

Table of Contents

| | |
|--|----------|
| Course: Sitellite XT Templating | 1 |
| <u>1. Syllabus</u> | 1 |
| <u>Course instructor</u> | 1 |
| <u>Difficulty level and prerequisites</u> | 1 |
| <u>Course description</u> | 1 |
| <u>Course objective</u> | 1 |
| <u>Course outline</u> | 1 |
| <u>2. What are XT templates?</u> | 2 |
| <u>What is Sitellite?</u> | 2 |
| <u>What does XT stand for?</u> | 2 |
| <u>What does XT do?</u> | 2 |
| <u>Separation of content vs. templates</u> | 4 |
| <u>3. Using Sitellite XT templates</u> | 4 |
| <u>Where do the templates reside?</u> | 4 |
| <u>SiteTemplate: The Sitellite template editor</u> | 5 |
| <u>Building a simple XT template</u> | 5 |
| <u>4. Advanced XT capabilities</u> | 9 |
| <u>Advanced XT expressions</u> | 9 |
| <u>Simple loops</u> | 11 |
| <u>Conditional rendering</u> | 11 |
| <u>Other XT tags</u> | 11 |
| <u>Why use XT programming?</u> | 12 |
| <u>When to use XT vs. PHP code</u> | 13 |
| <u>5. How to get help</u> | 13 |
| <u>Sitellite.org</u> | 13 |
| <u>Simian.ca</u> | 13 |

Course: Sitellite XT Templating

1. Syllabus

Course instructor

Name John Luxford

Email lux@simian.ca

Web Site <http://www.simian.ca/>

Difficulty level and prerequisites

This course assumes a comfortable level of familiarity with the HTML markup language, and is intended for web design and development professionals.

Course description

An introduction and overview to developing web site templates using the XT templating language.

Course objective

To familiarize the student with the various concepts as well as the ordinary usage of the XT template language, as well as to provide the steps necessary for more advanced XT templating.

Course outline

1. Syllabus
 1. Course instructor
 2. Difficulty and prerequisites
 3. Course description
 4. Course objectives
 5. Course outline
2. What are Sitellite XT templates?
 1. What is Sitellite?
 2. What does XT stand for?
 3. What does XT do?
 1. What is content in a CMS?
 2. Content types
 3. Template language
 4. Simple server-side templating language
 4. Separation of content vs. templates
3. Using Sitellite XT templates
 1. Where do the templates reside?
 1. Template sets
 2. The template naming convention
 1. Output modes
 2. Multiple templates

- 3. Templates and CSS
- 4. XHTML transitional
- 2. SiteTemplate: The Sitellite template editor
- 3. Building a simple XT template
 - 1. The template editor screen
 - 2. The default XT template
 - 3. Inserting page elements
 - 4. XT expressions
 - 5. Inserting boxes
 - 6. Checking for standards compliance
- 4. Advanced XT capabilities
 - 1. Advanced XT expressions
 - 1. Path expressions
 - 2. String expressions
 - 3. PHP expressions
 - 2. Simple loops
 - 3. Conditional rendering
 - 4. Other XT tags
 - 1. Multi-lingual text
 - 2. Changing attributes
 - 3. Outputting HTML entities
 - 5. Why use XT programming?
 - 6. When to use XT vs. PHP code
- 5. How to get help
 - 1. Sitellite.org
 - 2. Simian.ca

2. What are XT templates?

What is Sitellite?

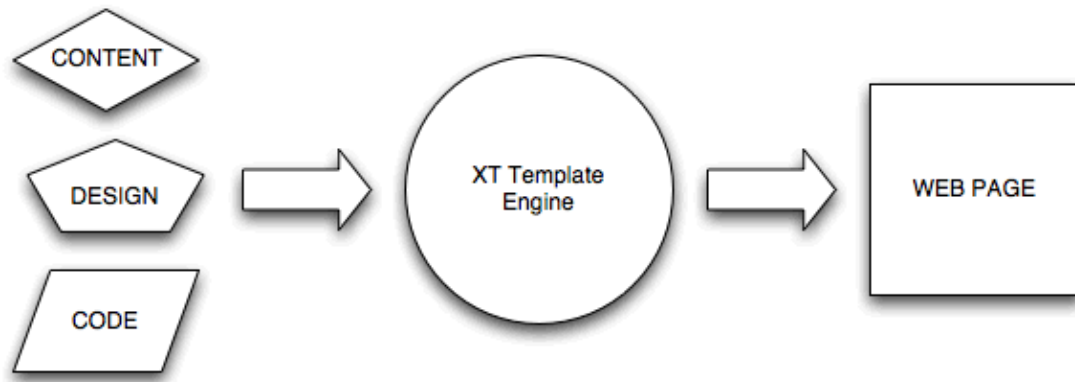
Sitellite is a web site content management system (CMS), which means that it provides the means for creating, publishing, and managing content on a web site. A CMS provides interfaces for the various roles in a web site project to come together, including the programmers, designers, administrators, and the content creators.

What does XT stand for?

XT stands for XML Templates, because XT templates are simply XML documents which are a combination of XHTML and specialized XML tags. XT is the template language and rendering engine used by the Sitellite CMS to create web pages.

What does XT do?

Sitellite uses the XT template engine to compile the different components of the web site user interface together for the web site visitors, including the content, the graphic design, and any programming code required to render a given web page.



What is content in a CMS?

Content means any piece of information on the site that conveys particular meaning (ie. the specific subject matter of a page), in contrast to the graphic design, which provides general or relational meaning (ie. web site navigation) and brand reinforcement.

Content types

In Sitellite, there can be any number of content types. Typically, these will include:

- Web Pages
- Web Files (ie. PDFs, Word documents, etc.)
- News Stories
- Event Listings
- Sidebars (ie. lead-ins to other content types)

Template language

XT templates are ordinary XHTML documents with the addition of XT-specific tags that are replaced with actual content by the template engine.

Simple server-side templating language

The need for XT is that XHTML on its own doesn't have the ability to talk to external sources, such as the Sitellite content repository, to retrieve content. So with XHTML, every page has to have the content completely mixed in with the design, or else you have to use some sort of intermediate processor to put the two together on-the-fly.

Traditionally this was done using programming languages such as Perl, Java, or PHP, however these are far too complicated and usually only result in mixing the design with the programming code instead of the content, which is still a problem.

To solve this problem, we created XT, an intermediate language which is much closer to standard XHTML and therefore less complex than a full-blown programming language, but still powerful enough to solve the problem of joining the content with the graphic design of a web site.

Separation of content vs. templates

The traditional method of simply storing each page in HTML format breaks down quickly because the content writers need to either bother themselves with HTML, or need to rely on the graphic designers to actually apply the content changes. This adds an unnecessary step to each and every content change made to a web site.

So the benefit of using a template language such as XT is that by storing your web site content separately from your web site graphic design, you make it possible for the people responsible for each of the two to manage their part themselves.

3. Using Sitellite XT templates

Where do the templates reside?

XT templates live in the "inc/html" sub-folder in any Sitellite installation.

Template sets

Templates are organized into multiple "sets", which typically contain completely separate designs. In this way, the Sitellite CMS user interface can use the same template organization as the rest of the web site, but still have its own graphic design. You can have as many designs for as many applications as you want running on top of the Sitellite platform.

The Template Naming Convention

Inside a template set, templates are stored as individual ".tpl" files that contain the XHTML and XT markup. A single template set can have multiple templates within it, which are organized according to output mode and by template name.

Output modes

Output modes allow you to offer different output formats on a web site, instead of just XHTML. This way, you can accommodate different types of visitors as new technologies become available, such as cell phones and PDAs. You can also provide alternative file formats such as PDF output for your web site using output modes.

Multiple templates

Even within the same output mode (ie. XHTML), you may still want some pages to look different than others. For example, your main index page might have more branding and imagery and less content than inside pages, and some inside pages may require more width than others. Sitellite allows you to do this as part of the standard template naming convention.

Templates and CSS

Any good design nowadays contains more than just XHTML, but rather a design consists of two distinct parts. The first, the XHTML, is meant to describe the document more so than to control the display of it. The second, the CSS, is meant to apply style to the different markup tags in the markup document. In this way, you have even more control over your design, and more meaningful markup (ie. metadata) about your documents as

well.

In Sitellite, CSS files live in the template set folders alongside templates. They are considered of equal importance to a complete web site graphic design.

In fact, because XT templates must be valid XML (ie. XHTML instead of HTML), Sitellite actually encourages additional standards compliance that are otherwise not required of most CMSes.

XHTML transitional

XHTML is the newest version of the HTML markup language. The reason for the added "X" is because XHTML is HTML with a few new added rules that allow it to be XML compliant.

XML compliance allows HTML documents to better interoperate with new technologies, as well as other XML-based markup languages, such as Scalable Vector Graphics (SVG), MathML, and XT.

By requiring XHTML compliance in Sitellite's templates, we were about to create the XT template language using only a standard XML parser, which makes template parsing much more stable and predictable.

SiteTemplate: The Sitellite template editor

XT templates can be edited directly using any text editor and a browser, an HTML design tool such as Dreamweaver, or via Sitellite's built-in template editor, named SiteTemplate.

The advantage of using XML as the basis for XT is that, unlike most templating languages, XT templates can be previewed by opening them up directly in a web browser, and can be edited safely by any modern HTML editor (such as Dreamweaver).

However, Sitellite also provides its own method of managing and modifying templates as well, which is a web-based template editor, complete with full preview capabilities, XT tag insertion toolbar, and validation against a number of standards, including:

- XML syntax validity
- XHTML compliance
- Web Accessibility Initiative (WAI) compliance
- Section 508 Accessibility compliance

You can also make changes to and verify the validity of CSS files as well.

Building a simple XT template

To begin using the SiteTemplate XT template editor, first open a web browser and log into your Sitellite account on your web site. Next, click on the "Control Panel" option in the Sitellite top bar. From the Control Panel view, select "SiteTemplate" under the "Tools" pane in the top right of the page.

This will bring you to the first screen of the SiteTemplate editor, where you can choose the template set that you would like to modify. When you select a template set, you will see a list of templates, CSS files, and images belonging to that template set.

For the purposes of our examples, let's click the "New Template Set" link and create our own individual template set. You can use your name for the name of your template set. Once you've finished it will take you to your new template set. From here, click on the "HTML > Default" template.

The template editor screen

When you click on a template within a set, you are brought to the template editor screen. This screen contains several features, which include:

- The template name and output mode listed at the top
- The template data itself in a large text area
- A toolbar for inserting frequently used XT tags
- The Save, Preview, and Cancel buttons
- A validation service selector

The default XT template

Looking at any XT template, you can immediately see that it is simply a mixture of XHTML and XT tags. The XT tags are just like the other XHTML tags except that their names all begin with "xt:". This helps keep them distinct for both ourselves as well as the XT template rendering engine.

Inserting page elements

From the template editor screen, we can insert any markup we want into the template, as long as it is XML compliant.

Exercise: Our first template

Create a new XT template named "test" and enter into it the following XHTML markup:

```
<html>
  <head>
    <title>Window Title</title>
  </head>
  <body>
    <h1>Page Title</h1>
    <p>Page Content</p>
  </body>
</html>
```

Click the "Preview" button to see your template.

Next, add the following XT tags:

```
<html>
  <head>
    <title>Window Title</title>
    <xt:content="#head_title">Window Title</title>
  </head>
  <body>
    <xt:content="#title">Page Title</h1>
<span xt:replace="body">
    <p>Page Content</p>
</span>
  </body>
</html>
```

Click the "Preview" button again to see your new changes. You should see your original page title, window title, and page content replaced with sample data from the template previewer.

Here we have introduced the main two XT commands, which act as attributes attached to ordinary XHTML tags. These are:

`xt:content` Replaces the contents of the XHTML tag with the specified XT expression.

`xt:replace` Replaces the entire XHTML tag with the specified XT expression.

XT expressions

XT expressions are the values placed inside the XT tags. These are what allow you to refer to the page elements such as title, description, and body.

The following is a list of all of the main web page properties that can be inserted into your templates:

| | |
|--------------------------|--|
| <code>id</code> | Page ID |
| <code>title</code> | Page title |
| <code>nav_title</code> | Page title as used in the web site navigation |
| <code>head_title</code> | Page title as used in the top bar of the browser window |
| <code>below_page</code> | The page that the current page is a child of in the web site hierarchy |
| <code>is_section</code> | Whether the page is a section index or not. The properties of a section index, such as the template it uses, are inherited by pages the underneath it |
| <code>template</code> | Which template name is being used for this page |
| <code>include</code> | Whether this page is included in the web site navigation or not |
| <code>keywords</code> | The keywords that have been assigned to the current page |
| <code>description</code> | A description of the current page (ie. for the meta tags) |
| <code>body</code> | The body of the page |
| <code>toc</code> | A table of contents linking to the headers within the body of the current page. This is useful for providing additional navigation within longer pages |

Exercise: Using different expressions

Try replacing the title with another field in your template, or adding new tags that reference additional fields.

Inserting boxes

"Box" is a term used in Sitellite to refer to an individual PHP script written using the Sitellite framework. They are called boxes because they are used to output some sort of dynamic functionality that is to be displayed for a web site visitor somewhere on the web site. Examples of boxes include:

- Latest headlines
- Upcoming events
- Any web site navigation

Boxes can be embedded into XT templates by using the "xt:box" tag. The syntax for a box is the name of the app that the box belongs to, ie. "news", followed by a forward-slash (/), followed by the name or path to the box. Here are a few examples:

- `<xt:box name="news/sidebar"> <xt:box>`
- `<xt:box name="sitellite/nav/breadcrumb"> <xt:box>`
- `<xt:box name="sitetracker/bug"> <xt:box>`

Exercise: Calling boxes

Try adding the above box calls to your template and previewing the results.

You can also add sample data between the box tags to immitate the box's output when viewing the template itself directly in a browser (ie. outside of the XT rendering engine). For example:

```
<xt:box name="sitellite/nav/breadcrumb">
  <p>
    <a href="#">Home</a> / <a href="#">Courses</a> / XT Templates
  </p>
</xt:box>
```

Checking for standards compliance

To check a template for standards compliance, simply select the desired standard from the list at the bottom right corner of the template editor screen and click "Go". The validator will appear in a new window, detailing any non-compliant parts of your template and usually making suggestions for correcting them.

Exercise: Template validity

Using our existing template, let's make a deliberate XML syntax error:

```
<html>
  <head>
    <title xt:content="head_title">Window Title</title>
  </head>
  <body>
    <h1 xt:content="title">Page Title</h1>
    <span xt:replace="body">
      <p>Page Content</p>
    </span>
  </body>
</html>
```

Select "Template Syntax (XML)" from the Validate list and click the "Go" button. You should see the following error message:

```
Template Error: not well-formed (invalid token) (Line 6, Column 2)
```

This tells you what the XML validator found wrong as well as the line and column number that the error occurred on. It will also highlight the line of the error below the error message, so that you can quickly inspect

the problem for yourself.

Now close the validation window and click the "Save" button to try to save the template with the error still in it. You should see the following notice appear at the top of the editor screen:

```
Oops! The following information must be corrected in order to continue:
```

```
* Your template contains XML syntax errors.
```

As you can see, SiteTemplate doesn't allow you to save a template if it contains errors that will prevent it from displaying. This is a nice safety feature that helps make sure you never see invalid template errors on your actual web site.

4. Advanced XT capabilities

Advanced XT expressions

In addition to the above expressions that simply called aspects of the web page object, there are several different types of expressions. These expressions can be used and combined with the various XT tags to create some really powerful templating features.

The three types of expressions are as follows:

- path** Path expressions are a simplified way of referring to the contents of objects in the XT object register. Path expressions start with the optional prefix "path: " followed by the name of the object, followed by the name of a property of that object.
- string** String expressions are used to output literal text, called "strings". A string expression starts with the prefix "string: " followed by any string of text.
- php** A PHP expression starts with the prefix "php: " followed by the PHP expression.

Understanding expressions is the key to understanding how all of the advanced XT features work.

Path expressions

A path expression looks exactly like how we named boxes in the "xt:box" tag. Some examples include:

- object/id
- site/domain
- cgi/page
- session/username

In each case, the path starts with the object name and ends with some property of that object, separated by a forward-slash. Each of those examples are real paths, referring to objects in the Sitellite CMS by each of those names.

The "object" object is a special case though. The "object" object is the default object being acted upon by the XT template, which in the case of a web page is the web page object itself. So when we said:

```
<h1 xt:content="title">Page Title</h1>
```

we were actually saying:

```
<h1 xt:content="object/title">Page Title</h1>
```

which, if you remember that paths start with "path: " so that XT knows for sure what type of expression they are, we are actually saying:

```
<h1 xt:content="path: object/title">Page Title</h1>
```

But since "object" is the default, and since the default expression type is a path, we can shorten it and keep it simpler and more readable.

String expressions

String expressions are very simple, and on the surface don't appear very powerful or even all that useful, but they can come in quite handy for the simple fact that you can include in them what we call "sub expressions". I'll illustrate this with an example:

```
<title xt:content="string: Simian Systems - ${head_title}"> </title>
```

Using the "\${sub-expression}" syntax, we can actually embed a path expression into the middle of a string expression, allowing us to combine the two to create content that is part static and part dynamic.

Exercise: Combining path and string expressions

Experiment with different paths and strings to create new combinations. Try making one that welcomes a user by name.

PHP expressions

Sometimes we need to refer to objects in ways other than by simply referencing their properties. In these cases, we need something more powerful than a path expression, so we use a PHP expression.

PHP expressions actually use a shorthand for the full PHP programming language syntax, which helps make them simpler and more convenient for inclusion in XT templates.

An example of a PHP expression is:

```
<h1 xt:content="title"
      xt:condition="php: not empty (object.title)">
    Page Title
</h1>
```

Exercise: Using PHP expressions

Experiment with different PHP functions that you know of, or look more up at <http://www.php.net/>, to come up with new possible uses for PHP expressions. Try outputting a PHP expression displaying the current date and time.

Now that we've looked at the raw expressions themselves, let's put them to use.

Simple loops

Loops in XT are used to output repeated blocks of the template, such as lists and tables. A loop uses the following syntax:

```
<ul>
  <xt:loop through="item php: range (1, 10)">
    <li xt:content="loop/item/value">1</li>
  </xt:loop>
</ul>
```

Or alternately, in some cases you can use the more concise syntax:

```
<ul>
  <li xt:loop="item php: range (1, 10)">
    <span xt:replace="loop/item/value">1</span>
  </li>
</ul>
```

Conditional rendering

Let's start by revisiting one of the examples from above that used the "xt:condition" tag attribute.

```
<h1 xt:content="title"
  xt:condition="php: not empty (object.title)">
  Page Title
</h1>
```

What happens here is that the condition tag is checked and only if it is determined to be true does the tag and anything inside of it display. Otherwise, the tag is erased from the template during the output phase.

You can also do more complex conditional rendering, such as displaying one thing in one case and something else in another. The following example shows a login form for anonymous visitors, but a welcome message for those who are logged in.

```
<xt:condition>
  <xt:if expr="php: session_valid ()">
    <p>Welcome,
      <xt:var name="session/firstname" />,
      <xt:var name="session/lastname" />
    </p>
  </xt:if>
  <xt:else>
    <form action="/index/sitemember-app" method="post">
      User: <input type="text" name="username" /><br />
      Pass: <input type="password" name="password" /><br />
      <input type="submit" value="Enter" />
    </form>
  </xt:else>
</xt:condition>
```

Other XT tags

Multi-lingual text

You can translate any text in a template automatically using Satellite's translation builder and the following tag syntax:

```
<xt:intl>Text to translate</xt:intl>
```

Changing attributes

In addition to changing the contents of a tag, you can also change specific attributes of it using the following two methods:

```
<p
    align="left"
    xt:attributes="align intl/align">
    Some text here
</p>
```

Or even simply:

```
<p align="{intl/align}">Some text here</p>
```

Outputting HTML entities

HTML entities, such as © (©) or é (é) or (non-breaking space), are not valid XML entities, and so you'll find that placing them into your templates can cause an XML validity error.

To get around this, we used the syntax from a project named the "XML Named Character Entity Library" which lives at <http://xmlchar.sourceforge.net/> to solve our problem. This syntax turns any entity into an XML tag. For example:

- © becomes <ch:copy />
- é becomes <ch:eacute />
- becomes <ch:nbsp />

All of the HTML entities defined in the HTML specification are supported.

Exercise: Using special characters

Try adding these special character tags to your template to see how they work. Experiment with different entities that you know, or look new ones up here:

<http://www.w3.org/TR/REC-html40/sgml/entities.html>

Why use XT programming?

Using these XT tags can save time and decrease complexity by not having to call out to PHP code for every type of dynamic template composition request.

When to use XT vs. PHP code

When something is as simple as a single condition or a loop, XT is often a good solution. XT is also a good solution when the code directly relates to the template itself. For anything more complex however, a box containing PHP code should be used.

5. How to get help

Sitellite.org

- The official XT reference:
<http://www.sitellite.org/index/news-app/story.48>
- XT expressions:
<http://www.sitellite.org/index/news-app/story.47>
- XT tips and caveats:
<http://www.sitellite.org/index/news-app/story.46>
- The complete template documentation:
<http://www.sitellite.org/index/news-app/section.Templates>
- Community forum:
<http://www.sitellite.org/index/siteforum-app>

Simian.ca

- Simian contact information:
<http://www.simian.ca/index/contact>
- Technical support page:
<http://www.simian.ca/index/support>

Copyright © 2004 Simian Systems Inc.
All rights reserved.