JShop Server

simple. effective. e-commerce

v2.3.0 Templates Manual

Documentation Version: 2.3.0
(Revision Date: 29/10/2010)

Table of Contents

1 Introduction

1.1 Intended Audience

This document covers the template system, default template set and template variables and attributes. It is intended for those looking for more information on changing the templates for their store and for designers looking to develop template sets for distribution or those wanting to provide their end-users with functionality that can be controlled with template options.

This documentation should be read in conjunction with the Templates section of the main JShop Server User Manual which details the end-user experience in the Templates section of the JShop Server administration system.

2 The Default Templates

2.1 Default Template Set

The default template set is located in the templates/default/ directory. Below is a description of each of the templates and what they are for. They are presented in alphabetical order for ease of location and the .html has been omitted. If you have a version previous to 2.2.0, you can obtain the new default templates from the Registered Users Area. A template set also includes a file called templateset.php in its directory - you should not need to add this unless you want to add some functionality for the template options screens in the administration system. Please see the Developing Template Sets section for more information.

Template	Description
3dsecure	Transferring page for 3D Secure transactions. Only applicable if you are using a direct integration gateway and your gateways supports 3D Secure.
3dsecurewrapper	Wraps your store design around the 3D Secure password entry page if the customer has to do this to authenticate their transaction. Only applicable if you are using a direct integration gateway and your gateways supports 3D Secure.
advsearch	The advanced search page
affiliateaccount	Main affiliate account page
affiliatebanners	Affiliate account page showing banners available
affiliateedit	Affiliate edit account details page
affiliatelogin	Affiliate login page
affiliatepayments	Affiliate account page showing payments made
affiliatesales	Affiliate account page showing sales commissions earned
affiliatesignup	New affiliate signup page
affiliatesignupthanks	Affiliate signup confirmation page
affiliatestats	Affiliate account statistics page
cart	The cart / basket page
cartstockproblem	Page shown if there is a stock problem when the customer tries to logout.
checkoutlogin	First step in checking out where a customer can login etc.
checkoutstep1	Checking out step 1 (billing address)
checkoutstep2	Checking out step 2 (delivery address)
checkoutstep3	Checking out step 3 (payment details)
checkoutstep4	Checking out step 4 (final confirmation)
contact	Contact form
contactsent	Contact form successfully sent
customeraccount	Main customer account page
customeraddress	Add / edit customer delivery address
customeraddresses	List of customer's delivery addresses
customerdetails	Edit customer's account details

customerlogin	Login to customer account
customernew	Create new customer account
customerorder	Showing a customer's order
customerorderlist	List of customer's past orders
customerreview	Customer entering review page
customerreviewerror	Error adding customer review page
customerreviewthanks	Successful adding of customer's review
customerwishlist	Customer's wish list
databaseproblem	Template used if the system cannot connect to the database
downloadproblem	Page shown if the customer's digital product download link is not correct or expired.
forgottenpassword	Forgotten password page
forgottenpasswordsuccess	Successfully sent customer's new password
gatewaytransfer	Page used when transferring the customer to a payment gateway to pay
giftcert_print	Postal gift certificate template
giftcertificate	Gift certificate details page
giftcheckoutstep1	Gift certificate checkout out step 1 (billing address)
giftcheckoutstep3	Gift certificate checking out step 3 (payment details) Note: there is no step 2 for gift certificates
giftcheckoutstep4	Gift certificate checking out step 4 (final confirmation)
help	Main help page
index	The front page for the store
newsletteractivation	Information page shown when signing up for the newsletter if you have the double-opt in option turned on.
newslettererror	Error adding email address to newsletter mailing list
newslettersubscribe	Subscribed to newsletter success
newsletterunsubscribe	Unsubscribed from newsletter success
orderfailed	Order payment has failed
ordersuccess	Order payment has been a success
privacy	Privacy statement
product	The product page
productreviews	Extended customer reviews page for a product
receipt	Order receipt template
search	Search results page
section	The section page
send2friend	The send to a friend page
send2friendsent	Page shown after filling in the send to a friend page
terms	Terms and conditions page

unavailable	Shop is unavailable page

There are also a number of includes that the default templates use to make re-using parts much easier. The includes can be found in the includes sub-directory of the templates directory. Here's what each of the includes shipped with JShop Server does:

Template	Description
address_fields	Contains the loop that outputs all the fields that a customer needs to enter for a delivery address.
affiliate_fields	Contains the loop that outputs all the fields that an affiliate needs to enter for their account details.
breadcrumb	Used to show the breadcrumb trail at the top of all the pages in the store
customer_fields	Contains the loop that outputs all the fields that a customer needs to enter for their main account details.
footer	Contains the footer for all the pages in your store, e.g. copyright statements etc.
leftmenu	This contains all the left menu boxes for your store pages
order_address_fields	As address_fields but for delivery address fields when ordering.
order_customer_fields	As customer_fields but for billing address fields when ordering.
order_extra_fields	Outputs the extra ordering fields on the order confirmation page.
order_giftcertificate	Outputs the fields used for gift certificate ordering
ordering_cart	Outputs the shopping cart. Used in the ordering process.
productblock	The HTML block used to output products on section pages and search pages etc.
productmenulist	HTML formatting used when outputting products on the right menu bar lists
rightmenu	This contains all the right menu boxes for your store pages
search_paging	Includes paging links on the search results page
section_paging	Includes paging links on the section results page
sectionblock	The HTML block used to output sections in your store
top	Create the top and left sections of the store pages.

Includes are a very powerful way to help keep maintenance down to a minimum by allowing you to create only one copy of a particular item. For instance, the standard templates with JShop Server use top.html and bottom.html to create all the navigation for the pages, so if we need to update the navigation we only need to update those files.

Finally in the main templates/ directory is where you'll find a number of special JShop Server templates and order paperwork templates. These are as follows:

Template	Description

giftcert_print	Postal gift certificate template
databaseproblem	Template used if the system cannot connect to the database
receipt	Order receipt template

2.2 Section And Product Page Templates

The default templates include one template for the sections page and one for the product pages in your store. However, against each section and product in your store you can pick which template you would like to use, so it's perfectly possible to have several different section page or product page templates and use different ones on different sections or products. Please see the 'Contents' section of the JShop Server User Manual for more information on how to edit sections and products.

If you change template sets and you currently have some products or sections setup to use templates other than product.html and section.html and these cannot be located in the new template set, JShop Server will automatically display them using the default product.html or section.html template.

2.3 CSS Files

The default template set includes a number of CSS files, mainly for different colour variations of the template set but also for modifications to the core CSS selectors in order to change the width of the site when this is changed in the template options for the default templates. The CSS files are as follows:

- **blue.css** - Full styling for the default templates with a blue colour scheme.
- **red.css** - Full styling for the default templates with a red colour scheme.
- **green.css** - Full styling for the default templates with a green colour scheme.
- **fixedwidth.css** - CSS file used to modify some selectors if you have selected to run your site at a fixed width in the template options for the default template set.
- **fullwidth.css** - CSS file used to modify some selectors if you have selected to run your site at full width in the template options for the default template set.

blue.css, red.css and green.css can be edited freely with JShop Server's in-built CSS editor. We do not recommend editing fixedwidth.css or fullwidth.css using this editor.

2.4 Email Templates

The email templates use exactly the same format for variables as the HTML templates for your actual store but are stored in the JShop Server database. Although information is more limited in availability in the email templates, many of the attributes and variables can still be used. Email Templates are edited by going to Templates -> Email Templates in the JShop Server administration system. Please refer to the JShop Server User Manual for more information on this.

2.5 About page.php

page.php is a special page in JShop Server. It has no particular function apart from showing whatever template you use with it. This means that you can create any number of additional templates for different pages within your site and still have the control that JShop Server gives you. For instance, in the templates that come with JShop Server, we use page.php to show the Privacy, Help and Terms and Conditions pages which enables us to keep the look, feel and store information on screen along with those pages.

To use page.php, you simply enter the following into your template where you want the link to appear:

```
<a href="{page=template.html}">Your Link</a>
```

Using the above method will ensure that JShop Server outputs the URL compatible with your search engine safe URL settings. You can also do the following, however:

```
<a href="page.php?xPage=template.html">Your Link</a>
```

In both examples template.html is the template you want to use to display this page and this is picked up from the directory for the current template set in use.

3 tSys - The Template Language

3.1 Introduction

JShop Server uses an in-built template system for designing your shop's appearance, developed by ourselves called tSys. This template system allows you to include data from your store quickly and easily, whilst keeping everything in html files and provides support for compiling templates to speed up page execution on sites where the design has been completed and remains static.

tSys is designed to be lightweight, easy to pick up and understand, but powerful enough to accomplish most tasks you would need to do in your templates. It also provides the ability to embed PHP directly into the templates so if there is functionality you require in order to achieve something in the templates not provided by tSys, you are free to include PHP as would normally do (in `<?php` and `?>` tags) directly within your templates. We would always recommend that you try to stick to tSys as much as possible and only break out into PHP where necessary. For those that need this functionality, please refer to the Embedding Variables in PHP section of this documentation which provides the format you should use for including normal template variables directly in your own PHP blocks.

In order to increase the speed of the template system, all commands, properties, variables and attributes are case-sensitive. This saves the template system having to do case insensitive matches.

3.2 Variables and Attributes

tSys uses the notion of variables and attributes that, whilst they may appear to be the same thing, are essentially different to tSys. A variable is a top-level collection of attributes. Take the following example, designed to output the name of a product on a product page template:

```
{product.name}
```

`product` is the variable, `name` is the attribute. It's important to remember this, especially when using things like the set directive, as you can only set or get values that are an attribute of a variable.

Within a variable, however, there can be other variables as well as attributes. For instance, `{product.pricing.quantitytable.entries}` is a variable which contains a collection of entries that can be looped around to output their attributes. Please see the loop section for more information on looping.

When used within tSys language directives, the { and } brackets are dropped from the variable/attribute reference, e.g.:

```
<#if:customer.loggedin:eq:Y#>
       Hey, welcome back {customer.firstname}
<#/if#>
```

This checks the value of `{customer.loggedin}`.

3.3 Variable Formatting Options

When outputting a text variable you can limit the number of characters that are displayed, like this:

```
{sections.shortdescription:10}
```

The above example will limit the number of characters of the section's short description to 10 characters, stripping out any HTML tags before hand. In addition you can add a further part to the formatting, like this:

```
{sections.shortdescription:10:dots}
```

This will not only limit the output to 10 characters but also add 3 dots to the end if the short description is actually longer than 10 characters.

In addition to 'dots', the following formatting options can also be used:

- **dots_keeptags** – The same as dots but does not strip out HTML tags
- **keeptags** – The same as using just a length limiter but does not strip out HTML tags

Note: These should only be used on string type variables, e.g. any number variables should not be treated in this way.

3.4 Language Reference

3.4.1 include

The include directive makes it easy to re-use templates on multiple pages. For instance, in the templates that are installed with JShop Server, we use an include to include top.html (which contains the header bar) and footer.html (which contains the footer for your store). This means we only have to keep one file up to date, to update these parts of the page. Of course you can use includes as much as you like and most people will find it useful to keep parts of the page you want to re-use in separate include files.

To include a file, you do the following:

```
<#include:includes/top.html#>
```

This will include the template top.html, replacing the include directive with the contents of top.html. Any variables or other directives within top.html will also be recognised and used. You'll notice that the example refers to a sub-directory called includes. All include directives are processed relative from the current template set's directory, e.g. templates/default/

3.4.2 loop

The loop command loops round a collection of records, outputting the HTML contained between the `<#loop#>` and `<#/loop#>` directives. For instance, on many pages a variable called products is available (for instance, on a section page). You can do the following to loop through all the products:

```
<#loop:products#>
       <br/>{products.name}
<#/loop#>
```

This will loop through all the products, outputting the product name for each.

When you setup a loop there are two special variables that are created. For instance in the example above we also have access to:

`{loop.products.total}` – the total number of records
`{loop.products.count}` – the current number of times the loop has been run (starting at 1).

So, working on the example above, we can change this to put a number in-front of each of the products that's output:

```
<#loop:products#>
       <br/>{loop.products.count}. {products.name}
<#/loop#>
```

Now, the product names will be output with a 1 in front of the first one, 2 in front of the second one etc. In addition, we could give a total count underneath the products:

```
<#loop:products#>
	<br/>{loop.products.count}. {products.name}
<#/loop#>
<p>Total Products: {loop.products.total}</p>
```

Note that the total variable can be used outside the loop and it will still remember the total number of records that was output.

There is a second part to the loop directive which can be excluded (as we have done in the above examples) or included, as below. This is the limit amount and, if included, it will limit the number of times the loop runs to this value.

```
<#loop:products:5#>
	<br/>{products.name}
<#/loop#>
```

The above example will limit the output to 5 products, no matter how many records are available. This is useful if you never want to show more than a certain number of records in a situation – for example, you may want to limit the bestsellers list to no more than 10 products.

3.4.3 if

For those of you not familiar with programming languages, an if statement has a test and does something if that test is true. For instance:

```
<#if:customer.loggedin:eq:Y#>
	Hey, welcome back {customer.firstname}
<#/if#>
```

The above example tests to see if the variable `{customer.loggedin}` is equal to Y. If it is, it will output the 'Hey, welcome back `{customer.firstname}`' HTML, replacing `{customer.firstname}` with the customer's first name. But the if statement also has another part to it, that means you can do something else if the test isn't true. Expanding on the above example, we could do the following:

```
<#if:customer.loggedin:eq:Y#>
	Hey, welcome back {customer.firstname}
<#else#>
	You're new here. Why don't you open an account with us!
<#/if#>
```

We've used the `<#else#>` directive in the above example to output something different if the customer isn't logged in.

In addition there is and `<#elseif#>` directive that can be used like this to chain together decisions:

```
<#if:product.scLevel:eq:0#>
	Out of stock!
<#elseif:product.scLevel:lt:5#>
	Only a few left!
<#else#>
	We've got lots of them!
<#/if#>
```

So what tests are available? We've seen the use of eq to test if a variable is equal (the same as) a value, but you can also use the following tests:

- **eq** - tests if the values are equal
- **neq** - tests if the values are not equal
- **blank** - tests if the value is blank
- **even** - tests if a number is even
- **odd** - tests if a number is odd
- **mod** - performs modulus on a number
- **lt** - tests to see if a variable is less than a value
- **lte** - tests to see if a variable is less than or equal to a value
- **gt** - tests to see if a variable is greater than a value
- **gte** - tests to see if a variable is greater than or equal to a value
- **starts** - tests to see if a string starts with certain characters
- **nstarts** - tests to see if a string does not start with certain characters
- **ends** - tests to see if a string ends with certain characters
- **nends** - tests to see if a string does not end with certain characters
- **contains** - tests to see if a string contains certain characters
- **ncontains** - tests to see if a string does not contain certain characters

3.4.4 set

The set directive allows you to quickly and easily copy a variable to another variable for use in your templates. It is ideal for creating re-usable include files across the whole of your site. For instance, a variable collection in the templates can be copied to a new variable and existing code re-used:

```
<#if:loop.randomproducts.total:gt:0#>
	<#loop:randomproducts#>
		<#set:prod:randomproducts#>
		<#set:counter:loop.randomproducts.count#>
		<#include:includes/listoutput.html#>
	<#/loop#>
<#/if#>

<#if:loop.specialoffers.total:gt:0#>
	<#loop:specialoffers#>
		<#set:prod:specialoffers#>
		<#set:counter:loop.specialoffers.count#>
		<#include:includes/listoutput.html#>
	<#/loop#>
<#/if#>
```

You'll see this used extensively in the default templates for outputting the product lists on the right part of the screen. This allows you to maintain a single piece of HTML to output similar styles for different collections. In addition you can also use the set directive to set static values:

```
<#set:myvar.myattribute#>this is a string<#/set#>
```

You can also include existing attributes within such a set directive:

```
<#set:myvar.myattribute#>this is a<#/set#>
<#set:myvar.myattribute#>{myvar.myattribute} string<#/set#>
```

This would result is `{myvar.myattribute}` outputting 'this is a string'.

When using set to create an attribute you should use the {variable.attribute} format, rather than just {attribute} for it to operate correctly. The only exception to this is when you're wanting to set a group of attributes to a variable, as in the first example in this topic.

3.4.5 math

The math directive allows you to perform mathematical operations on numeric variables and attributes and set the result to another variable. For example:

```
<#math:myvar.myattribute#>20<#/math#>
<#math:myvar.myattribute#>20*5<#/math#>
<#math:myvar.myattribute#>20*{loop.products.total}<#/math#>
```

You can re-use variables already set with math, like this:

```
<#math:myvar.myattribute#>20<#/math#>
<#math:myvar.myattribute#>{myvar.myattribute}/2<#/math#>
```

When `{myvar.myattribute}` is used in a template it would output 10 in the above example.

In addition there is a second argument for the math function that allows you to set the rounding, e.g.:

```
<#math:myvar.myattribute:integer#>20/4.5<#/math#>
```

Would give a value of 4 for `{myvar.myattribute}`. You can use 'integer', 'int' or 0 to output as an integer or any other number to output a required number of decimal places, e.g:

```
<#math:myvar.myattribute:2#>20/4.5<#/math#>
```

Would give a value of 4.44 for `{myvar.myattribute}`

Standard arithmetic operators are supported, e.g. * / % + - along with parenthesis.

3.4.6 option

The option directive let's you set tSys options directly in the templates. There are currently two options implemented, although this will expand in the future.

```
<#option:striphtmlcomments:true#>
```

A new feature in tSys is the stripping of HTML comments which does have a new option in the JShop Server admin system but can also be set using the options directive. Set it to true if you would like HTML comments removed, false otherwise. Setting this in the template will, with the new version of JShop Server, override the setting within JShop Server's admin system.

```
<#option:contentheader:text/html:iso-8859-1#>
```

This option lets you set the mime type and character set for the document and is output as a PHP `header()`. You can omit the character set if you wish to so `<#option:contentheader:text/html#>` is also a valid setting for this option. Obviously the text/html and iso-8859-1 can be set to your own choice.

3.5 Standard Variables

tSys has access to some special variables that you can use in your templates. These are provided as-is and have not been sanitized in any way so if you are using them directly in your templates you should ensure that you sanitize them as required using PHP in your templates.

Variable: browser

This contains a few attributes to help you identify the currently browser being used to view your store.

- **{browser.short}** - JShop Server's own short identification for the browser viewing the site, e.g. IE for Internet Explorer, FF for Firefox.
- **{browser.long}** - A longer identification including version number, e.g. IE6, IE7 etc.
- **{browser.version}** - A numeric representation of the browser version, e.g. 7, 5.5 etc.
- **{browser.agent}** - The full agent as provided by the browser.

The routines/browserDetect.php file contains a complete list of the browsers currently detected by JShop Server and what the equivalent entries will be in the browser variable.

Variable: form

This variable contains any POST or GET information submitted to the page being viewed. This is provided as-is with no sanitization.

- **{form}** - All POST and GET variables together, e.g. a field called 'id' by either GET or POST would be available as `{form.id}`
- **{form.get}** - Separate collection of just GET variables, e.g. a field called customer submitted by GET would be available as `{form.get.customer}`
- **{form.post}** - Separate collection of just POST variables, e.g. a field called name submitted by POST would be available as `{form.post.name}`

Variable: server

This simply contains a collection of all the server environment variables available to PHP. It is a direct copy of PHP's `$_SERVER` global variable.

3.6 Embedding Variables in PHP

If you wish to include PHP in your templates and you need access to some of the attributes available in the template system you should prefix the variable with a *, like this:

```
<?php
	$myVariable = {*product.name};
?>
```

This will ensure that the tSys template system outputs the correct variable code, otherwise parse errors will occur.

3.7 Error Messages

JShop Server includes some error trapping in the tSys template system which will alert you to directives that have not been used correctly, e.g. the wrong number of parameters or a parameter that is not recognised. However, we cannot capture everything and occasionally you will see PHP parse errors like the following:

```
Parse error: parse error in c:\Development\JShopServer\routines\tSys.php(332) : eval()'d
code on line 387
```

This parse error basically means that there is a problem with the PHP that your template is converted into. The line number refers to lines in the compiled version of the template and not in the original template itself. Generally these are due to opening a loop and not closing it, or opening an if statement and not closing it.

3.8 Compiled vs Uncompiled Templates

JShop Server's template system includes the ability for you to pre-compile your templates and for it to store this pre-compiled version. This makes them run much, much faster, than if they were uncompiled, which requires that on each page load the template by parsed and the PHP version of it created and then run with your data. For compiled templates, the stored compiled version is opened and run with your data.

When a template is compiled by JShop Server, it is basically converted into native PHP. This runs very fast when a page is requested and adds only a few milliseconds onto the CPU time it takes a page to be created and sent to the user. It is always advisable that you run your production store from compiled templates as this basically means that taking the template and converting it into PHP (which is the time consuming part) is only done once. When the page is requested again the native PHP version is opened instead.

There are some things you need to make sure of if you're going to use compiled templates.

1. PHP must have full read/write access to the 'compiled' directory for the template set you are using.
2. Making changes to your templates will not automatically take effect on the site unless you force the templates to recompile (see the 'Templates' section of the documentation for more information and the 'Overriding Normal Template Settings' section further on).

3.9 Embedding tSys in JShop Server Fields

With the release of 2.3.0, tSys variables and commands can now be embedded in any field within the admin system. For instance, you may wish to link directly to a product from a main section description. If you had a section with an ID of 2, you could then add a link to the section description as follows:

```
<a href="{section=2.link}">{section=2.title}</a>
```

JShop Server would then parse this as normal and replace the variables with the correct information. This change has been possible due to the addition of the new multiple parsing mechanic of tSys in 2.3.0 which provides the ability to parse a template more than once. Please see the next section for more information on this.

3.10 Multiple Parsing & Delayed Parsing

Version 2.3.0 introduces the concept of multiple template parses into JShop Server. This has been done to allow the embedding of template variables and commands into any field in JShop Server, making it easier to link to products or sections, for instance, within description fields etc. This also means that it's now possible to construct template variables from other template variables, allowing much more complex execution in templates. This facility is intended for experienced users.

A simple example, where a productID is passed via. a GET or POST form field is as follows:

```
{product={form.productid}.name}
```

On the first template parse {form.productid} would be replaced by the content of the form field "productid" - let's say we're passing the value 2 in this field. On the second template parse tSys would parse the following:

```
{product=2.name}
```

And thus output the name attribute for the product.

Delayed Parsing

The above simple example is of limited use but tSys now also has the ability for you to tell it it to delay the parsing of a command or variable. This enables you to only run template code once a variable has been constructed. Delayed parsing is activated by adding extra # signs to the start of a template command or variable. The number of # signs you add tell tSys how many parses you want to skip before the code is run. Here's an example:

```
<##set:myvar.product:product={form.productid}#>
<##if:myvar.product.name:neq:blank#>
  The product has a name {#myvar.product.name}
<##/if#>
```

As with the first example the first parse would change {form.productid} to the value in the GET or POST form field. None of the rest of the code would be parsed or run as every command and variable has a # in front of it. So after the first run we have the following:

```
<#set:myvar.product:product=2#>
<#if:myvar.product.name:neq:blank#>
  The product has a name {myvar.product.name}
<#/if#>
```

On the second run the rest is parsed and run - delaying the parsing has ensured that {myvar.product} has actually been created before the rest of the code runs. Generally, if you wish to do complex things in this manner it's easier to do the following:

```
<##set:myvar.product:product={form.productid}#>
<##include:includes/mystuff.html#>
```

In the mystuff.html include file you don't need to add extra # signs in as this won't be included until the correct point and so the content of mystuff.html can be normal tSys:

```
<#if:myvar.product.name:neq:blank#>
  The product has a name {myvar.product.name}
<#/if#>
```

4 Template Settings

4.1 Template Settings in JShop Server

Please refer to the Templates section of the JShop Server User Manual for descriptions of the options available in this section of the administration system. It provides the ability to change the mode that the templates are being run in (compiled/uncompiled) and changing some configuration options for how tSys operates (including stripping comments etc.)

4.2 Overriding Normal Template Settings

The xTFC command, when used with a URL will override the normal template settings – but only for you, e.g.:

```
http://www.yourdomain.com/index.php?xTFC=1
```

This will force the template system to run in Force Compile mode.

Likewise, there is an xRTU command which when used like this:

```
http://www.yourdomain.com/index.php?xRTU=1
```

This will force the template system to run in Uncompiled mode.

Both of the above settings can be removed by using 0 as the value, e.g.

```
http://www.yourdomain.com/index.php?xTFC=0
```

And this will reset your session to using the normal template settings.

4.3 Setting Different Template Directories

There's no reason why you can't run the same shop with different template sets, to give you completely different output. For instance, you may have two different domain names and want the shop to have different branding depending on which domain name the customer has used. This is possible with JShop Server, by using a template command with the URL as so:

```
http://www.yourdomain.com/index.php?xTemplates=mydesign/
```

This example would use the template set located in templates/mydesign/ for the shop's look, rather than the default template set directory.

For reverse compatibility with versions prior to 2.2.0 you can still use new main template directories, e.g. http://www.yourdomain.com/index.php?xTemplates=templates2/ would also check for a directory called templates2 in the main JShop Server directory. We do not recommend continuing the use of such main template directories (these will not appear for template selection in the admin system and no support for template options or the CSS editor will be available). You should move these directories so they are within the main templates/ directory as this previous functionality should now be considered depreciated and we cannot guarantee continued support for it going forwards.

4.4 The xForce Command

Although not directly related to the templates themselves, xForce is a useful feature that enables you to start a new cart session.

Adding xForce to the end of your URL, like this:

```
http://www.yourdomain.com/index.php?xForce=Y
```

will force a new cart to be created for your session. This is useful if you have made changes to things such as the default tax as it will reset your session to the defaults.

5 Template Variables And Attributes

5.1 How JShop Server's Variables Work

This section contains detailed information on the variables and attributes that are available in your templates for displaying form fields, products, section, the shopping cart etc. Taken on their own this section may be confusing at first which is why we suggest that you spend time looking over the default templates that are installed with JShop Server.

As you know JShop Server is ready "out of the box" and is a fully functioning web store from the moment it is installed. The templates are complete and correct and will probably give you better knowledge if they are examined in conjunction with this documentation.

There is no way that we can cover all the possibilities within the documentation regarding the template system – there are simply too many ways that you could choose to display information, so you should expect some trial and error. Some of the pages in JShop Server, such as customer login forms etc. only really have one set of data, all of which should be used. Those pages and variables aren't included in this documentation and the default template set should be used as your reference. As we said in the previous topic, the best way to learn about the variables available is to look at the templates – we've tried to include all those variables and attributes in this documentation that may not appear in the default templates and to give you an overall feel for how data is structured for your templates.

In addition, some things such as the options variable just have too many attributes to include in the documentation although again we may be able to include these in the future. Finally, however, it is simply impossible to cover every piece of data and every possible way of using that data in this documentation.

5.2 Outputting Variables For Debugging

A nice little trick if you want to see exactly what attributes you have available for a given variable, is to place something like this temporarily at the top of your template:

```
<?php
	print_r($this->theVariables["cart"]);
?>
```

This would output all the attributes and sub-attributes of the cart variable to the page which you can then look through (this is normally best done through 'View Source…' as it's much easier to read this way with proper indentation).

5.3 Multiple Languages

Internally JShop Server finds the correct alternative language version for all those fields that can be specified in multiple languages and outputs the version that should be used in the standard attribute name for the language a customer is viewing your store with. For example with the products, `{product.name}` will always contain the applicable language version of the product name. If the customer is using one of the alternative languages you have setup and an alternative language version of the product name wasn't found, then the default language version would be used.

For all other text displayed on your store, JShop Server utilises labels, which can be maintained for all languages in the JShop Server administration system in Templates -> Labels. The default template set that comes with JShop Server uses labels exclusively for all text, so if you setup an alternative language in your store you can enter the alternative language versions for all labels quickly and easily.

5.4 General Form Notes

All forms within JShop Server are created with normal form tags but using variables sent from JShop Server to the template system. This ensures that the forms are constructed correctly with the correct form action. You can add your own onSubmit validation to any of the forms if you wish and, you can also change the form method from POST to GET should you need to. JShop Server is flexible in how form information can be submitted.

In the default templates, on many of the customer fill-in forms, such as account details or the contact form etc. we use a loop to display the form elements. This has only been done this way for ease of creating the default templates. However, you can access form fields individually should you want to. For instance, the contact form's group of fields is as follows:

`{contactform.fields}` – This is the group of contact form fields that can be looped around.

In addition to loop method of accessing form fields, you can also access them directly. For instance with the contactform, if you had a field called EmailAddress you can access the attributes of this field like this:

```
{contactform.field.EmailAddress.titleText}
```

This will show the title text for the EmailAddress field. Presenting fields in this manner gives you the freedom to construct your forms however you wish.

Form fields generally have the following attributes:

Attribute	Values	Description
titleText		The title of the field
fieldname		The internal field name of the field
validation	0 or 1	If set to 1 this field has validation and will be checked when submitted
validationmessage		Any applicable validation message for the field
fieldtype	TEXT or TEXTAREA or CHECKBOX or SELECT	The internal type of the field. You should ensure that fields are displayed in the correct manner.
size		The size of the field (only available for TEXT)
maxlength		The maximum number of characters that can be entered into the field (only available for TEXT)
cols		The number of columns to display (only available for TEXTAREA)
rows		The number of rows to display (only available for TEXTAREA)
content		The actual content of the field if any already exists, e.g. when editing account details this will contain the current value of the field.
error	Y or blank	If set to Y then this field has been rejected by the validation and you can use this to show the validationmessage.
selected		The currently selected item (only available for SELECT)

options		The group of options (only available for SELECT)

The best way to work out how JShop Server handles form fields is to look at the default templates. Using the default templates as a guide you should find it easy to change how form fields are displayed on your store's pages.

Using the direct method, reordering form fields, deleting fields and adding new fields in the JShop Server administration system will not take effect unless you manually add them into your templates. With the loop method obviously this does not occur as the system simply loop around all the form fields available in the correct order.

5.5 Available Variables

5.5.1 Products

Below is a list of the main attributes available for products.

Attribute	Values	Description
productID		Internal product ID
code		Product code
name		Product name
shortdescription		Short description
description		Full description
thumbnail	Image + Image path	Thumbnail image
mainimage	Image + Image path	Main product image
visible	Y or N	Product visibility status
metaDescription		Meta Tag Description
metaKeywords		Meta Tag Keywords
keywords		Search keywords
templateFile		Template used for product page
newproduct	Y or N	New Product status
topproduct	Y or N	Top Product status
scEnabled	Y or N	Stock control enabled or disabled for this product
scLevel		Actual stock level
scWarningLevel		Stock warning level
scActionZero		Action to perform is stock falls below 1 unit.
productType	N	Only N at the moment, reserved for future use.
weight		Weight of the product
taxrate	0, 1 or 2	Tax rate for the product
freeShipping	Y or N	Does product have free shipping.
specialoffer	Y or N	Is product on special offer.

price	Formatted price	This is the product price and will include tax if the customer account type that the user is assigned to has this option set.
priceextax	Formatted price	This is the product price excluding any tax
priceincax	Formatted price	This is the product price including any tax
pricetax	Formatted price	This is the tax price for the product
rrp	Formatted price	This is the normal price and will include tax if the customer account type that the user is assigned to has this option set.
rrpextax	Formatted price	This is the normal price excluding tax
rrpinctax	Formatted price	This is the normal price including tax
rrptax	Formatted price	This is the tax amount for the normal price
rrpDifference	Formatted price	This is the difference between the price and normal price fields
rrpPercent		This is the percentage difference between the price and normal price fields
inCart	Y or N	Flag showing if the product is in the customer's cart or not.
ooprice	Formatted Price	This is the one-off product price and will include tax if the customer account type that the user is assigned to has this option set.
oopriceextax	Formatted Price	One-Off price excluding tax
oopriceinctax	Formatted Price	One-Off price including tax
oopricetax	Formatted Price	Tax amount for the One-Off price
isDigital	Y or N	Flag showing if the product is a digital product or not
allowDirect	Y or N	Flag showing if the product can be viewed direct, even if the product is set to be invisible.
ignoreDiscounts	Y or N	Flag showing if the product is excluded from customer account type discounts
minQty	Number	Minimum quantity that can be ordered.
maxQty	Number	Maximum quantity that can be ordered.
groupedProduct	Y or N	Flag showing if the product is a group product or not.

5.5.1.1 Product Extra Fields

There are two ways in which the extra fields for products can be accessed.

First of all there is a group of extra fields. This is what the standard JShop Server templates use. It ensures that all extra fields are picked up and displayed (if applicable to a particular product). The main group of extra fields is `{product.extrafields}`

Secondly, the extra fields can be accessed directly by their field name. For instance, if you had an extra field called autograph, this can be accessed like so:

`{product.extra_autograph.title}` – to display the title of the autograph field.

Notice that extra_ has to be added in front of the extra field's name for it to be picked up correctly.

Which ever way you access the extra fields, the following attributes are available for each extra field:

Attribute	Values	Description
name		The extra field's name
title		The extra field's title
type	TEXT, TEXTAREA, SELECT, IMAGE, CHECKBOXES and RADIOBUTTONS	The extra field's type. Note: You should ensure that you display the extra field correctly according to its type, otherwise this may cause incorrect functioning of the automatic price update scripts.
content		The content of the field for this product.
options		Only for SELECT, CHECKBOXES and RADIOBUTTONS types. This is a group of options that can be looped around. Each option has the following attributes: option – the text for the option exvalID – the internal option ID which must be used as the value sent to the cart.

The extra field attributes are only available on the product page and only for the product that is being viewed. They are not available for each product in a group.

5.5.1.2 Product Flags

If you have setup any extra product flags using the Product Flags option in the Contents section of the administration system, you can access these in the template system via. a sub-group called flags. For example if you had a flag called isLimited then you could use the following to check the value:

```
<#if:product.flags.isLimited:eq:Y#>
       This product is available in limited amounts
<#/if#>
```

5.5.1.3 Add to Cart Form

You can include the Add To Cart form for a product anywhere where you list a product and, obviously, on the product page itself. For each product page the form is started like this:

```
<form name="{product.form.name}" action="{product.form.action}" method="POST"
onSubmit="{product.form.onsubmit}">
```

This ensures that the correct form is created for the product. If you look at the default templates you'll see that on the section pages we use the following for the 'Add To Cart' link:

```
<a href="{products.add.link}" class="middle-links">
```

This is a special variable created for each product and basically it submits the product form. In addition to this you can use a normal submit button on the form, as the default product.html template does.

If you do not wish to include a quantity box on your pages (as we do on the section pages in the default

templates) you do not have to and JShop Server will assume a quantity of 1 when the product is added to the basket. Alternatively, the following will create a correctly named quantity box:

```
<input type="text" size="5" value="1" class="textbox" name="{product.qtyboxname}"
onBlur="{product.recalculateprice}">
```

The onBlur part of this uses a built-in link to the JavaScript recalculate price function that is simply used to update the price on the product page when options are selected. You should not include the onBlur part if you use a quantity box on any page other than actual product pages. Of course, if you wanted to you could use a select box instead of a text field for the quantity box – the display is up to you as long as the field is called the correct name, e.g `{product.qtyboxname}`

5.5.1.4 Product Links

The following are other in-built product links that you can use for products in your store:

```
<a href="{products.link}" class="middle-links">
```

This is a link that takes the customer directly to the product's product page. You would normally use this where a product is listed – for instance in the random products section etc.

```
<a href="{products.wishlist.link}" class="middle-links">
```

This is a link that lets the customer add the product to their wish list.

5.5.1.5 Option Tables

There are 4 extra variable collections as part of the {product} variable that can be used on the product page templates to display option information about the products, for instance quantity discount information, option-level stock control etc. These variables are as follows:

Complete information for these tables is available on the default template set's product.html template where they are output, including checks to determine whether or not they should be output for a given product (they are not shown if there is no applicable information for the product being viewed.)

- **`{product.quantitytable}`** - Contains quantity discount information for display
- **`{product.combinationstable}`** - Displays attribute combination information for display
- **`{product.exclusionstable}`** - Displays any attribute combination exclusions for display
- **`{product.stocktable}`** - Displays option-level stock levels for the product if applicable

The quantitytable variable is used to display any quantity discounts you have for an individual product. The default templates show the discount amount for each quantity breakdown using the following:

```
{product.pricing.quantitytable.entries.discount}
```

Atrtibutes discountExTax, discountIncTax and discountTax are also available. However, it is possible to show the price after discount by using this instead:
`{product.pricing.quantitytable.entries.priceDiscounted}`

Attributes priceDiscountedExTax, priceDiscountedIncTax and priceDiscountedTax are also available.

Importantly, however, you should note that this price is calculated from the main price from the product and any options etc. that a product may have will not be included in this price. It also does not update as the normal price fields do on the product page when options are selected that affect a product's price.

5.5.1.6 Customer Reviews

Available on the individual product pages the customer reviews variable, accessed in the form `{reviews.attribute}`, has the following attributes:

Attribute	Values	Description
enabled	Y or N	Whether customer reviews are enabled or not
total		Total number of customer reviews for this product
content	Group	The group of reviews, see below for more information
averagerating		The average rating given to this product

The 'content' attribute is a group containing all the reviews for the product and you should use a loop to access each individual review. The attributes of the content variable, accessed in the form of {reviews.content. attribute} are as follows:

Attribute	Values	Description
name		Review author's name
rating		The rating this review gave to the product
title		The title of the review
review		The actual review itself

5.5.1.7 Groups of Products

There are a number of different groups of products available in JShop Server. These are as follows:

Group	Where Used	Description
products	Section pages	Group of products in a section (section. html) and on the main root section in index.html
randomproducts	Anywhere	Pick out a list of random products
specialoffers	Anywhere	Group of special offer products
bestsellers	Anywhere	Automatically generated group of best selling products
newproducts	Anywhere	Group of new products
topproducts	Anywhere	Group of top products
recommended	On a Product page	Automatically generated group of recommended products based on order database
associated	On a Product page	Group of associated products
groupedproducts	On a Product Page	Collection containing all products if current product is a group product.

As you have previously seen attributes for single products are accessed by using the `{product.attribute}` format. With a group of products, you first need to create a loop and then access the attributes. For instance, the following outputs the list of randomproducts:

```
<#loop:randomproducts#>
       {randomproducts.name}<br>
<#/loop#>
```

You see that rather than use `{prdoduct.attribute}` we use `{group.attribute}`, in this case { `randomproducts.name}`.

Products in a group have access to the same attributes that are available in the normal single product method, with the exception of any extra fields that can only be displayed using the `{product.attribute}` method on a product page.

5.5.1.8 Other Ways to Show Products

If you wish to pick out individual products then you can use a special variable to do so. Normally the product variable on a product page is constructed like this:

`{product.code}` – to display the product code.

You can also do this, however, to show a particular product's variable anywhere:

`{product=144.code}` – to display product ID 144's product code

In this way you can easily pick out particular products in places where you want them to show.

5.5.2 Sections

Below is a list of the main variables available for sections.

Attribute	Values	Description
sectionID		Internal section ID
title		Section title
thumbnail	Image + Image path	Thumbnail image
image	Image + Image path	Main section image
metaDescription		Meta Tag description
metaKeywords		Meta Tag keywords
shortDescription		Section's short description
fullDescription		Section's full description
parent		Internal section ID of parent section
templateFile		Template used for section page
totalproducts		Outputs the total number of products in a section. Does not include any products in sub-sections.

The following are in-built section links to ensure that you link to other sections correctly:

```
<a href="{sections.link}">
```

The example above is used with the sections group and accesses the link URL for the customer to be taken directly to the section.

5.5.2.1 Groups of Sections

There are a number of different groups of sections available in JShop Server. These are as follows:

Group	Where Used	Description
rootsection	Anywhere	Group of the top level sections in your store.
subsections	In conjunction with rootsection	A group of the next level of sections for each of the sections in rootsection. Only available if activated within the General section of the administration system
sections	On section page	Group of sub-sections for the currently viewed section.

5.5.3 Cart

Cart is the variable that is used to display both the full cart and, in the default templates, the always shown "mini-cart" on the right hand navigation menu. You can access the cart variable on any of your template pages.

Below is a list of the main attributes available for the cart:

Attribute	Values	Description
products		Group of the products in the cart.
totals		Group of totals for the shopping cart (these are only ever accessed individually).
emptylink		URL Link to empty the shopping cart of all products.
currencyID		Internal currency ID for the cart
accTypeID		Internal customer account type ID
languageID		Internal language ID for the customer

5.5.3.1 Fields Available For Products

Not all the available fields for a product are available for list of products in the shopping cart and there are some extra ones specific to the cart itself. The following are available:

Attribute	Values	Description
productID		Internal product ID
code		Product code
name		Product name
Shortdescription		Short Description
thumbnail	Image + Image path	Thumbnail image for the product
link		URL Link to the product page for this product.
extrafields		Group of option fields with selected options (see next section).
qty		Quantity for the product
qtyboxname		Form name for the quantity box

price	Formatted Price	Price for the product. If customer account type shows prices including tax, tax will be included in this price.
priceextax	Formatted Price	Price excluding tax for the product
priceinctax	Formatted Price	Price including tax for the product
pricetax	Formatted Price	Tax amount for the product
total	Formatted Price	Total cost (price x qty)
deleteItemLink		URL Link to delete the item from the cart

Although structured in the same way as the normal product extrafields variable the one available for products in the cart does not include any non-option fields (e.g. no IMAGE, TEXT or TEXTAREA fields). In addition it only includes the selected options for the product and not the complete list of options for each field. In all other respects it is accessed in the same way. Again these extrafields can be accessed directly using the same method as is used for the products variables.

5.5.3.2 totals Variable

The totals variable contains the following attributes:

Attribute	Values	Description
goods	Formatted Price	The goods total for the cart
discount	Formatted Price	The special discount total for the cart
order	Formatted Price	The current final ordering total (excluding shipping and tax if customer account type does not include tax in the standard prices).

5.5.3.3 The Cart Form

The cart form is used for the 'Update Cart' button (which, in the default templates, is simply a submit button). The form is started like this:

```
<form name="{cart.form.name}" action="{cart.form.action}" method="POST">
```

The form should surround all the products on the page to ensure that the quantity boxes are shown and updated correctly when the 'Update Cart' button is clicked.

5.5.3.4 Cart Error States

There are a number of error states that are added as attributes to the cart variable if there is a problem when adding a product to the cart. You can use these on the cart page to display appropriate error messages. These are detailed below:

Attribute	Values	Description
error	EXCLUDED	The product with the options selected could not be added to the basket as the combination is excluded.

In addition there are also error states that are added if the customer tries to checkout but JShop Server will not let them.

Attribute	Values	Description
checkouterror	EMPTY	Customer could not check out as they have an empty cart.
checkouterror	VALUE	Customer could not check out as they have not reached the minimum order amount (if applicable)

5.5.4 Company

Below is a list of the main attributes available for the company (the values for these fields are setup in General -> Company Details and General -> Meta Tag Details in the administration system).

Attribute	Values	Description
companyName		Your company name
addressLine1		First line of your address
addressLine2		Second line of your address
city		Postal city
county		Postal county / state
country		Postal country
postcode		Postcode / ZIP
telephone		Telephone number
fax		Fax number
storeurl		The store URL
metaAuthor		Meta Tag Author
metaDescription		Meta Tag Description
metaKeywords		Meta Tag Keywords

5.5.5 Customer

Below is a list of the main attributes available for the customer (if the customer is logged in).

Attribute	Values	Description
customerID		Internal customer ID
accTypeID		Internal customer account type ID
title		Customer's title
forename		Forename
surname		Surname
address1		Address line 1
address2		Address line 2
town		Town
county		County / State
postcode		Postcode / ZIP

telephone		Telephone number
fax		Fax number
email		Email address
company		Company name
newsletter	Y or N	Whether the customer is subscribed to your newsletter or not.
loggedin	Y or N	Whether the customer is actually logged in or not.

Any extra customer fields you have setup will also be available directly by the field name you used in Customers -> Customer Fields.

5.5.5.1 Customer Links

The following customer link attributes are available so you can include navigation links for customers.

Attribute	Values	Description
registerlink		URL that takes an unregistered customer to the customer registration screen.
forgottenlink		URL that takes a customer to the forgotten password screen.
homelink		URL that takes a logged in customer to their account home page.
addresseslink		URL that takes a logged in customer to their delivery addresses page.
orderlink		URL that takes a logged in customer to their order history page.
wishlistlink		URL that takes a logged in customer to their wishlist
logoutlink		URL that logs a customer out.

5.5.6 Addresses (Delivery Addresses)

On the customer address editing page this is available as `{customer.addressfields}` and on the ordering section this is available as `{order.address}`. It contains all the field information for delivery addresses both stored with an order and in the customer's address book. It is structured in the same way as the customer record and the form fields are displayed in the same way as described in 20.6 General Form Fields.

In addition to the pages that display individual address, such as adding and editing a customer address or adding a new delivery address when ordering, there is a group of delivery addresses called addresses which becomes part of the customer variable on the main customer address screen that lists all the addresses in a customer's address book (customeraddresses.html).

The addresses group adds the following attribute links to each address in the customer's address book:

Attribute	Values	Description
editlink		URL link that takes the customer to the editing screen for an address.

deletelink		URL link that allows the customer to delete an address in their address book.

In addition to this the customer variable also obtains a link attribute called `{customer.addressaddlink}` that takes them to the editing screen to add a new address into their address book.

5.5.7 Wishlist

Wishlist is the variable that is used to display the customer's wish list.

Below is a list of the main attributes available for the wishlist:

Attribute	Values	Description
products		Group of the products in the cart.
emptylink		URL link to empty the wishlist of all products.

Not all the available fields for a product are available for list of products in wishlist and there are some extra ones specific to the wishlist itself. The following are available:

Attribute	Values	Description
productID		Internal product ID
code		Product code
name		Product name
thumbnail	Image + Image path	Thumbnail image for the product
link		URL Link to the product page for this product.
extrafields		Group of option fields with selected options (see next section).
qty		Quantity for the product
qtyboxname		Form name for the quantity box
commentboxname		Form name for the comments box
comment		Any comment that the customer has entered against the product.
price	Formatted Price	Price for the product. If customer account type shows prices including tax, tax will be included in this price.
priceextax	Formatted Price	Price excluding tax for the product
priceinctax	Formatted Price	Price including tax for the product
pricetax	Formatted Price	Tax amount for the product
total	Formatted Price	Total cost (price x qty)
deletelink		URL Link to delete the item from the wish list
AddToBasketLink		URL Link to add the product to the current basket.

5.5.7.1 Wishlist Form

The wishlist form is used for the 'Update Wishlist' button (which, in the default templates, is simply a submit button). The form is started like this:

```
<form name="{wishlist.form.name}" action="{wishlist.form.action}" method="POST">
```

The form should surround all the products on the page to ensure that the quantity boxes are shown and updated correctly when the 'Update Wishlist' button is clicked.
In addition to the normal wishlist form there is a Send Wishlist form as well. The form is started like this:

```
<form name="{wishlist.send.form.name}" action="{wishlist.send.form.action}" method="POST">
```

On this form a required field is the list of email addresses that the customer would like to send their wish list to. This can be included with the following:

```
<input type="text" name="{wishlist.send.form.emaillist}" value="" class="textbox"
size="50">
```

5.5.8 Ordering

The `{ordering}` variable (which is used for both normal ordering and gift certificate ordering although the templates for these are different) inherits from many of the other variables available in JShop Server. For instance, the customer's address fields are available either to an existing customer, a new customer or to a customer that is not opening an account (JShop Server treats them all the same in this respect and asks that the same customer fields are filled in). Delivery addresses are available as well for an existing customer to select one from their address book, to select their billing address as the delivery address or to enter a completely new delivery address.

One exception to this is cart which can be addressed in exactly the same way as on the normal cart pages.

There are, of course, extra fields shown on the cart which are not available on the normal cart page for the tax totals, gift certificate totals and shipping totals, but these work in much the same way as the normal total fields for the cart.

In addition there are groups for things such as shipping methods and payment options and in this regard the templates should be self explanatory from all that you have seen. For instance the credit card fields for the Credit Card payment option can be accessed directly as was discussed in General Form Notes

The ordering process is fairly static although it can be altered via. various options within JShop Server. For instance, you can opt not to allow separate delivery addresses, or to not use the gift certificate system and the ordering pages are setup to check for these occurrences.

You may obviously wish to change the layout of the pages but in terms of functionality they can be pretty much left as they are (as long as the if statements are left in) and they will alter themselves to suit your settings.

5.5.9 Order

`{order}` is a variable that contains order information. It is used on the order success page, the customer order history page (only the order header information is available in the group `{orderlist.orders}` that is used on the main customer order history list page) and when printing a receipt in the order administration system.

Attributes for the main order variable are as follows:

Attribute	Values	Description
orderID		Internal order ID
ordernumber		Internal order ID plus you base order number

		setting
orderdate		Formatted order date
ordertime		Formatted order time
totals		Group containing order totals
products		Group containing order product information
ip		IP address saved with the order
customerID		Internal customer ID (if applicable)
All customer fields		All customer fields – see the Customer part of this section for information on the attribute names available.
All delivery address fields		All customer delivery address fields
currencyID		Internal currency ID of the order
status		Status of the order. Please see the Order Management section of this documentation for more information.
shippingMethod		Name of the shipping method used. There is also shippingMethodNative which is stored in the language the customer checked out in.
paymentID		Internal payment ID for the order
paymentName		Actual name of the payment method. There is also paymentNameNative which is stored in the language the customer checkout out in.
orderPrinted	Y or N	Whether or not the order has been printed
orderNotes		Internal Order Notes field
languageID		Language customer used when ordering
giftCertOrder	Y or N	Y if the order was for a gift certificate
Any extra order fields		Any extra order fields referenced by field name

The order list group contains a list of orders and is used on the main customer order history page. Information is as above for the main order variable (without any product information) with the addition of an attribute called viewlink which creates a link to the full order page for any given order.

5.5.9.1 Totals Group

The totals group contains the following attributes:

Attribute	Values	Description
goods	Formatted Price	The goods total
shipping	Formatted Price	The shipping total
tax	Formatted Price	The tax total
discount	Formatted Price	The special discount total
giftcertificates	Formatted Price	Total of any gift certificates used on the order
order	Formatted Price	Final order total

5.5.9.2 Products Group

The products group contains the following fields:

Attribute	Values	Description
productID		Internal product ID
code		Product code
name		Product name. On the front-end this will be in the language the order was placed in.
extrafields		Group of option fields with selected options (see next section).
qty		Quantity for the product
price	Formatted Price	Price excluding tax.
total	Formatted Price	Total cost (price x qty)

Product extra fields are in the same format as the extra fields for the cart variable. When the customer views their order all option names will be in their native language which is saved along with the product details at the time of ordering.

5.5.10 Languages

languages is a group containing all the languages setup in your store and it is normally used to display a language selection feature on your store. Attributes for each language are as follows:

Attribute	Values	Description
languageID		Internal language ID
name		The actual language name.
link		A URL link that changes the language the customer is viewing the store in.

Only visible languages will be included in the group.

The cart variable contains an attribute called languageID that contains the languageID of the currently selected language.

5.5.11 Currencies

currencies is a group containing all the currencies setup in your store and it is normally used to display a currency selection feature on your store. Attributes for each currency are as follows:

Attribute	Values	Description
currencyID		Internal currency ID
code		ISO 3-character code for the currency
name		Descriptive name for the currency
decimals		Number of decimal places
pretext		Formatting pre-text
middletext		Formatting middle-text

posttext		Formatting post-text
useexchangerate	Y or N	Whether the currency is calculated via. an exchange rate or not.
exchangerate		The applicable exchange rate.
checkout	Y or N	Whether checking out in the currency is allowed or disallowed.

Only visible currencies will be included in the group.

The cart variable contains an attribute called currencyID that contains the currencyID of the currently selected currency.

5.5.12 Snippets

Snippets are accessed in a slightly different way to normal variables as you need to pick out individual snippets, rather than the whole database of snippets. If you have a snippet called News, you can display the title and the content of the snippet by including the following on your templates.

```
{snippet=News.title}
{snippet=News.content}
```

5.5.13 Labels

All labels are automatically made available to your templates by JShop Server. They are accessed in the following format:

```
{labels.type.labelname}
```

Where type = the label type, e.g. navigation for example and labelname = the actual label name. The default templates are setup to use labels throughout and you can add / edit / remove labels from the Templates section of the administration system.

5.5.14 Options

Options is a special variable that contains all the configuration options from the jss_options database table. This isn't used a great deal in the default templates but can be employed to provide some powerful conditional statements based on the internal settings of your JShop Server store.

There are many different options stored in the jss_options database table and these are made available in the following format:

```
{options.optionName}
```

Where optionName = the actual option name from the jss_options database table.

For instance you could do the following to check if customer accounts are enabled for your store:

```
<#if:options.customerAccounts:eq:1#>
       Customer accounts are enabled!
<#else#>
       Customer accounts are disabled!
<#/if#>
```

There are too many options in the jss_options database table to go into details on here but they all have been given very descriptive names.

5.5.15 Paging Variables

Paging variables become available on the search page for products and on the section page for products. They provide access to any number of pages in the results, along with Previous and Next links.

The following attributes become part of the sectionpages variable on section pages and the search variable on search results pages:

Attribute	Values	Description
page		Currently viewed page
pages		Group of page links
previouslink		URL link to the previous page (blank if not applicable)
nextlink		URL link to the next page (blank if not applicable)

The pages group should be looped around and contains links to all the other pages available. Attributes of pages are:

Attribute	Values	Description
page		Page number
link		URL link to the page

5.5.16 News

The news group contains all the Latest News items you have entered for your store. The attributes available for each news item are as follows:

Attribute	Values	Description
newsID		Internal unique ID for the news item
title		Title of the news item
content		Content of the news item
date		Formatted date of when the news item was first created
time		Formatted time of when the news item was first created
postedBy		The username of the person who created the news item.

Note: In addition to the news group there is also a group called newstitles. This group contains much the same information and is used in the same way. The only difference is that this group does not include the content attribute so is useful when you simply want to create a list of news items that link through to your full news page. This gives a speed benefit as the content field is never retrieved from the database.

5.5.16.1 Linking to News Items

By default a template called news.html is used to display the full content of all your news items in the store. This page is setup to be used with page.php, like this:

```
page.php?xPage=news.html
```

A link created as above will show the news.html template and show your news items. news.html includes anchors for each of the new items, like this:

```
<a name="news{news.newsID}"/>
```

As you can see this is created by the word news followed by the unique ID for the news item. In the default templates top.html also contains a list of news items for the left hand menu bar in your store. This links to individual news items like this:

```
page.php?xPage=news.html#news{newstitles.newsID}
```

All this is doing is showing the news page and going straight to the anchor for the particular news item the customer has clicked on.

5.5.17 Users Online

The usersonline variable contains information about the number of users currently looking at your store and the most ever online at once. Attributes are:

Attribute	Values	Description
current		Current number of users on your site
timelimit		The length of time (in minutes) used to calculate the number of users online
most		The most number of users ever recorded online at once
mostdate		The date when the most number of users online was recorded

5.6 Paperwork Templates

If you wish to setup extra paperwork templates for printing when processing orders, the basic format is the same as for the receipt template. However, there are a couple of extra variables that you can use and some CSS tricks you can use to output page breaks, making the production of labels, for instance, much easier.

The two extra variables you have access to are:

- `{process.pagecount}` – This gives the number of the current order you're printing, e.g. for the first selected order this will be 1, the second 2 etc.
- `{process.totalcount}` – This gives the total number of orders in the print run.

Using The Variables To Print Page Breaks.

By using the `{process.pagecount}` and `{process.totalcount}` variables you can print multiple orders per page and only issue page breaks to the printer according to your paperwork, e.g. printing 2 orders per page, or labels. It is best to start your paperwork template with something like the following:

```
<#if:process.pagecount:eq:1#>
	<html>
		<head>
			<title>Paperwork</title>
		</head>
		<body>
<#/if#>
```

This will only output a HTML header if we're printing the first order. Correspondingly you can put the following at the bottom:

```
<#if:process.pagecount:eq:process.totalcount#>
		</body>
	</html>
<#/if#>
```

That will only output the HTML footer if we're printing the last order in the list.

By using the mod feature of tSys you can select when you wish to output page breaks. Something like the following will output a page break only every 2 orders:

```
<#if:process.pagecount:mod2:0#>
	<div style='page-break-before:always'></div>
<#/if#>
```

Of course, as in the default templates, you can use mod as well to output multiple columns, which would be suitable for printing order labels, for instance.

All Paperwork templates should be stored directly in the main templates/ directory as this is where JShop Server will look for them. If they are not in this directory they will not be available for selection in the JShop Server administration system.

6 Developing Template Sets

6.1 Introduction

This section of the documentation is specifically for those looking to develop their own templates sets either for distribution or for a specific site where you want to provide the end-user with the ability to customise the templates in some way through template options and the in-built CSS editor in JShop Server. It is important to read this section fully to ensure that the template set you create adheres to the way that JShop Server expects the information about the templates to be presented and the way the CSS editor behaves when saving updated CSS selectors.

Within the main templates directory sub-directories reside, each of which contains a complete template set. Each template set will have the following files:

- **templateset.php** - This is a special XML file that JShop Server reads to determine basic information for the template set and expose any options to the end-user that are available for editing in the JShop Server administration system.
- **thumbnail.png** - This is a 250 x 150 PNG image displayed as a thumbnail to identify the template set in Templates -> Installed Template Sets in the JShop Server administration system
- **HTML files** - These are the templates themselves and all main content pages should be covered, in addition to any include files placed in a sub-directory called 'includes'
- **CSS files** - These are the css files for the template set and should be placed in a directory called 'css' for consistency.
- **Image files** - These are the image files for the template set and should be placed in a directory called 'images' for consistency.
- **compiled directory** - A blank directory called 'compiled' should be included for completeness.

If you look at the organisation of the default JShop Server template set you'll see that it follows this basic structure and we'd encourage any designers developing template sets for distribution to adhere to this basic structure.

6.2 Creating a templateset.php File

The templateset.php file is really the glue that binds your template set together and provides information to the end-user via. the JShop Server administration system and exposes any template options you have setup. If you take a look at the templateset.php file for the default template set that comes with JShop Server, it looks like the following:

Default templateset.php

```
<?php exit; ?>
[[ DEFINITION ]]
<template>
	<header>
		<title>JShop Server Default Templates</title>
		<author>Whorl Ltd.</author>
		<description><![CDATA[This is the standard, default template set for JShop
Server. Options on this template include the selection of different colour style sheets,
logo upload and product display options (columns or rows.)!]]></description>
		<url>http://www.jshop.co.uk/</url>
		<version>2.2</version>
	</header>
	<fields>
		<field>
			<name>logo</name>
			<display>Your Logo</display>
			<type>image</type>
```

```
                    <help><![CDATA[Upload your own logo for the top left of the site. For
this template we recommend a size no bigger than 400 pixels wide by 70 pixels high.]]></
help>
                    <default>images/logo.png</default>
             </field>
             <field>
                    <name>css</name>
                    <display>Choose Colour Scheme</display>
                    <type>select</type>
                    <options>
                           <option value="blue.css">Blue (css/blue.css)</option>
                           <option value="red.css">Red (css/red.css)</option>
                           <option value="green.css">Green (css/green.css)</option>
                    </options>
                    <help><![CDATA[The name in brackets tells you which CSS file will be
used. There is one for each colour set and these can be edited by going to 'Installed
Template Sets' and selecting 'Edit Templates']]></help>
                    <default>blue.css</default>
             </field>
             <field>
                    <name>sitewidth</name>
                    <display>Site Width</display>
                    <type>select</type>
                    <options>
                           <option value="full">Full width of the browser (css/fullwidth.
css)</option>
                           <option value="fixed">Fixed width of the browser (css/
fixedwidth.css)</option>
                    </options>
                    <help><![CDATA[You can select with a full width layout or a fixed
width layout. Depending on which one you use an extra css file that alters some of the css
classes will be included in your template. For fixed width, ensure that you fill in the
option below as well.]]></help>
                    <default>full</default>
             </field>
             <field>
                    <name>pixelwidth</name>
                    <display>If Fixed width, enter width in pixels</display>
                    <type>text</type>
                    <length>8</length>
                    <maxlength>6</maxlength>
                    <characters>integer</characters>
                    <help><![CDATA[If you have selected fixed width above please enter the
width in pixels you would like for your layout below. The site will be centered on the
screen. Don't forget to take into account the scroll bars on browsers, so rather than
entering 800 pixels, for instance, you should enter 774.]]></help>
                    <default>774</default>
             </field>
             <field>
                    <name>pdfront</name>
                    <display>Front Page Product Display</display>
                    <type>select</type>
                    <options>
                           <option value="single">Single column, full width</option>
                           <option value="double">Two columns, half width each</option>
                    </options>
                    <help><![CDATA[Use this option to change the display of products on
the front page of your store to use either one column or two columns]]></help>
                    <default>single</default>
             </field>
             <field>
```

```
                    <name>pdsection</name>
                    <display>Section Page Product Display</display>
                    <type>select</type>
                    <options>
                           <option value="single">Single column, full width</option>
                           <option value="double">Two columns, half width each</option>
                    </options>
                    <help><![CDATA[Use this option to change the display of products on
section pages of your store to use either one column or two columns]]></help>
                    <default>single</default>
             </field>
             <field>
                    <name>pdsearch</name>
                    <display>Search Page Product Display</display>
                    <type>select</type>
                    <options>
                           <option value="single">Single column, full width</option>
                           <option value="double">Two columns, half width each</option>
                    </options>
                    <help><![CDATA[Use this option to change the display of products on
the search page of your store to use either one column or two columns]]></help>
                    <default>single</default>
             </field>
       </fields>
</template>
```

The templateset.php file should reside in your main template set directory, not in any of the sub-directories, otherwise it will not be found by JShop Server.

The basic structure of any template sets you create should remain the same with the top two lines left intact and the general tag structure retained.

The header tags should be fairly self-explanatory and these fields are generally for display on the Templates -> Installed Template Sets section of the administration system to inform the end user.

The fields collection is the part that sets up template options for the template set and there are several different types of field that can be included, which are gone through in turn in the sub-topics. There are some common tags for each field, however, and these are as follows:

- **<name>** - this is the internal name for the option and should not include anything other than alpha-numeric characters. JShop Server will only display and use fields that stick to this naming convention.
- **<display>** - this is the text that is displayed to the end user for the option when editing in the admin system.
- **<help>** - if provided, this will provide a tooltip help of the text you enter on the options screen in the admin system.
- **<default>** - this is the default value you'd like to use when the template set has options first setup for it by JShop Server.

All the fields will accept valid CDATA comment tags and strip these automatically for display or use.

6.2.1 Field Type: text

This provides a text entry field in the admin system for the user to enter any text you need them to. These fields should always have a `<length>`, `<maxlength>` tag and a `<characters>` tag. The `<characters>` tag limits what can be input, using client side JavaScript.

Here's an example of a text field that we want to allow no more than 6 characters for and will only allow the numbers 0-9 to be entered into those characters:

```
<field>
	<name>pixelwidth</name>
	<display>If Fixed width, enter width in pixels</display>
	<type>text</type>
	<length>8</length>
	<maxlength>6</maxlength>
	<characters>integer</characters>
	<help><![CDATA[If you have selected fixed width above please enter the width in
pixels you would like for your layout below. The site will be centered on the screen.
Don't forget to take into account the scroll bars on browsers, so rather than entering 800
pixels, for instance, you should enter 774.]]></help>
	<default>774</default>
</field>
```

The `<length>` tag dictates how wide the text entry field should be on the options page and the `<maxlength>` tag dictates how many characters can be entered.

Valid options for the <characters> tag are as follows:

- **alpha-numeric** - allows A-Z, a-z and 0-9
- **general** - allows anything to be entered
- **email** - allows an email address to be entered
- **url** - allows a URL to be entered
- **decimal** - allows a decimal number to be entered
- **integer** - allows an integer (whole number) to be entered

6.2.2 Field Type: select

This provides a select box (drop-down) to allow the end-user to make a selection from a number of options. Here's an example select field with two options:

```
<field>
	<name>sitewidth</name>
	<display>Site Width</display>
	<type>select</type>
	<options>
		<option value="full">Full width of the browser (css/fullwidth.css)</option>
		<option value="fixed">Fixed width (css/fixedwidth.css)</option>
	</options>
	<help><![CDATA[You can select with a full width layout or a fixed width layout.
Depending on which one you use an extra css file that alters some of the css classes will
be included in your template. For fixed width, ensure that you fill in the option below as
well.]]></help>
	<default>full</default>
</field>
```

The `<options>` group should surround all the options you wish to make available for the select box and, for each option, you should ensure that you provide both a value (which will be the value that's actually stored for the selection) and more description text between the `<option>` and `</option>` tags which will be displayed to the end-user.

6.2.3 Field Type: image

This will provide an image upload widget for the user, similar to the standard image upload widgets available throughout the JShop Server administration system. Images are automatically uploaded to the local images directory for the template set, e.g. the 'images' directory within your template set's directory.

```
<field>
```

```
	<name>logo</name>
	<display>Your Logo</display>
	<type>image</type>
	<help><![CDATA[Upload your own logo for the top left of the site. For this template
we recommend a size no bigger than 400 pixels wide by 70 pixels high.]]></help>
	<default>images/logo.png</default>
</field>
```

If you provide a default for this field you should ensure that you actually include the image in your template set's distribution zip file.

6.2.4 Field Type: yesno

This provides yes no radio buttons to the user to make a selection from and one of them is guaranteed to be select. This will be stored as either Y or N against the template option when retrieved for use in templates.

Here's an example of a yesno field:

```
<field>
	<name>sitewidth</name>
	<display>Would you like your site the full width of the browser?</display>
	<type>yesno</type>
	<default>Y</default>
</field>
```

6.2.5 Field Type: color

This will provide the user with the standard colour picker in JShop Server. Please note that you can enter this as either of the following in your field definition:

```
<type>color</type>
<type>colour</type>
```

Here's an example colour field:

```
<field>
	<name>favcolour</name>
	<display>Favourite Colour</display>
	<type>color</type>
	<help><![CDATA[Please pick a colour that you like.]]></help>
	<default>#0000ff</default>
</field>
```

6.2.6 Field Type: separator

Rather than being a field that can be edited by the user, this field type is used to provide separation between your options, so you can display them more as logical groups. Only the name and display tag are required for this field and it will cause a horizontal bar to be output with the display contents, splitting up the fields you are providing to the end-user.

Here's a simple example:

```
<field>
	<name>swotitle</name>
	<display>Site Width Options</display>
	<type>separator</type>
</field>
```

The `<help>` tag will not be parsed for this field type even if it has content.

6.3 tSys templateset Collection

A variable is available in the templates called `{templateset}` which contains all the information about the template set, including some special attributes that JShop Server creates and all the user-selected options that you may be using for your templates. Important attributes are as follows:

- `{templateset.title}` - will provide the title tag data
- `{templateset.path}` - this is very important and will always provide you with a relative URL to the location your templates are being run from
- `{templateset.options}` - this collection contains all the field options that you've provided to the end-user to edit via. the administration system. For instance, if you had setup a field called sitewidth, the value for this would be available in `{templateset.options.sitewidth}`

`{templateset.path}` should be used whenever you reference CSS files, javascript files or images from your template set in the individual templates. This provides you with the path to your template set's directory from within a template and so can be used to reference your CSS and image files, regardless of the name of the directory they are stored in. For instance, in the new default JShop Server templates, you'll see the following in includes/top.html:

```
<link rel="stylesheet" href="{templateset.path}css/basic.css" type="text/css"/>
```

Using that method means that no matter what the name of the directory your template set is running from (for instance, maybe somebody has multiple copies of the template set in differently named directories) you'll always be referencing the correct location.

6.4 How Options Are Stored

It helps to understand how JShop Server stores and template set options you've exposed to the end-user through the administration system when developing a template set, especially if you issue updates to a template set and want to retain previously selected options. Basically JShop Server, when it first comes across a new template set, will create an entry in the jss_options database table with a special name to reflect that template set's options. The format is as follows:

```
ts_hash_directory
```

- **hash** is an MD5 hash of the `<title>` and `<author>` tags in the templateset.php file, so it's important that these fields always stay the same if you update the template set you have created otherwise previously selected options will be lost.
- **directory** is the name of the directory the template set is stored in, relative to the main templates/ directory, e.g. templates/default would have a directory of 'default'

This naming convention enables JShop Server to automatically create options for new template sets but also provides the ability to use the same template set in multiple directories, each with their own template options set. It also avoids uploading two different template sets at different times to the same directory having a cross-over of options.

6.5 CSS Editor Support

The final piece of the puzzle when creating a template set is ensuring that your CSS files are compatible with JShop Server's CSS editor. Although you are more than welcome to indicate in your templateset.php file's description field if they are not, for the best end-user experience we would suggest ensuring that they are. The CSS editor itself, and the parser we have written, is designed to pick up on a couple of special comment lines to help break up the display of selectors to the end user and provide context-sensitive help for any selectors you wish to describe more fully. The two types of comment are as follows, and should be embedded directly in your CSS files:

```
/* HEADER: General Tags &amp;amp; Text */
```

A comment that starts with HEADER: will create a heading in the left hand list of selectors in the CSS editor. This is good for providing better separation between selectors for the end-user, and grouping logical parts together.

```
/* HELP: Basic body background for your store. */
```

A comment that starts with HELP: will display a blue help icon against the selector that comes immediately after it in the selector list in the CSS editor. This allows the provision of extra explanations about what each selector does.

We'd suggest taking a look at one of the CSS files for the default templates as they use many headers and each selector has help associated with it.

In addition to using comments as above you need to be aware of some limitations in the parsing ability of the CSS parser we've developed when creating your CSS files, and some information on how it will output selectors again when saving. These are currently as follows:

1. The editor will convert shorthand tags, such as background, font, border, margin, padding into long forms when writing out edited selectors
2. Comments within a selector will not be preserved
3. The editor will lose any import or media lines and is not currently designed to pick these up - we suggest using separate CSS files for different media and avoid using imports.
4. The CSS Editor will not allow you to add styles - it's designed to edit styles that already exist
5. There is an additional CSS attributes section at the bottom of the editing screen where any properties the editor is not designed to pick up will be placed for editing

We are definitely interested in being provided with any valid CSS file that has problems with the CSS editor. If you have such an example please let us know and we'll be happy to look into it.

Index

U

V

W

X

Y