptionPeriodText" default="&nbsp;"/>
```

2: Catalog detail templates

```
<ss:if test="$product.subscriptionEnabled">
  <ss:select source="$product.subscriptions" name="productattr"/>
</ss:if>
```

<:subtotal>

```
<ss:value source="$cart.subtotal" default="0"/>
```

<:supplier>	<code><ss:value source="\$supplier.name"/></code>
<:suppliercity>	<code><ss:value source="\$supplier.city"/></code>
<:supplieredicode>	<code><ss:value source="\$supplier.ediCode"/></code>
<:supplierstate>	<code><ss:value source="\$supplier.state"/></code>
<:supplierstreet>	<code><ss:value source="\$supplier.street"/></code>
<:supplierzip>	<code><ss:value source="\$supplier.postalCode"/></code>
<:taxcounty>	<code><ss:select source="\$system. taxCounties(\$address.country, \$address.postalCode)" name="county"/></code>
<:taxregionlist>	<code><ss:select name="taxregion" source="\$store.taxRegionsSelect" selected="\$customer.defaultTaxCode"></code>
<:taxregionselect>	<code><ss:include template="TaxRegionSelect"/></code>
<:total>	<code><ss:value source="\$cart.total" default="0"/></code>
<:totalextensions>	<code><ss:value source="\$extension.total"></code>
<:tracking>	<code><ss:form shortcut="Tracking"/></code>
<:trigger>	<code><input type="hidden" name="trigger-post" value="VALUE"></code>
<:worldpay name="OPTION">	<code><ss:value source="\$worldpay.OPTION"/></code>
<:zip>	

1: Form Field, lead templates

`<ss:edit source="$lead.postalCode" size="10"/>`

2: Display Value, lead templates

`<ss:value source="$lead.postalCode"/>`

3: Form Field, invoice templates

`<ss:edit source="$address.postalCode" size="10"/>`

4: Display Value, customer templates

`<ss:value source="$customer.postalCode"/>`

5: Form Field, customer templates

`<ss:edit source="$customer.postalCode" size="10"/>`

A

About Us Template.....	78
add templates	37
ampersand, editing.....	43
ascii code, when importing templates.....	37
attributedetailfull	63
attributes	
hide	64
show on sale	68, 69
Auction Outcome Email Template	78

B

background colors, change in store	59
baselines	
reset all templates	24
set all templates	23
Billing Options Template	78
borders, template	43
build store	13
business objects overview	9

C

cached templates	36
cart template	
display related products.....	68
Cart Templates	
Cart.....	79
Cart Add.....	79
Cart Save	79
Cart Select.....	79
Cascading Style Sheets	55
Catalog List Templates	
Catalog List.....	80
Catalog List - Members Only.....	81
Catalog List - Not Found	81
Catalog Templates	
Catalog.....	79
Catalog Detail.....	80
Catalog Detail - Members Only	80
Catalog, Members Only	80
Related Products.....	80
categories	
custom templates	57
upload images.....	50
category list	
change display	56
category lists	
collapsible	63
comparison.....	67

Checkout Template	
Deadend	82
New Customers	82
Start.....	81
Closed Template	82
code samples	62
codes, show on templates.....	43
colors	
change.....	14
change in theme	59
change label in storefront	57
conditions	32
context menu	43
copy center	18
copy site	18
County Tax Template	82
create templates	37
Credit Templates	
Email.....	83
HTML	82
CSS.....	55
currency formats	
setup wizard	12
custom templates	57
Customer History Template	83
Customer Properties Template	83
Customer Service Template	83
Customer Sign In Template.....	83
Customer Subscription Template.....	84

D

data	30
date formats	
setup wizard	12
design mode.....	39
design mode editing	41
Design Wizard	10
design, importance of.....	4
documents, additional	4
Download List Template	84
drill down	
Catalog List	65
dynamic pages.....	5

E

eBay Outcome Email Template	78
elements	
property, field reference	38
property, working with	38
Email A Friend Content Template.....	84
Email A Friend Results Template	84

Email A Friend Template.....	84
error messages in Design Studio.....	34
existing sites	
transfer look and feel.....	16
working with StoreSense.....	16
export	
templates.....	36

F

file management.....	55
files	
working with.....	53
files, upload.....	53
fonts	
change.....	14
fonts, change in store.....	59
footer	
hide.....	7
hide, examples.....	56
Footer Template.....	85
forms	
quick search.....	62
Frequently Asked Questions Templates.....	85
FTP area.....	55

G

Generated Account Password - Email Template.....	85
glossary.....	75
guidelines, template.....	43

H

handlebars.....	25
html issues	
Template Sets.....	45
header	
hide.....	7
hide, examples.....	56
Header Template.....	85

I

images.....	29
category, upload.....	50
customizable.....	52
mall, upload.....	50
manufacturer, upload.....	50
product.....	52
product, upload.....	50
shared images.....	50
store.....	51
submit button.....	52
template set images.....	50
troubleshooting.....	72
upload.....	50
working with.....	50
import	

templates.....	37
introduction.....	4
Invoice Template	
Email.....	86
HTML.....	86
Print.....	86
invoice template, display order extensions.....	65

J

JavaScript.....	55
-----------------	----

K

keyboard shortcuts.....	42
-------------------------	----

L

languages, template set detail.....	47
layouts	
change.....	14
Link Builder.....	19
links.....	28
from external site.....	19
to your other sites.....	20
lists.....	31
logo	
add to store.....	59
look and feel.....	14
Lost Password Templates	
E-mail.....	87
HTML.....	87
Low Inventory - Email Template.....	87

M

manufacturers	
custom templates.....	57
upload images.....	50
markers, show on templates.....	43
Member Login Templates	
Error Message.....	88
Login.....	87
Message Template.....	88
meta-data	
add to store.....	59
adding to templates.....	38
meta-tags, creating on static page.....	54
migrate site.....	18
multimedia files.....	55
multimedia files, upload.....	53
multiple templates.....	58
overview.....	57

O

objects	
overview.....	9
OEM	

custom templates	57
upload images	50
Order Confirm Template	88
order extensions	65
Order Extensions Template	88

P

package tracking	96
Packing Slip Template	89
pages, troubleshooting	73
PayPal Purchase Token Templates	89
phone device issues, for Template Sets	45
Policies Template	89
preview	
template set	45
templates	24
price formatting	63
Privacy Template	89
Product Authorization Template	90
product images	52
products	
add to cart	62
add using setup wizard	12
custom templates	57
duplicate search results	74
upload images	50
Promotion Alert Templates	90
property elements	
field reference	38
working with	38
publish templates	24
Purchase Order Limit Template	90

Q

quick search form	62
-------------------------	----

R

Recurring Billing Template	90
redesign store	59
Registration Templates	
Confirmation	91
Register	91
related products	
display on cart	68
reset baselines	24
right-click menu	43
rules for template sets	47

S

Sales Affiliate Templates	
Confirmation	91
Email	91
Registration	91
Sales Lead Templates	
Confirmation	92

Lead	92
select content	25
Select Shipping Template	92
server file management	55
services	
add to cart	62
add using setup wizard	12
custom templates	57
upload images	50
sets	45
setup wizard	
add products	12
custom text	12
design	11
overview	10
settings	12
Setup Wizard	10
shared images	50
Ship To Templates	
Ship To	92
Ship to Gift	93
shipping	
suppress method	65
Shipping - Email Template	93
shortcut menu	43
show borders on templates	43
show codes on templates	43
show markers on templates	43
Site Connector	20
site migration	18
sizes	
change label in storefront	57
source code, editing templates	43
source mode	39
special characters, when importing templates	37
spiders	54
splash pages, create	54
static files	
create	54
css	55
JavaScript	55
multimedia	55
upload	53
working with	53
static pages, create	54
store content area	55
store data area	55
store design, importance of	4
store images	51
Store Locator Template	93
store logo, upload	50
Storefront Template	93
stylesheets	55
subcategories	65
submit button image	52
Subscription Templates	
Billing Options	94
Cancellation Email	94

Recurring Bill Error - Email.....	90
Renewal Email	94
Start	93
Supplier Templates	
EDI	94
EDI 850	95
E-mail	95
Fax.....	95

T

tag reference	
conversion	124
mapping.....	124
overview.....	8
reference	98
Tax Region Select Template	95
template set detail	
apply	47
create custom	47
delete.....	47
field definitions	48
language settings.....	47
overview.....	47
template set images	50
template sets	
activate.....	45
deactivate.....	45
default.....	45
lock visitors	45
maintain.....	45
manage	45
preview	45
redesign.....	45
rules	47
understanding	45
using.....	45
templates	
add	20, 37
ascii code when importing templates	37
baselines, all templates	23
caching.....	36
context menu	43
copy from web site.....	18
design mode	41
design toolbar	42
editing source code.....	43
elements, property	38
elements, property field reference.....	38
export	36
how built.....	7
how displayed	5
import.....	37
keyboard shortcuts	42
manage work	22
multiple	58
preview	24
publish.....	24

resetting all.....	24
right-click menu	43
rules (see also template set detail)	47
saving	34
sets	45
shortcut menu	43
show borders	43
show markers.....	43
special characters when importing templates	37
template assistant	13
template sets.....	45
toolbar	42
undoing	34
user defined	58
validating.....	34
viewing	39
visual aids.....	43
working with	21
test plans	70
testing	70
browsers	70
catalog templates	71
checkout templates.....	71
customer service templates.....	72
equipment	70
membership	71
operating systems	70
plans	70
product templates.....	71
screen resolution.....	70
store templates	70
text.....	28
themes	
change.....	14
change colors	59
choosing.....	16
customize	16
tpl files	36
tracking packages	96
Tracking Template	96
troubleshooting.....	72
images	72
pages	73
validation.....	72

U

undo template changes.....	59
upload	
images	50
multimedia	53
static files	53
store logo	50
UPS, tracking link.....	63
user defined templates	58
User Defined Templates.....	96

V

validating templates	34
validation	
troubleshooting.....	72
view, set defaults.....	39

W

wireless issues for Template Sets	45
---	----

wml issues for Template Sets	45
workspace, Design Studio	21
WorldPay Templates	
Cancel	96
Confirm	97
Error	96
Purchase Token.....	97
WYSIWYG editing.....	41