onCourse web design handbook

Aristedes Maniatis

onCourse web design handbook

by Aristedes Maniatis

version unspecified

Publication date 4 Oct 2017

Licence

Disclaimer

No liability for the contents of this document can be accepted. Use the concepts, examples and information at your own risk. There may be errors and inaccuracies that could be damaging to your system. Proceed with caution, and although it is highly unlikely that accidents will happen because of following advice or procedures described in this document, the author(s) do not take any responsibility for any damage claimed to be caused by doing so.

Contents

An overview

Contents

Customising onCourse Web

onCourse Web is designed to be extremely flexible, allowing you to implement almost any concept you create for your site. When you engage your graphic designer to work on your brand and website, you will want to make your onCourse website reflect your organisation at every level. That means not only skinning the site to reflect your colour and logo, but thinking about navigation, search and the overall user experience (UX). onCourse has been built with this in mind.

At the same time, we want to save you time and money by giving you many of the basic building blocks you need to create your site.

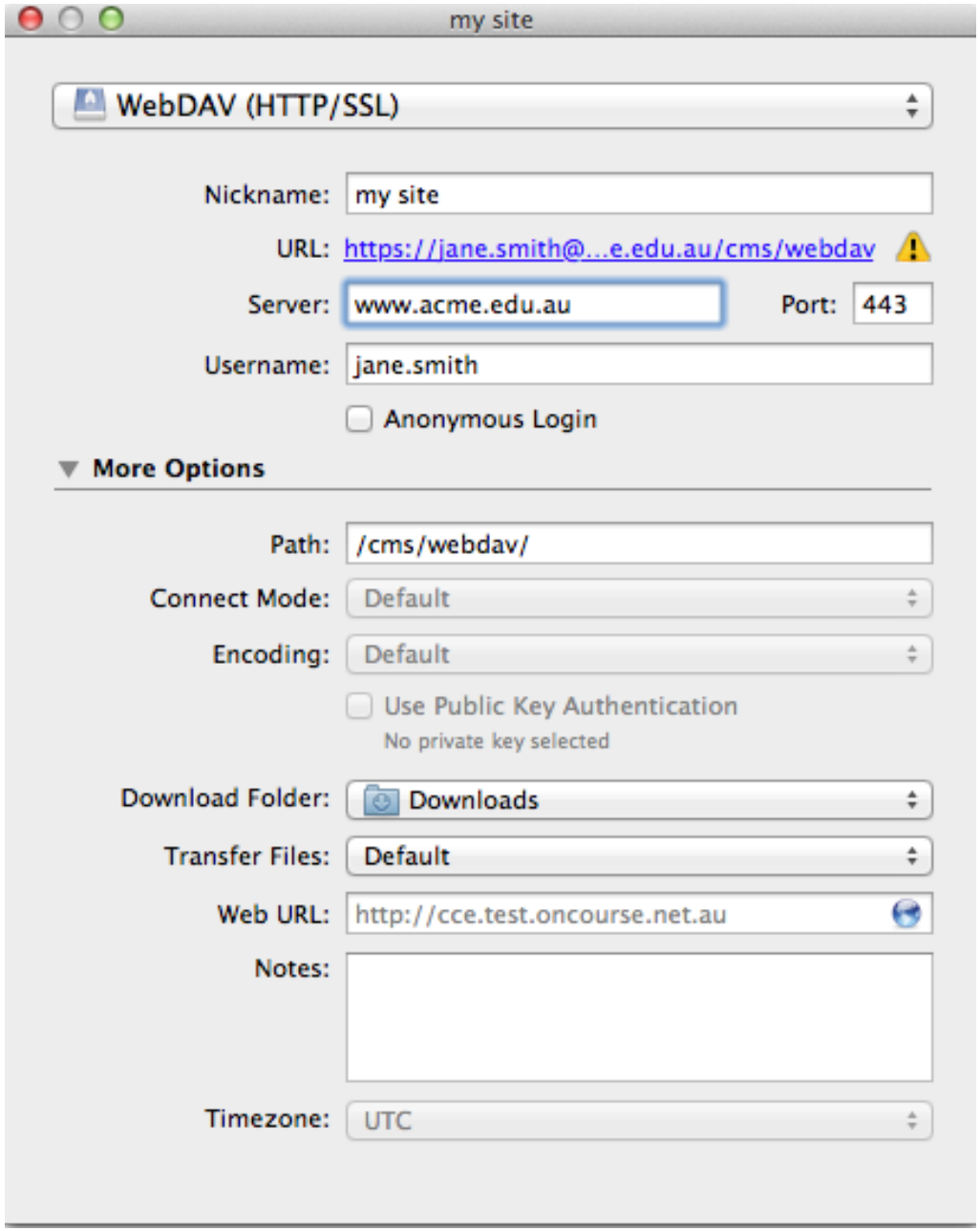
If you are after quick simple results, skip straight to the stylesheet css chapter and focus on changing just the stylesheets to get the look you are after. A lot can be done with little effort in this way. If you want to make changes to the structure of the site itself, read onto the templates chapter for an understanding on how to change the html which makes up the site.

If you are not a web designer and reading this manual, you may find many of the concepts quite technical. Don't worry: you can still edit content using the online CMS without understanding anything here. Your designer, or ish, will be able to work on the page design and create something into which you can drop your content.

Editing your site

While you can log into your CMS to edit content, if you want to change the design of your website you'll need to use WebDAV. This is very similar to ftp except it makes it easier for us to encrypt your connection and apply special automatic rules and filters.

Log into WebDAV using any standards compliant application. We recommend Cyberduck which is free for use on Windows and Mac. Configure a connection in Cyberduck with the URL https://www.acme.edu.au/cms/webdav/

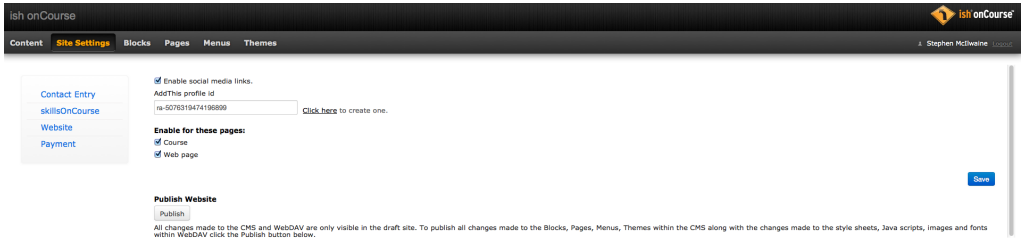


Cyberduck Configuration

Draft site

All onCourse websites have both draft site and a live website. The draft site will be visible to you only when you are logged into the CMS. If your site is http://www.acme.edu.au, then you can log into the CMS at https://www.acme.edu.au/cms/. Any changes you make in the CMS or through WebDAV are only visible to you and any other users logged into the CMS. You can make content changes and experiment in any way you choose without disturbing your live website.

Once you have finished with your changes to the draft site and are ready to publish them to the live site you can do this within the CMS. The "publish" button can be found in the Website section of the Site Settings tab within the CMS. Any changes made within either WebDAV or the CMS will be deployed to the live site when you press the publish button. You'll need to wait a few minutes before the content is visible on your production site due to caching in our cluster of servers. Some users may need to wait even longer if they have a local proxy server, which keeps a copy of the old site until it expires from its cache.



How to publish your site

Special URLs

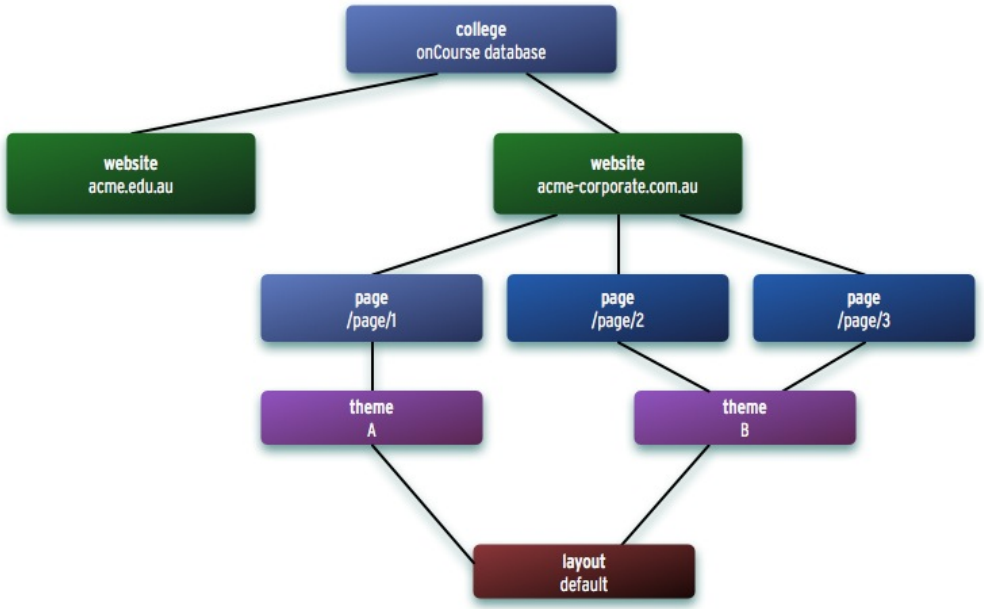
Some URLs in your site are reserved for special pages. These pages are delivered by the onCourse software itself. To customise them, consult the templates chapter for details of how these pages are created. You cannot edit the content of the pages

within the CMS because you will use the onCourse software to edit the courses, classes and other information which drives them. These pages include:

/courses	A list of all courses which are marked as web visible.
/courses/arts	A list of all courses tagged with the tag "arts" from the "subject" tag group.
/course/ABC	The detail of the course with code ABC.
/class/ABC-123	The detail of the class with code ABC-123
/sites	A list of all sites marked as web visible.
/site/12	The detail of site with internal id 12.
/tutor/23	The detail of the tutor with internal id 23.
/page/1	Every page you make in the CMS will be given an automatic page number reference. Even if you don't give it a nicer path, it will always be accessible by this URL.
/enrol	All pages under this path are reserved for the enrolment, application and sales processing pages.
/cms	All pages under this path are reserved for the cms engine.
/cms/webdav	All pages under this path are reserved for webdav access.

Building CMS pages

Pages which are not 'special' are delivered from the CMS. That is, you are able to create pages and assign them to any URL you choose. Each page belongs to a website. You can have more than one website for the same onCourse database. Each page can also be given a theme which defines which blocks appear on the page. Themes are then linked to layouts which defines the html and stylesheets used. The following graph shows the relationship between the onCourse database, websites, pages, themes and layouts.



Object hierarchy

The database

Starting from the top of the diagram we have the onCourse database itself. This is the application which contains courses, classes, tags, students, tutors, and so on. The data in the onCourse database is entered completely within the onCourse client/server applications and is automatically synchronised with the website.

The website

Each onCourse database can drive one or many websites. Each website will have one or more domains which are used to access them. So Acme Training might have the general leisure learning site at http://www.acme.edu.au and the corporate training at http://corporate.acme.edu.au and also http://www.acme-corporate.com. The leisure

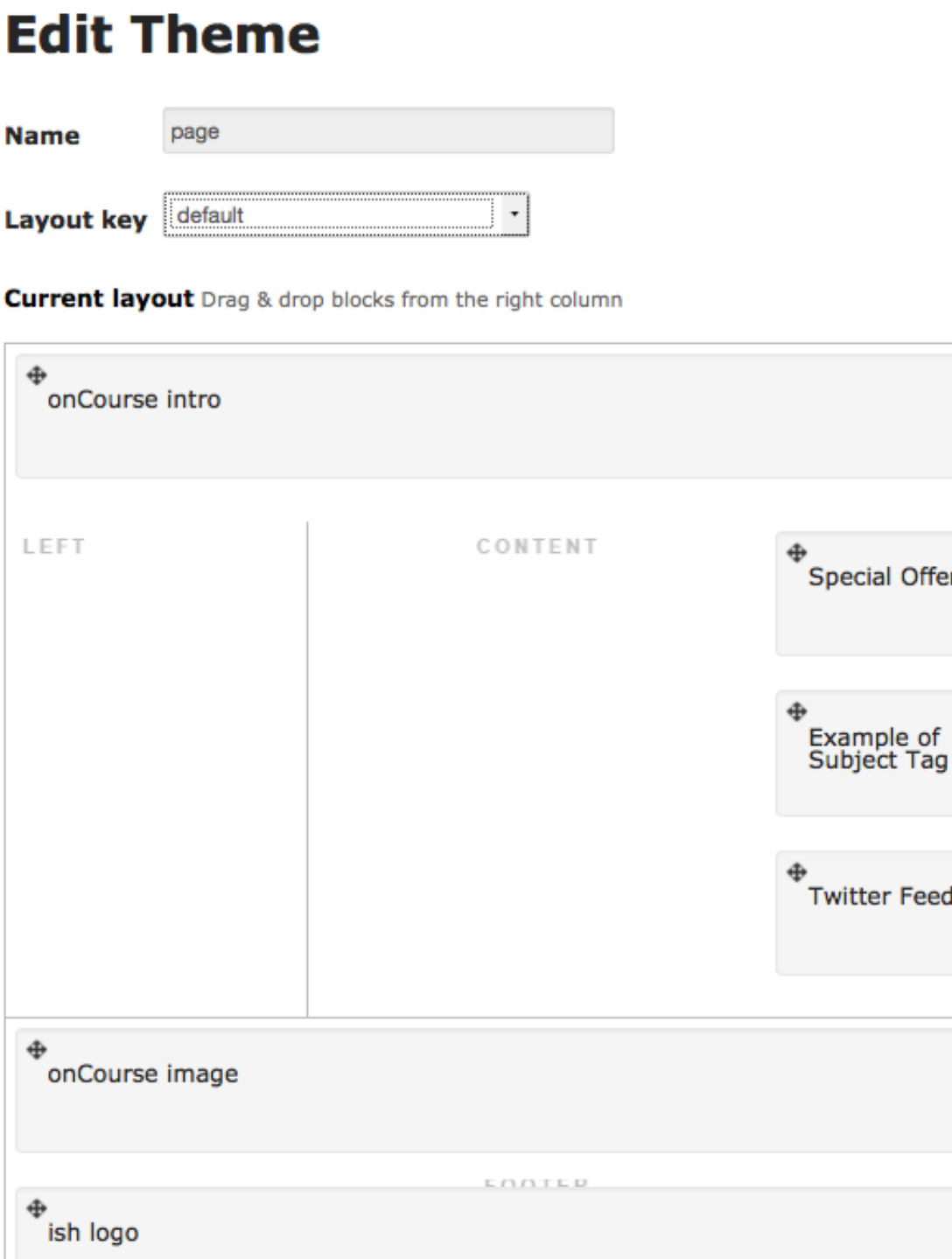
and corporate sites can have completely different content (pages), different graphic design (layouts) and even display different sets of courses.

The pages

The website is made up of pages entered through the CMS. For full details on how to write and build web pages, consult the onCourse website and CMS handbook. Each page has content (text, pictures, etc) and will be linked to a theme.

The theme

Themes are a way of grouping pages and giving them their own character. You can place blocks on a theme, so you might create a theme for policies, another theme for news, and one for general pages. The policy theme could then include a block on the right side with information for students about lodging complaints and contacting the principal. The news theme might contain a block down the left with a random 'hot' course and a block across the top with a rotating banner ad. Finally, the general theme contains a block on one side with navigation elements and assorted other blocks of special offers.



Theme Editor

The layout

The final piece of this structure is the layout. This is where you as a designer will weave your magic. The layout is represented by a folder in the layouts folder you access in WebDAV. You can place templates inside that folder where each template overrides a certain piece of html on the page. In this way you have full control over the entire layout and design of the site, right down to every line of html sent to the browser.

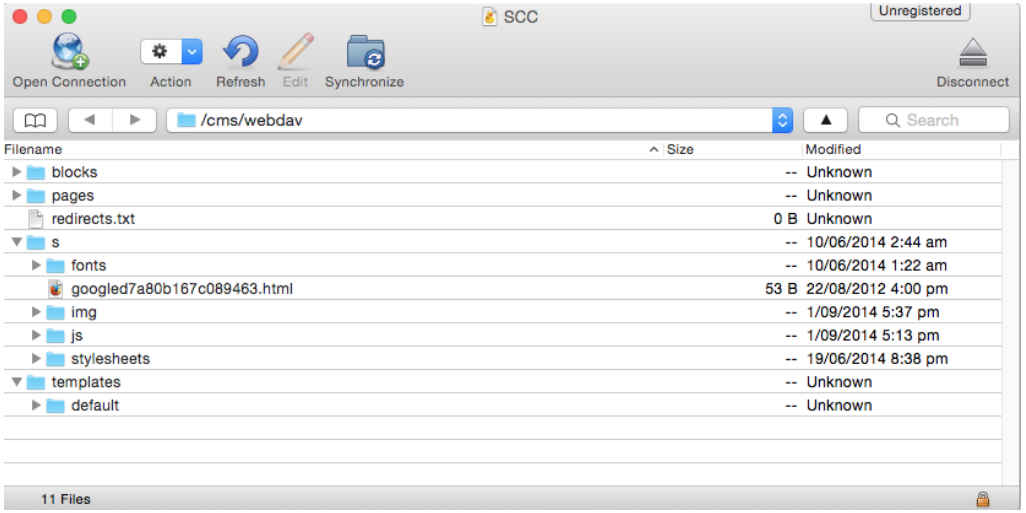
There is always a layout called 'default' which is used by the system for the special pages outlined above. You can create as many layouts you like and link them to themes you create in the CMS.

WebDAV files

When you login into WebDAV you will see folders like this:

- pages
- blocks
- templates
  - default
- s
  - stylesheets
  - fonts
  - img
  - js

In addition you will see a redirects.txt file that will contain a full list of the redirects on your website.



WebDAV folders

The **s** folder contains static files which are not parsed by the application server. These include css, fonts, images and javascript. Keeping things organised within the folders as provided will make everything easier, but you are free to create whatever additional folders you need to here. Template overrides live inside the **templates** folder. One layout is provided for you called **default**which you cannot delete.

You can edit pages and blocks directly from within WebDAV. The result is the same as if you had edited those same items from within the CMS.

Video

Although it is possible to load video files directly into the static folder, the onCourse servers are not optimised for serving video. You will get much better results by hosting your video at a site such as YouTube or Vimeo and linking to them from within your pages. Not only do they have servers placed in data centres around the world, but also the allow streaming of video. That is, the video can start playing even before all of the file has downloaded.

Alternatively you can upload video to the onCourse document management system and deliver it from there, however you'll have to create your own video player and tie

them all together so we still recommend one of the third party video delivery systems like YouTube or Vimeo.

Favicon

Some websites show a tiny icon in the URL bar and in bookmarks. This is called a favicon. To add a favicon to your onCourse website, you need an image in both '.ico' and '.png' format. The standard is to have the .ico images in either 16x16 or 32x32 size, and the .png up to 180x180.

Both of these images have to be uploaded via WebDAV to the directory '/s/images' with the names 'favicon.ico' and 'apple-touch-icon-precomposed.png' (you can use any names for these images, but these are the convention).

Next, add the relevant html to the PageHead.tml file, found in the '/templates/' directory in WebDAV.

Adding the following lines (assuming you have named the .ico and .png files conventionally) should have web browsers auto detect and display the favicon images

```
<link href = "/s/images/favicon.ico" rel = "icon" type = "image/vnd.microsoft.icon"/>
<link href = "/s/images/favicon.ico" rel = "shortcut icon" type = "images/x-icon"/>
<link rel = "apple-touch-icon-precomposed" href = "/s/images/apple-touch-icon-
precomposed.png"/>
```

Stylesheets

Contents

One of the first ways you'll want to modify the look of your site is by changing css stylesheets. onCourse comes with a set of default stylesheets to make your life easier, so you will probably start with a copy of our template-a, template-b or template-c. These in turn build on our base stylesheets which we update once a year or so.

Because you are building on existing stylesheets, lots of things are already taken care of for you. Your html/css developer can save weeks of work with our existing responsive layouts, grid and basic styling.

File structure

First let's see where all the files are. To make it easier to navigate, all the CSS is broken up into lots of files within a set of folders.

Start inside the '**/s/stylesheets**' folder in WebDAV. You'll see two folders: 'css' and 'src'. It is important that you do not edit the files in 'css' directly, but instead only edit the files in 'src'.

```
/s/stylesheets/src/
```

Start by looking at the file site.scss. That's the top of the stylesheet structure and it includes all the other files you need. Look for lines like this:

```
@import "application/settings";
```

We recommend you create a new file with your customisations and add a reference to that from site.scss. Don't remove the import of the 'base' stylesheets. That import brings in the core stylesheet definitions which are required for your onCourse site: shortlists, course and class listings, enrolment templates and much more. Override them however you want, but they will save you a lot of work rather than starting from scratch.

SCSS

If you didn't recognise the `@import` command above as CSS, that's because onCourse uses a variation of CSS called SCSS (also known as SASS). This extends the basic stylesheet concepts and adds some very useful abilities:

nested rules
variables
mixins
selector inheritance

Every time you edit any file in `/s/stylesheets/src/`, the destination `/s/stylesheets/css/site.css` is automatically regenerated by libsass. This takes less than a second, so you can see the results almost immediately. Remember that to see changes in the staging site you must be logged into the CMS in your web browser. You should not edit the site.css file directly, as any changes you make will be overwritten.

If you don't want to bother with learning SCSS that's fine. Just write ordinary CSS in the site.scss file. As your stylesheets become more complex, you will find that SCSS gives you valuable shortcuts to achieving what you want and you will never want to go back.

If you would like to break up your stylesheets into more manageable pieces, add another import statement under the 'base' import like this:

```
@import "colours";
```

Then create a file `/s/stylesheets/src/_colours.scss` (with the underscore). When you make changes to that file, onCourse will automatically merge any content from _colours.scss into the main css for your site.

Look through the default styles for variables which you can easily modify to change your site. For example, override `$bodyFontFamily` in order to change the font right through your site. Or change `$primaryColor`, `$secondaryColor` and `$containerWidth`.

Bourbon

By default your stylesheets include Bourbon. This css library gives you lots of useful functionality that you'd otherwise have to write by hand. Read up on this library and get instant rounded corners across all browsers, typography features, reset, and much much more. As just one simple example

```
section {
  @includelinear-gradient(totop,red,orange);
}
```

will give you the following output

```
section {
  background-color:red;
  background-image:-webkit-linear-gradient(bottom,red,orange);
  background-image:linear-gradient(totop,red,orange);
}
```

without having to remember to put Opera, webkit, Mozilla and html5 elements into your css. You don't need to use bourbon, but it can help you keep your site consistent across browsers more easily and save you time with common css blocks.

Read the docs for Bourbon.

Compression

No matter how many separate files you break up your stylestyles into, the output will be compressed into one file and minified. This means whitespace is stripped and the file is pretty hard to read. However browsers will be able to parse it just fine; this minification can make a big improvement to page load speeds and also to your SEO.

Comments will be stripped out, so don't hesitate to put lots of useful notes in your scss files.

Finally we compress the file with gzip to serve it across the internet as fast as possible. You'll see these output files as:

- `/s/stylesheets/css/site.css`
- `/s/stylesheets/css/site.css.gz`

Map

Because it can be hard to review minified and combined CSS in your browser, we also output a map file. This allows Chrome and Firefox developer tools to identify the real file and line number in the source scss where your stylesheet rule can be found, saving you a lot of searching. The map can be seen as:

- `/s/stylesheets/css/site.css.map`

Your browser will automatically find and use that file if it knows how.

Responsive design

It is extremely important in a modern world filled with tablets and smart phones that your site is built to make life easy for those users. onCourse sites already are prepared with responsive designs at four sizes. That means that as the browser window gets smaller with different sized devices, the design itself alters to work better at that size. It is still up to you as a designer to properly take advantage of this responsive design, but the groundwork is already there for you in onCourse and the enrolment pages as well as skillsOnCourse are optimised already.

onCourse has media sizes of $small-screen, $tablet-screen and $desktop-screen throughout the base stylesheets and a grid based on bootstrap 3.

Bootstrap

Because we bring in bootstrap 3 by default, you get not only a nice grid but also a lot of common component styles which are very useful.

Javascript

Contents

File structure

Log into webDAV and look at this folder:

```
/s/js/
```

In there you'll see a number of javascript files which drive your onCourse website. Some are third party vendor supplied files and others are default parts of onCourse itself. site.js is the starting point and it has a list of all the other files which are included. Note that the order of including files can sometimes be important. Let's look at base.js now:

```
//= minify off
//= require base.js
//= require extra.js
```

This is telling you that minification is disabled and that the javascript files base.js and extra.js are included. This is a different syntax to SCSS so don't get them confused. The combined output file is called all.js and the files which are merged don't need to start with an underscore.

Minification

Unlike CSS, the javascript is not automatically minified. This is because sometimes javascript minification can actually break your javascript (particularly if it contains errors). Browsers will often manage to figure out your broken javascript, but once minified, errors really cause problems.

You can enable minification by changing the "off" to "on" and saving the file. Wait about 30 seconds and then review your site in staging before pushing it to live.

We use the Google Closure compiler to verify and minify the output.

Compression

all.js.gz is automatically created and used by browsers. This is another reason minification may not be quite so important: compression often does a pretty good job of reducing the file sizes.

Core libraries

We supply a number of third party javascript libraries with your site by default. The following list are bundled together as dynamic.js and cannot be removed without breaking some key shopping basket functionality on the site:

- classnames
- react
- react-dom
- react-redux
- redux
- redux-thunk
- jquery 3.x

In addition there are some useful libraries which are often used when building sites:

- BxSlider 4.x
- jquery.customSelect
- jQuery Validation Plugin

You may wish to remove or add more libraries for your particular design

SEO and analysis

Contents

Overview

onCourse is pre-configured with a whole range of SEO features to make your life easier and get good search ranking with minimal effort. Although we take care of many of the technical details, there is absolutely no substitute for writing good copy. That is, if you don't spend the time to write words which will get you ranking, then you are doomed from the outset. But if you write course descriptions and tag descriptions which are both engaging to potential students and use the correct search terms that people will search for, you are well on your way to a successful website.

Tag Manager

Every site we build is created with Google Tag Manager integrated. This service drives the injection of javascript and tracking pixels or html into your site in really neat ways. In particular it allows:

- Adding livechat systems to your site
- Facebook, Google and other tracking code
- A/B testing (show some elements to some people and not others)
- Different content for different country regions
- Inject advertising panels
- Javascript is asynchronously loaded (so it doesn't slow the page draw)
- Versioned changes (so you can roll back easily)
- Event tracking (such as add to shopping basket)

When we build your site, we'll give you access to Google Tag Manager, set up some initial Analytics tags for you, and get you started. You can then build more tracking or hand over access to your SEO consultants.

Google analytics

onCourse injects some additional data into your analytics beyond the standard page views:

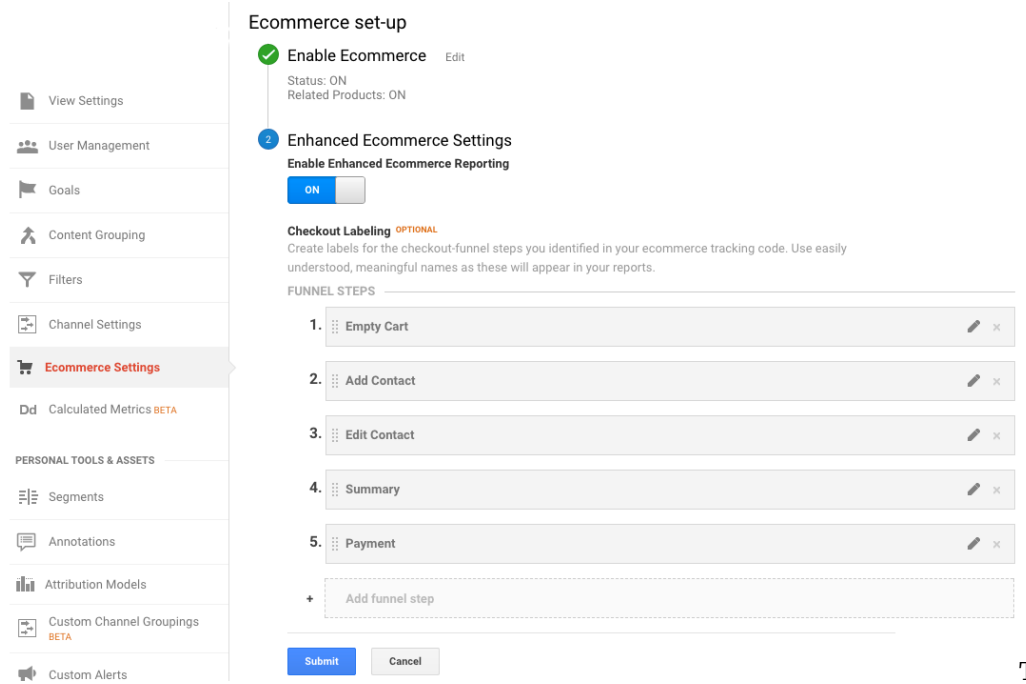
- Transactional sales dollars. Every sale through the website puts the dollar value into your Analytics. So not only can you see how many page views you got from that ad campaign, you can see the dollar value of the sales that generated. You'll discover that the two aren't always correlated. Do you care about ten thousand views from Facebook likes or the 200 real sales from a targeted campaign? This is how you can tell the difference.
- Course and products. Drill down in your data to see where sales are coming from. Getting lots of language course sales from an ad you placed on a friendly site? Great, now you know what works and where to focus your time and money.
- Events. Because we log "add to shopping basket" and other events, you'll be able to see who is engaging with your site but not following through to the end.

- Checkout steps. You can track each step in the checkout process through a Google Analytics acquisition funnel. See where and why people are dropping off.

Google Analytics Setup

In order to get data into your Google Analytics account you'll need to complete the following steps:

1. Create a new Google Analytics account (or use an existing one). Under `Admin > Property > Tracking code` look for the tracking id and make a note of it.
2. Log into Google TagManager and find `ish onCourse Google Analytics settings` under variables. Put your tracking id in there to replace the dummy one.
3. Back in Google Analytics, you should soon be able to the Tracking Code status update to show data coming in. Visit your website to generate a bit of traffic.
4. Enable enhanced ecommerce reporting in the Google Analytics under `Admin > View > E-commerce settings`
5. Select names for your checkout steps
   a. Empty Cart
   b. Add Contact
   c. Edit Contact
   d. Summary
   e. Payment
6. Submit changes



There is no real time e-commerce reporting yet, so you'll need to wait a few hours for events to process and then view the shopping behaviour and checkout behaviour reports.

When you have some time, explore the Analytics "goals" setup for ways to get useful information such as dropoff in your checkout rates.

Sitemap

We generate a /sitemap.xml file automatically which you can add to Google's webmaster tools. This allows Google to quickly find all the pages on your site without crawling through them one at a time, and it also gives Google hints about when those pages are updates so that they are crawled more quickly.

HTML markup

The html markup of your site is customisable throughout, but by default you get some robust templates that Google will love.

Semantic HTML

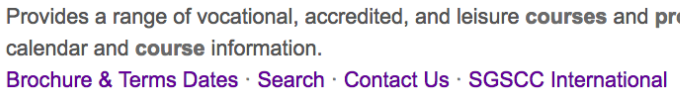
By building the html of your onCourse site with a clear structure and meaning, search engines such as Google are able to make sense of the structure and meaning. onCourse comes with good html that gives you a good starting point. For example, each page should only have a single h1 element, sections, footer and other modern html elements.

Accessibility

Although this is not strictly an SEO benefit, adhering to accessibility standards means that students with disabilities such as poor sight will be able to access your website more easily. This goes hand in hand with good semantic HTML and also means that Google and other search engines will properly index your site and understand the content structure rather than just index a mass of words.

Schema.org

Schema.org is a set of markup rules endorsed by Google and other search engines. It specifies ways to annotate the html to give it specific meaning allowing Google to create rich search results. For example, we can markup data so that Google can provide navigation structure, course dates, prices and other information right in their search results.



Navigation

The navigation and structure of the site's page are important to good search results.

Canonical links

Some pages in your onCourse site don't last very long. In particular, the class pages with a URL like `/class/ABC-123` will be irrelevant as soon as that class is finished. You don't want to accumulate page rank on these pages, only for that to evaporate. onCourse automatically adds a canonical link to the correct course page, transferring any page rank and inbound links to somewhere it will do good.

URLs

onCourse has clean URLs which are easy for users and contribute to your SEO. All course detail and list pages contain the word "course" and you are free to create long course codes which contain useful SEO keywords. And your tag structure can be structured however you want.. For example, a URL like "/courses/business/communication" picks up several important keywords.

Redirects

If you ever change URLs it is vital that you don't lose page rank on those old pages. While you don't need to worry about classes, tags and course pages can accumulate valuable scores and you should implement redirects to the closest new page.

Meta headers

onCourse does not implement meta-keyword headers since it is well documented that no search engines use them for indexing. However we do implement:

- og:image for course and class detail pages. The image is pulled from the first attachment of the appropriate type (jpg/png) linked to the course.

- og:type is hardcoded to "website" to satisfy Facebook.
- og:description and meta-description. The contents of this field is automatically populated from the beginning of the text in the course description or page content. Special formatting is stripped out (eg. headers and images). For this reason, it is useful for you to ensure the first paragraph of text is relevant and well written.
- Page title (and og:title) is constructed using the name of your college and other details such as the name of the course or tag, or the name of the page.

These tags are useful for both Facebook, Bing and Google, helping you display better search results from data discovered by each of their crawl engines.

Should you wish to customise the behaviour of how onCourse inserts meta headers you can do this in PageHead.tml. You can customise the page title in Title.tml

SSL/TLS encryption

Google has indicated that sites with end-to-end encryption will get higher search rankings since they are more likely to be ligitimate sites which care about the privacy of their users. All onCourse sites redirect immediately to HTTPS for even the non-payment gateway parts of the site.

Page speed

A lot of effort goes into making onCourse really fast, even when displaying faceted search results from thousands of courses. We use separate SSL/TLS load balancers to remove encryption load from the application servers, we cache database requests and page rendering, and use Apache Solr as a high speed search cache. We also offload large image serving to a third party AWS S3 storage, again increasing the speed with which your site displays.

The end result of all this is that search engines give your site bonus points for being fast. Which is good.

HTML Templates

Contents

Editing templates

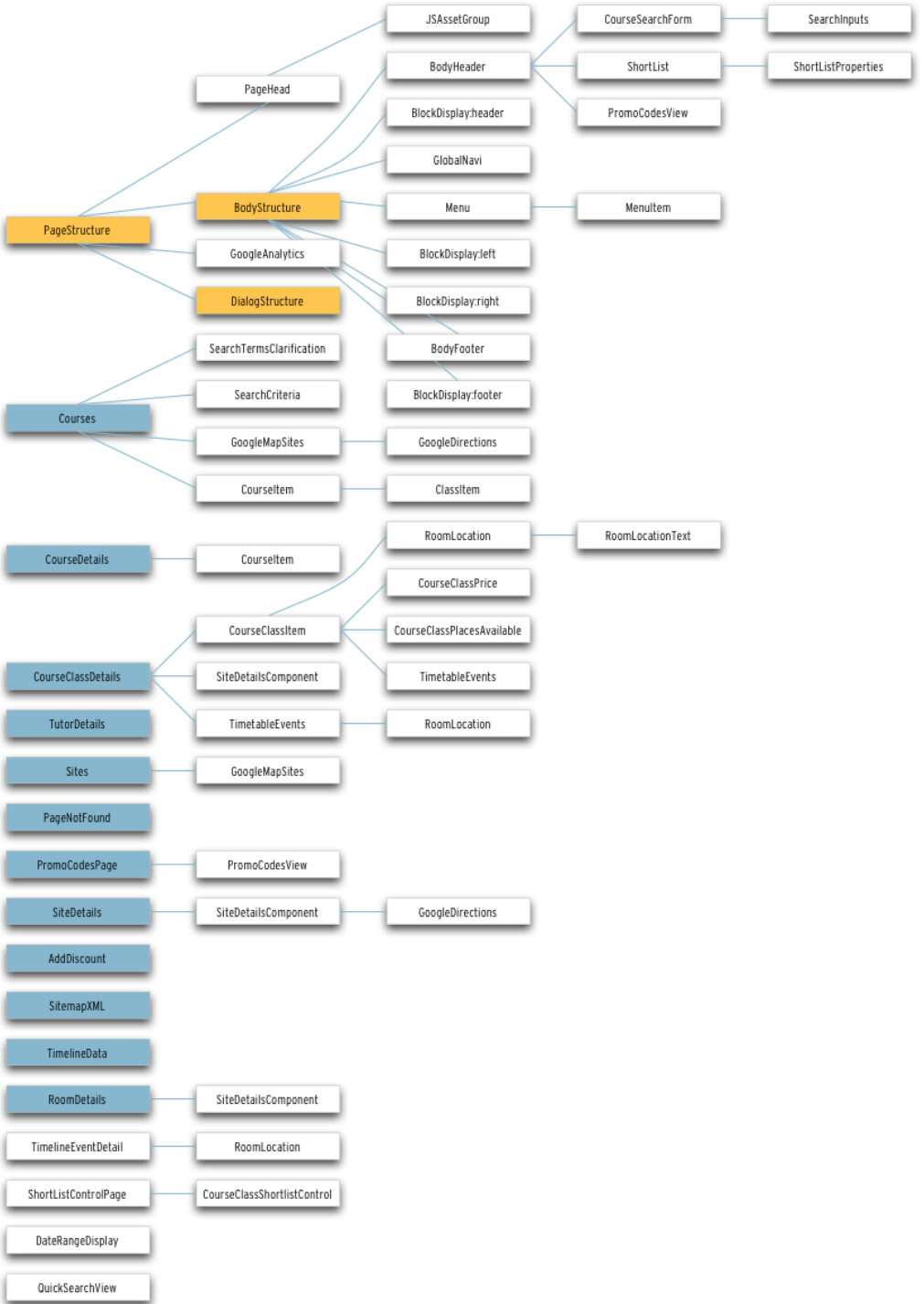
Once you have exhausted the capabilities of styling your site purely through modifying css, you may want to customise the html which is generated by the web application. To do this, you will edit the templates which are used to build the site. You can override any of the templates below, but don't go overboard: if you can achieve the results through modifying the css, do that first since it will be easier to upgrade your site to take advantage of new features as they become available in onCourse.

If you want to override the templates below, simply place your modified files into an appropriate layout. For example to customise the body structure on the default template, create a file here:

```
/resources/layouts/default/BodyStructure.tml
```

In onCourse templates are **well formed XML documents**. That means that every open tag must have a matching close tag, every attribute must be quoted, and so forth. You

can override any of the templates in this handbook by simply creating a file with the appropriate name in the resources/layouts/default folder. Naturally if you are using more than one layout, you can have multiple sets of templates, each in their own folder within resources/layouts.



The hierarchy of the templates is quite complex

Page wrappers

All pages on the site are wrapped in one of two main templates. Most pages will have PageStructure.tml wrapped around them to render menus, headers, blocks, etc. Other web requests might be drawn inside a 'lightbox' effect and so should not have all this extra content.

Page Structure

Page structure is one of the few templates that is not editable. You cannot override this particular template since it controls how the CMS integrates to the site. But you have enough other hooks that you should not need to worry about this template.

```
<!DOCTYPE html>
     <html xmlns = "http://www.w3.org/1999/xhtml" xmlns:t = "http://tapestry.apache.org/
schema/tapestry_5_1_0.xsd"
     xmlns:p = "tapestry:parameter" lang = "en">
     <head>
     <head t:type = "ui/pagehead" title = "${title}"/>
     </head>

     <body t:type = "any" class = "prop:agentAwareBodyClass" id = "prop:bodyId">
     <div  t:type = "ui/BodyStructure" webNodeType = "webNodeType">
     <t:body/>
     </div>
     <span t:type = "ui/googleAnalytics"></span>
     </body>
     </html>
```

Body Structure

Filename BodyStructure.tml

The important thing about the body structure is that it is called by other page templates. So when Courses.tml wants to render itself, PageStructure is used to provide the main structure of the page. When you reach `<t:body/>` in the template below, Courses.tml is inserted.

```
<section class = "site-wrapper" xmlns:t = "http://tapestry.apache.org/schema/
tapestry_5_1_0.xsd">
 <section class = "site-container" id = "container">
  <header class = "site-header" id = "header">
   <div t:type = "ui/bodyheader" />
   <div t:type = "ui/blockdisplay" webNodeType = "webNodeType" regionKey =
 "literal:header" />
   <div t:type = "ui/globalnavi" />
   <nav class = "site-nav" id = "nav">
    <div t:type = "ui/menu"/>
   </nav>
  </header>
  <section class = "content-container" id = "contentContainer">
   <aside class = "sidebar-left" id = "sidebarLeft">
    <div t:type = "ui/blockdisplay" webNodeType = "webNodeType" regionKey = "literal:left"
 />
   </aside>
   <article class = "content" id = "content">
    <div t:type = "ui/blockdisplay" webNodeType = "webNodeType" regionKey =
 "literal:content" />
    <t:body />
   </article>
   <aside class = "sidebar-right" id = "sidebarRight">
    <div t:type = "ui/blockdisplay" webNodeType = "webNodeType" regionKey =
 "literal:right" />
   </aside>
  </section>
  <footer class = "site-footer" id = "footer">
   <div t:type = "ui/bodyfooter" />
   <div t:type = "ui/blockdisplay" webNodeType = "webNodeType" regionKey =
 "literal:footer" />
  </footer>
 </section>
 <div id = "overlay" style = "display: none; height: 500px" />
 <div id = "timeline-wrap" style = "visibility: hidden;">
  <div id = "timeline" />
 </div>
</section>
```

Dialog Structure

Filename DialogStructure.tml

This is a more minimalist page layout which is used for things that display in a 'lightbox' style dialog within the page. It contains no javascript references, but does include the main site.css stylesheet.

```
<!DOCTYPE html>
<html xmlns = "http://www.w3.org/1999/xhtml" xmlns:t = "http://tapestry.apache.org/schema/
tapestry_5_1_0.xsd"
   lang = "en">
 <t:remove> Name: DialogStructure Purpose: displays the dialog page for nyromodal
 popups. </t:remove>
 <head>
  <title>${title}</title>

  <script type = "text/javascript" src = "/s/js/all.js"></script>
  <link href = "/s/stylesheets/css/site.css" media = "screen" rel = "stylesheet" />
 </head>
 <body class = "dialogBox">
  <div id = "popup-header"></div>
  <div id = "popup-content">
   <h2>${title}</h2>
   <t:body></t:body>
  </div>
 </body>
</html>
```

Page templates

Some of the templates you can edit are reached by a particular URL. For example, the /course/ABC renders CourseDetails.tml. All of these top level page templates begin with something like this:

```
<html t:type="ui/pagestructure" title="Page title" bodyId="literal:DetailsPage">
```

This will render the template, wrapped with the PageStructure.tml template (to draw the main page design and layout), and give you the opportunity to override the page title and body css id.

Courses

Filename Courses.tml

URL: /courses/[tag]/[tag]/*

URL: /courses?s=[search terms]

Renders the course list after a user performs a search or browses the subject tags. The parameters in the URL define the position in the subject tag tree the user is looking at, or the search parameters they used. The first form of URL showing the list for a particular subject tag is deliberately designed to be Google friendly and encourage bookmarking.

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd"
             xmlns:p = "tapestry:parameter">

    <t:delegate to = "currentBlock"/>

    <t:block t:id = "filteredCourses">
        <span id = "courses-list">
            <t:if t:test = "hasInvalidSearchTerms">
                <span t:type = "ui/SearchTermsClarification" paramsInError =
"paramsInError"></span>
                <p:else>
                    <span t:type = "ui/SearchCriteria" hasMapItemList =
"hasMapItemList"></span>

                    <div id = "sitesMap">
                                <span t:type = "ui/GoogleMapSites" sites =
"mapSites" focuses = "focusesForMapSites"
                                        collapsed = "true" showLocationMap = "false"></span>
                    </div>

                    <t:if test = "browseTag">
                        <t:if test = "browseTag.hasDetails">
                            <t:outputraw value = "${browseTagDetail}"/>
                        </t:if>
                    </t:if>

                    <t:if t:test = "isException">
                        <h2>Because of an error in the search engine, all courses are
                            being displayed. Our technical team is working to resolve this
                            problem now.</h2>
```

```
                </t:if>
                <t:if test = "DebugRequest">
                    <t:outputraw value = "${debugInfo}"/>
                </t:if>
                <t:if t:test = "hasAnyItems" negate = "true">
                    <h2>No results</h2>
                    <p:else>
                        <t:delegate to = "block:moreCourses"/>
                    </p:else>
                </t:if>
            </p:else>
        </t:if>
    </span>
</t:block>

<t:block t:id = "moreCourses">
 <span t:type = "ui/CoursesList" courses = "courses" coursesCount = "coursesCount"
            loadedCoursesIds = "previouslyLoadedCourseIds"
            itemIndex = "itemIndex" sitesParameter = "sitesParameter" searchParams =
"searchParams" debugInfoMap = "debugInfoMap"/>
</t:block>

<t:block t:id = "htmlCourses">
    <html t:type = "ui/internal/pagestructure" bodyId = "literal:ListPage"
          bodyClass = "literal:internal-page"
          canonicalLinkPath = "${canonicalLinkPath}"
          canonicalRelativeLinkPath = "${canonicalRelativeLinkPath}"
          metaDescription = "${metaDescription}">
    <t:delegate to = "block:filteredCourses"/>
    </html>
</t:block>

</t:container>
```

Course Details

Filename CourseDetails.tml

URL: /course/[course code]

Renders a single course description page.

```
<html t:type = "ui/internal/pagestructure"
   canonicalLinkPath = "${canonicalLinkPath}"
   canonicalRelativeLinkPath = "${canonicalRelativeLinkPath}"
   metaDescription = "${metaDescription}"
   title = "${courseDetailsTitle}"
   bodyId = "literal:DetailsPage"
   bodyClass = "literal:internal-page"
   xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
"tapestry:parameter">
<t:if t:test = "course">
 <div t:type = "ui/confirmorderdialog"/>
 <span t:type = "ui/courseitem" course = "course" linkToLocationsMap = "true" isList =
"false" courseItemModel = "courseItemModel"></span>
<p:else>The requested course is not found.</p:else>
</t:if>
<span t:type = "ui/Attachments" entityIdentifier = "literal:Course" entityIdNum =
"${course.id}"/>
</html>
```

Class Details

Filename CourseClassDetails.tml

URL: /class/[class code]

Renders a single class description page. For technical reasons a class is called "CourseClass" within the internals of onCourse.

```
<html t:type = "ui/internal/pagestructure"
   canonicalLinkPath = "${canonicalLinkPath}"
   canonicalRelativeLinkPath = "${canonicalRelativeLinkPath}"
   metaDescription = "${metaDescription}"
      title = "${courseClass.uniqueIdentifier} | ${courseClass.course.name}"
   bodyId = "literal:DetailsPage"
      bodyClass = "literal:internal-page"
      xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
<div class = "class-details-title">
 <h1>${courseClass.course.name}</h1>

 <div class = "print-page" onclick = "window.print();">
  Print this page
  <input type = "button" value = "Print" />
 </div>
</div>

<div t:type = "ui/confirmorderdialog"/>
```

```
<span t:type = "ui/courseclassitem" courseClass = "courseClass" linkToLocationsMap =
"true" isList = "false" allowByAplication = "allowByAplication" feeOverride =
"feeOverride"></span>


<div class = "courseDescription expandable">
 <t:outputraw value = "${courseDetail}"></t:outputraw>
</div>

<hr />
<t:if t:test = "${courseClass.hasRoom}">
 <div class = "courseClassLocationFinder">
  <span t:type = "ui/SiteDetailsComponent" room =
"courseClass.room" collapseLocationMap = "true"></span>
 </div>
</t:if>


<t:if t:test = "${showInlineTimetable}">
 <div class = "blockdetail">
  <t:if t:test = "courseClass.hasAnyTimelineableSessions">
  <h2>Sessions</h2>
  <span t:type = "ui/TimetableEvents" displayedObjects = "sortedTimelineableSessions"
  cssTableClass = "cssTableClass"
  cssEvenRowClass = "cssEvenRowClass" cssOddRowClass = "cssOddRowClass"/>
  </t:if>
 </div>
</t:if>

<span t:type = "ui/Attachments" entityIdentifier = "literal:CourseClass" entityIdNum =
"${courseClass.id}"/>

</html>
```

Tutor Details

Filename TutorDetails.tml

URL: not public (ajax only)

Renders a tutor profile in a lightbox.

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <t:if test = "request.XHR">
  <html t:type = "ui/dialogstructure" title = "${tutorDetailsTitle}" bodyId =
"literal:DetailsPage"
     bodyClass = "literal:internal-page"
     xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
   <t:delegate to = "block:tutorDetailsBlock"/>
  </html>
 </t:if>
 <t:if test = "request.XHR" negate = "true">
  <html t:type = "ui/internal/pagestructure" title = "${tutorDetailsTitle}" bodyId =
"literal:DetailsPage"
      bodyClass = "literal:internal-page">
   <div id = "popup-content">
    <h2>${tutorDetailsTitle}</h2>
    <t:delegate to = "block:tutorDetailsBlock"/>
   </div>
  </html>
 </t:if>

 <t:block t:id = "tutorDetailsBlock">
  <div class = "tutors_details">
   <t:if t:test = "${tutorFound}">

    <div class = "tutor_info">
     <t:if t:test = "${profilePicture}">
      <div class = "tutor_pic">
       <img class = "profilePicture" src = "${profilePictureUrl}" alt = ""/>
      </div>
     </t:if>
     <t:if t:test = "${hasResume}">
      <h4>Resume</h4>
      <t:outputraw value = "resume"/>
     </t:if>
    </div>
    <t:if t:test = "${hasRoles}">
     <h4>Classes</h4>
     <div class = "tutors_classes">
      <t:loop t:source = "currentVisibleTutorRoles" t:value = "role">
      <span>
       <a href = "/class/
${role.courseClass.uniqueIdentifier}">${role.courseClass.uniqueIdentifier}
 ${role.courseClass.course.name}</a>
      </span>
      </t:loop>
     </div>
    </t:if>
```

```
    <span t:type = "ui/Attachments" entityIdentifier = "literal:Contact" entityIdNum =
 "${tutor.contact.id}"/>

   </t:if>
  </div>
 </t:block>

</t:container>
```

Sites

Filename Sites.tml

URL: /sites

When called, renders a listing of Course sites (venues) and appropriate maps.

```
<html t:type = "ui/internal/pagestructure" bodyId = "literal:SiteList" bodyClass =
 "literal:internal-page"
   xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <h1>Sites</h1>
 <span t:type = "ui/GoogleMapSites" sites = "sites" showLocationMap = "true"/>

 <ul class = "siteList">
  <t:loop source = "sites" value = "site">
   <li class = "vcard">
    <h3>${site.name}</h3>
                <h3 class = "fn org">
                    <a class = "url" href = "/site/${site.angelId}">${site.name}</a>
                </h3>
                <div class = "sites-loc-wrapper">
                    <div class = "sites-map-link">
                        <t:if t:test = "site.hasCoordinates">
                            <a href = "#" onclick =
 "zoomMapForSite(${site.angelId})" title = "click to show on the map above" class =
 "class_location">Show on map</a>
                            <br />
                            <a href = "/site/${site.angelId}">Details and directions</a>
                            <br />
                        </t:if>
                        <a href = "/courses?site=${site.id}">Show classes</a>
                    </div>
                    <div class = "siteAddress adr">
                        <span class = "street-address">${site.street}</span>
                        <span class = "extended-address">
                            <span class = "locality">${site.suburb}</span>
                            <abbr class = "region">${site.postcode}</abbr>
                            <span class = "postal-code">${site.state}</span>
                        </span>
                    </div>
                </div>
   </li>
  </t:loop>
 </ul>
</html>
```

Site Details

Filename SiteDetails.tml

URL: /site/[id]

Displays the details of the site including the map and site information.

```
<html t:type = "ui/internal/pagestructure" title = "${site.name}" bodyId =
 "literal:DetailPage"
      bodyClass = "literal:internal-page"
      xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <span t:type = "ui/SiteDetailsComponent" site = "site" collapseLocationMap = "false"></
span>
 <span t:type = "ui/Attachments" entityIdentifier = "literal:Site" entityIdNum =
 "site.id"/>
</html>
```

Page Not Found

Filename PageNotFound.tml

URL: undefined

Renders a page when the URL can not be found.

```
<html t:type = "ui/internal/pagestructure" bodyId = "literal:PageNotFound" bodyClass =
 "literal:internal-page"
 xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <h2>Page Not Found</h2>
 <p>The page you are looking for was not found. You may have used an outdated link or may
 have typed the address (URL) incorrectly.</p>
```

```
</html>
```

Promo Codes Page

Filename PromoCodesPage.tml

URL: not public (ajax only)

Allows the user to enter a promotional code within a lightbox.

```
<html t:type = "ui/internal/pagestructure" bodyId = "literal:ListPage" bodyClass =
 "literal:internal-page"
 xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <span t:type = "ui/PromoCodesView"></span>
</html>
```

Room Details

Filename RoomDetails.tml

URL: /room/[id]

Displays details about the room, including the site details.

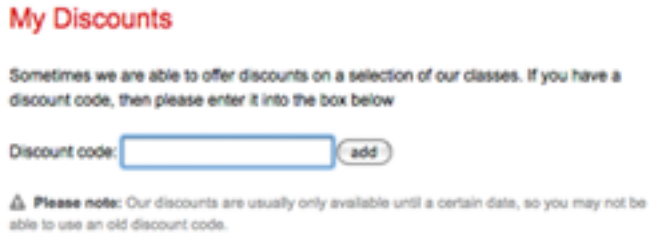
```
<html t:type = "ui/internal/pagestructure" title = "${room.name} |
 ${room.site.name}" bodyId = "literal:DetailsPage"
      bodyClass = "literal:internal-page"
      xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
    <t:if test = "room">
        <h1 class = "roomname">${room.name}</h1>
        <t:if test = "directions">
            <h4 class = "room_directions">Room Directions</h4>
            <p>
                <t:outputRaw value = "directions" />
            </p>
        </t:if>
        <span t:type = "ui/Attachments" entityIdentifiers = "identifiers" entityIdNums =
 "ids"/>
        <span t:type = "ui/SiteDetailsComponent" room = "room" collapseLocationMap =
 "false"/>
    </t:if>
</html>
```

Add Discount

Filename AddDiscount.tml

URL: not public (ajax only)

Allows the user to enter a promotional code within a lightbox.



Discount Redemption

```
<html t:type = "ui/dialogstructure" title = "literal:My Discounts"
   xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">

 <p>Sometimes we are able to offer discounts on a selection of our classes. If you have a
 discount code, then please enter it into the box
  below</p>
 <t:delegate to = "block:discountForm"/>
 <p class = "note"><strong class = "alert">Please note:</strong> Our discounts are usually
 only available until a certain date, so you may not be able to use an old discount
 code.</p>

 <t:block t:id = "discountForm">
  <t:form t:id = "addDiscountForm" zone = "addDiscountZone" clientValidation =
"literal:NONE">
   <label for = "promo">Discount code:</label>
   <t:textField t:id = "promo" value = "promoCode"/>
   <button id = "addDiscountButton" type = "button">add</button>
   <t:eventlink t:id = "addDiscountEvent" style = "{display:none}"/>
   <div class = "validation">${errorMessage}</div>
  </t:form>
  <t:if test = "promotionsList.empty" negate = "true">
```

```
   <br />
   <h3>You have entered the following codes.</h3>
   <t:loop source = "promotionsList" value = "addedPromotion">
    <h5 class = "popup-name">${addedPromotion.code}</h5>
    <div class = "clear"></div>
    <p>
     <t:outputRaw value = "addedPromotion.detail"></t:outputRaw>
    </p>
    <div class = "divideline"></div>
   </t:loop>
   <p>The discounted prices will be now shown next to each course as you browse this
 site.</p>
   <a href = "#" class = "button" onclick = "window.parent.location.reload(true);return
 false;">Close</a>
  </t:if>
 </t:block>
</html>
```

Sitemap XML

Filename SitemapXML.tml

URL: /sitemap.xml

Google, Bing, Yahoo and other search engines love this.

```
<?xml version="1.0" encoding="UTF-8" ?>
<urlset xmlns = "http://www.sitemaps.org/schemas/sitemap/0.9"
        xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd"
  xmlns:p = "tapestry:parameter">
 <url>
  <loc>https://${hostName}/</loc>
  <lastmod>
   <t:output value = "siteModificationDate" format = "dateFormat" />
  </lastmod>
  <changefreq>daily</changefreq>
  <priority>0.3</priority>
 </url>
 <t:loop source = "courses" value = "course">
  <url>
   <loc>https://${hostName}/course/${course.code}</loc>
   <lastmod>
    <t:output value = "course.modified" format = "dateFormat" />
   </lastmod>
   <changefreq>daily</changefreq>
   <priority>0.8</priority>
  </url>
 </t:loop>
 <t:loop source = "sites" value = "site">
  <url>
   <loc>https://${hostName}/site/${site.angelId}</loc>
   <lastmod>
    <t:output value = "site.modified" format = "dateFormat" />
   </lastmod>
   <changefreq>daily</changefreq>
   <priority>0.4</priority>
  </url>
 </t:loop>
 <t:loop source = "tutors" value = "tutor">
       <t:if test = "isActiveTutor()">
           <url>
               <loc>https://${hostName}/tutor/${tutor.angelId}</loc>
               <lastmod>
                   <t:output value = "tutor.modified" format = "dateFormat" />
               </lastmod>
               <changefreq>daily</changefreq>
               <priority>0.4</priority>
           </url>
       </t:if>
 </t:loop>
 <t:loop source = "pages" value = "page">
  <url>
   <t:if test = "${webNodeService.getDefaultWebURLAlias(page)}">
    <loc>https://${hostName}${webNodeService.getDefaultWebURLAlias(page).urlPath}</loc>
    <p:else>
     <loc>https://${hostName}/page/${page.nodeNumber}</loc>
    </p:else>
   </t:if>
   <lastmod>
    <t:output value = "page.modified" format = "dateFormat" />
   </lastmod>
   <changefreq>daily</changefreq>
   <priority>0.7</priority>
  </url>
 </t:loop>
</urlset>
```

Component templates

These templates can never be called directly from an URL. Instead they are used by other templates to build up a page.

Block Display

Filename BlockDisplay.tml

Builds the parameters to include content - copy, images, components

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <t:loop source = "regions" value = "region">
  <t:outputraw value = "regionContent" />
 </t:loop>
</t:container>
```

Body Footer

Filename BodyFooter.tml

Builds the parameters for the Copyright, Disclaimer info, etc at the foot of the page

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <div class = "footer-container">
  <p class = "footer-info">Copyright © 2007-${year} ${collegeName} and
   <a href = "http://www.ish.com.au/copyright.html" title = "ish copyright notice" class =
 "popup">ish</a>, all rights reserved.
  </p>

  <p class = "footer-logo">
   <a href = "http://www.ish.com.au/oncourse" class = "popup">
   <img src = "/s/img/poweredby.png" width = "102" height = "9" alt = "powered by ish
 onCourse"/>
   </a>
  </p>
 </div>
</t:container>
```

Body Header

Filename BodyHeader.tml

Builds the Header area for the mast - logo, navigation, search, etc at the top of the page.

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <hgroup class = "header-hgroup">
  <h2 class = "site-logo" id = "siteLogo">
   <a href = "${homeLink}">${collegeName}</a>
  </h2>
  <h1 class = "site-title" id = "siteTitle">
   <a href = "${homeLink}">${collegeName}</a>
  </h1>
 </hgroup>
 <aside class = "header-toolbar" id = "headerToolbar">
  <span t:type = "ui/CourseSearchForm" />
  <t:if t:test = "paymentGatewayEnabled">
   <span t:type = "ui/ShortList" />
  </t:if>
  <span t:type = "ui/PromoCodesView" />
 </aside>
</t:container>
```

Class Item

Filename CourseClassItem.tml

Produces the brief, panelled class descriptions.

Snapshot display of Course Class and Class Times

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
    <div class = "classItem vevent ${classCommenced}" data-classid = "${courseClass.id}"
         data-sku = "${courseClass.uniqueIdentifier}" itemtype = "http://schema.org/
EducationEvent" itemscope = "itemscope">
```

```
        <!--hcalendar implementation-->
        <div class = "class-item-info">
            <div class = "class-item-info-l">
                <t:if t:test = "courseClass.firstSession">
                    <div class = "date">
                        <a timezone = "10" rel = "session" href = "/Timeline/sessions?ids=
${courseClass.id}" class = "timeline">
                            <abbr itemprop = "startDate" class = "dtstart" title = "">
                                <t:output format = "startDateFormat" value =
 "firstSessionStartDate"/>
                            </abbr>

                            <t:if t:test = "showDateEnd"> - <abbr class =
 "dtend" itemprop = "endDate" title = "">
                                <t:output format = "endDateFormat" value =
 "courseClass.endDate"/>
                            </abbr>
                            </t:if>
                        </a>
                        <br/>
                        <a timezone = "10" href = "/Timeline/sessions?ids=
${courseClass.id}" class = "timeline" rel = "session">
                            <abbr itemprop = "startDate" class = "dtstart" title = "">
                                <t:output format = "timeFormat" value =
 "firstSessionStartDate"/>
                            </abbr> - <abbr itemprop = "endDate" class = "dtend" title =
 "">
                            <t:output format = "timeFormatWithTimeZone" value =
 "firstSessionEndDate"/>
                        </abbr>
                            <t:if t:test = "hasManySessions"> (first session)</t:if>
                        </a>
                        <br/>
                        <a timezone = "10" href = "/Timeline/sessions?ids=
${courseClass.id}" itemprop = "duration"
                           class = "timeline" rel = "session">${classSessions}</a>
                    </div>

                    <p:else>
                        <t:if t:test = "selfPacedClass">
                            <div class = "date">Self paced<br/>${expectedHours}<br/
>${maximumDaysToComplete}</div>
                            <p:else>
                                <b>no session</b>
                            </p:else>
                            </t:if>
                        </p:else>
                    </t:if>
                    <div itemprop = "name" class = "summary">${courseClass.course.name}</div>
                </div>
                <div class = "class-item-info-r">
                    <t:if test = "hasLinkToLocation">
                        <div itemprop = "location" class = "location">
                            <t:if test = "courseClass.hasRoom">
                                <span t:type = "ui/roomLocation" room =
 "courseClass.room" withRoomName = "false"
                                      disabledLink = "notExistsOnMap" isList = "isList"/>
                                <p:else>Venue TBA</p:else>
                            </t:if>
                        </div>
                    </t:if>
                    <t:if test = "hasTutorRoles">
                        <div class = "tutor">
                            <t:loop source = "visibleTutorRoles" value = "tutorRole" index =
 "index">
                                <a class = "nyromodal" itemprop = "performers" href = "/tutor/
${tutorRole.tutor.angelId}"
                                   title = "View tutor profile">${tutorName}
                                </a>
                                <t:if test = "lastIndex" negate = "true">
                                    ,
                                </t:if>
                            </t:loop>
                        </div>
                    </t:if>
                <t:if test = "allowByAplication" negate = "true">
                 <div class = "price">
                  <span t:type = "ui/CourseClassPrice" courseClass = "courseClass" feeOverride =
 "feeOverride"/>
                </div>
                </t:if>
                    </div>
                </div>
                <div class = "classAction">
                    <t:if t:test = "courseClass.cancelled">
                        <a class = "enrolAction disabled" disabled = "true" title =
 "${enrolHoverTitle}">Cancelled</a>
                    </t:if>
                    <t:if t:test = "currentClass">
                        <t:if t:test = "hasAvailableEnrolmentPlaces">
                            <t:if t:test = "paymentGatewayEnabled">
```

```
                    <t:if test = "addedClass">
                        <a class = "enrolAction enrol-added-class"
                           href = "/addToCookies?key=shortlist&amp;addItemId=
${courseClass.id}">Added</a>
                        <p:else>
        <t:if test = "allowByAplication">
         <a class = "enrolAction"
            href = "/addToCookies?key=shortlist&amp;addItemId=${courseClass.id}"
            title = "${enrolHoverTitle}" data = "application">Apply Now</a>
         <p:else>
          <a class = "enrolAction"
             href = "/addToCookies?key=shortlist&amp;addItemId=${courseClass.id}"
             title = "${enrolHoverTitle}">Enrol Now</a>
         </p:else>
        </t:if>
                        </p:else>
                    </t:if>
                </t:if>
                <div class = "classStatus">
                    <span t:type = "ui/CourseClassPlacesAvailable" courseClass =
 "courseClass"/>
                </div>
                <p:else>
                    <t:if t:test = "finishedClass">
                        <a class = "enrolAction disabled" disabled = "true" title =
 "${enrolHoverTitle}">Finished</a>
                    </t:if>
                </p:else>
            </t:if>


            <!-- div class="courseEmail">
             <a href="">Email to a friend</a>
         </div-->
        </div>


        <t:if test = "courseClass.detail">
            <div class = "classDescription class-note-important">
                <t:outputraw value = "courseClassDetail"/>
            </div>
        </t:if>


        <div class = "bubbleInfo">
            <div class = "tooltip_popup enrol_tooltip">
                <div class = "arrow">
                    <span/>
                </div>
                <div class = "bubble_top bubble_content">
    <span class = "timing-display">
     <t:loop source = "weekdays" value = "day" index = "dayIndex">
                            <span class = "timing-week${dayKind} timing-${day} ${dayClass}
 timing-${hasTiming}">${dayShortName}</span>
                        </t:loop>
    </span>
    <span class = "timing-display">
     <span class = "timing-daytime match-10 timing-daytime-${dayTimeClass}">
      <img alt = "" src = "/s/img/blank.png"/>
     </span>
     <span class = "timing-evening match-10 timing-evening-${eveningClass}">
      <img alt = "" src = "/s/img/blank.png"/>
     </span>
    </span>
                </div>
                <div class = "bubble_middle bubble_content">
                    <div class = "class-link">
                        <a href = "/class/${courseClass.course.code}-
${courseClass.code}" itemprop = "url"> View this
                            class... </a>
                    </div>
                    <div class = "course-link">
                        <a href = "/course/${courseClass.course.code}"> More
 about <em>${courseClass.course.name}</em>...
                        </a>
                    </div>
                </div>
                <div class = "bubble_bottom"/>
            </div>
        </div>
        <t:if test = "${courseClass.cancelled}" negate = "true">
            <t:if test = "courseClass.hasAnyTimelineableSessions">
                <div class = "sessions_for_class hidden classSessions" id =
 "sessions_for_class_${courseClass.id}">
    <span t:type = "ui/TimetableEvents" displayedObjects = "sortedTimelineableSessions"
                        cssTableClass = "cssTableClass" cssEvenRowClass =
 "cssEvenRowClass"
                        cssOddRowClass = "cssOddRowClass"/>
                </div>
            </t:if>
        </t:if>
```

```
    </div>
</t:container>
```

Course Class Places Available

Filename CourseClassPlacesAvailable.tml

Produces a Class enrolment status display. Called by "CourseClassItem".

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
    <t:if test = "hasAvailableEnrolmentPlaces()">
        <t:if test = "hasCommonAvailable">
            There are places available
            <p:else>
                There
                <t:if test = "hasManyPlaces"> are
                    <t:output value = "courseClass.availableEnrolmentPlaces" format =
 "format"/>
                    places
                    <p:else>is one place</p:else>
                </t:if>
                available
            </p:else>
        </t:if>
        <p:else>
            <a class = "enrolAction disabled" disabled = "true">Class Full</a>
        </p:else>
    </t:if>
</t:container>
```

Course Class Price

Filename CourseClassPrice.tml

Produces a class pricing display. Called by "CourseClassItem".

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">

 <t:if test = "overridden">
  <t:output value = "feeOverride" format = "feeFormat"/>
  <t:delegate to = "block:tax"/>
 <p:else>
  <t:if test = "hasFee">
   <t:if test = "hasDiscountValue">
  <span class = "fee-disabled">
   <t:output value = "fee" format = "feeFormat" />
  </span>
  <span class = "fee-discounted">
   <acronym title = "${appliedDiscountsTitle}">
    <t:output value = "discountedFee" format = "feeFormat" />
   </acronym>
  </span>
    <p:else><t:output value = "fee" format = "feeFormat" /></p:else>
   </t:if>

   <t:delegate to = "block:tax"/>

   <t:if t:test = "paymentGatewayEnabled">
    <t:loop source = "discountItems" value = "discountItem">
     <span class = "discount-price">/</span>
     <acronym class = "discount-price" title = "${discountItem.title}">
      <t:output value = "discountItem.feeIncTax" format = "discountItem.feeFormat" />
     </acronym>
    </t:loop>
   </t:if>

   <p:else>
    <acronym title = "To Be Advised">TBA</acronym>
   </p:else>

  </t:if>
 </p:else>
</t:if>

<t:block t:id = "tax">
 <t:if test = "showTax">
  <span class = "gst">
   <t:if test = "taxExempt" negate = "true"> inc </t:if><acronym title = "Goods and
 Services Tax">GST</acronym><t:if test = "taxExempt"> free</t:if>
  </span>
 </t:if>
</t:block>

</t:container>
```

Course item

Filename CourseItem.tml

Produces a brief introduction to a Course Class

Called by "Courses" and "CourseDetails"

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <div itemtype = "http://schema.org/Product" itemscope = "itemscope" class = "vevent
 courseItem${tagClasses}">
  <t:if t:test = "isList" negate = "true">
   <h1 class = "summary page-title">${courseItemModel.course.name}</h1>
   <t:if t:test = "courseItemModel.nrt">
    <img class = "NRTLogo" alt = "NRT" src = "/s/img/core/nrt.png"/>
   </t:if>
   <t:if t:test = "addThisEnabled">
    <div class = "addthis">
     <span t:type = "ui/socialmedia" addThisProfileId = "addThisProfileId"/>
    </div>
   </t:if>
   <div class = "clearboth"></div>
  <p:else>
   <h2 itemprop = "name">
    <a href = "/course/${courseItemModel.course.code}" class =
 "url">${courseItemModel.course.name}</a>
    <t:if t:test = "courseItemModel.nrt">
     <img class = "NRTLogo" alt = "NRT" src = "/s/img/core/nrt.png"/>
    </t:if>
   </h2>
   <div class = "description courseShortDescription">
    <t:outputraw value = "${courseDetail}"/>
    <strong class = "more">[<a href = "/course/${courseItemModel.course.code}">More</a>]</
strong>
   </div>
  </p:else>
  </t:if>
  <t:if t:test = "${courseItemModel.showModules}">
   <div t:type = "ui/CourseModules" modules = "courseItemModel.course.modules"
     qualification = "courseItemModel.course.qualification"
     course = "courseItemModel.course"></div>
  </t:if>
  <t:loop source = "courseItemModel.availableClasses" value = "courseClass">
   <span t:type = "ui/courseclassitem" courseClass = "courseClass" linkToLocationsMap =
"linkToLocationsMap" isList = "isList" allowByAplication =
"allowByAplication" feeOverride = "feeOverride"/>
  </t:loop>
  <t:if test = "courseItemModel.otherClasses.empty" negate = "true">
   <p class = "other-classes-control clearfix">
    <a href = "javascript:return false;" >Show other classes</a>
   </p>
   <div class = "other-classes" style = "display:none;">
    <t:loop source = "courseItemModel.otherClasses" value = "courseClass">
     <span t:type = "ui/courseclassitem" courseClass = "courseClass" linkToLocationsMap =
"linkToLocationsMap" isList = "isList" allowByAplication =
"allowByAplication" feeOverride = "feeOverride"/>
    </t:loop>
   </div>
  </t:if>
  <t:if test = "courseItemModel.fullClasses.empty" negate = "true">
   <p class = "full-classes-control clearfix">
    <a href = "javascript:return false;" >Show full classes</a>
   </p>
   <div class = "full-classes" style = "display:none;">
    <t:loop source = "courseItemModel.fullClasses" value = "courseClass">
     <span t:type = "ui/courseclassitem" courseClass = "courseClass" linkToLocationsMap =
"linkToLocationsMap" isList = "isList" allowByAplication =
"allowByAplication" feeOverride = "feeOverride"/>
    </t:loop>
   </div>
  </t:if>
  <t:if t:test = "courseItemModel.course.allowWaitingList">
   <p class = "waiting-list-title" id = "wl${courseItemModel.course.id}">
    <a class = "actionLink" href = "/enrol/waitinglistform/${courseItemModel.course.id}">
     <t:if t:test = "courseItemModel.enrollableClasses.empty" negate = "true">
      <t:if t:test = "hasMoreAvailablePlaces"> If there isn't a class to suit you,
please <p:else> Classes are full. Please </p:else>
     </t:if>
     <p:else> This course has no current classes. Please </p:else>
    </t:if>
    <img src = "/s/img/button_join.png" alt = "join" /> the waiting list. </a>
   </p>
  </t:if>
  <t:if t:test = "isList" negate = "true">
   <div class = "courseDescription">
    <t:outputraw value = "${courseDetail}" />
   </div>
  </t:if>
  <t:if t:test = "courseItemModel.haveRelatedCourses">
```

```
   <h3>Related Courses</h3>
   <div t:type = "ui/CourseRelations" course = "courseItemModel.course"></div>
  </t:if>
  <t:if t:test = "courseItemModel.hasRelatedProducts()">
   <h3>Related Products</h3>
   <div t:type = "ui/relatedporducts" model = "courseItemModel"></div>
  </t:if>
 </div>
</t:container>
```

Course Search Form

Filename CourseSearchForm.tml

Produces a Course Class specific search

Called by "BodyHeader" and in turn calls "Search Inputs"

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <div class = "noprint search-box" id = "search_box">
  <h3 class = "search-box-title">
   <span>Search</span>
  </h3>
  <form action = "/courses" method = "get" id = "search" name = "search">
   <input type = "text" id = "s" class = "quicksearch" name = "s" size =
 "15" autocomplete = "off" placeholder = "search" />
   <button type = "submit" id = "find" class = "find">Go!</button>
  </form>
        <!-- searchTagNames="literal:Subjects" by default even if no property specified,
        if we need to add more then one element we need to separate then using ';'
 character without extra spaces
        for example searchTagNames="literal:Subjects;some_group_tag_1;some_group_tag_1"
        please note that group elements available only for tag group with valid names -->
  <span t:type = "ui/SearchInputs" searchTagNames = "literal:Subjects"/>
  <div class = "advanced-search-button">
   <a class = "show-advanced-search">
    <span>More options</span>
   </a>
  </div>
  <div class = "quicksearch-wrap-container">
   <div class = "quicksearch-wrap" />
  </div>
 </div>
</t:container>
```

Global Navi

Filename GlobalNavi.tml

Defines the parameters for global menus

Called by "BodyStructure"

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter" />
```

Google Analytics

Filename GoogleAnalytics.tml

Places the appropriate Google Analytics code on the page

Called by "CourseClassItem" in "CourseClassDetails"

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <t:if test = "analyticsAccount">
    <script type = "text/javascript">
     (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
    (i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
    m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
   })(window,document,'script','//www.google-analytics.com/analytics.js','ga');

   ga('create', '${analyticsAccount}');
   ga('send', 'pageview');
     </script>
 </t:if>
</t:container>
```

Google Map Sites

Filename GoogleMapSites.tml

Produces a Google map of the predefined site/sites/venue

Called by "Courses" and "Sites" and in turn calls "GoogleDirections"

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
```

```
 <t:if t:test = "hasMapItemList">

  <div id = "focus-map" class = "${focusMapClass}">
   <span t:type = "ui/GoogleDirections" sites = "sites" focuses =
 "focuses" showLocationMap = "showLocationMap" />
  </div>

  <div id = "gmapCanvas"></div>

 </t:if>
</t:container>
```

Google Directions

Filename GoogleDirections.tml

Produces directions - written and verbal for site directions

Called by "GoogleMapSites" and "SiteDetailsComponent"

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd"
  >
 <t:if t:test = "hasLocation">

  <script type = "text/javascript">


   function initialize() {
    vLat = "${mapPositionLatitude}";
    vLong = "${mapPositionLongitude}";
    gMapOptions = {
     zoom: 14,
     <t:if test = "needCenter()">
      center: new google.maps.LatLng(vLat, vLong),
     </t:if>
     mapTypeId: google.maps.MapTypeId.ROADMAP
    };
    near = "${searchingNear}";
    vSiteAddress = "<t:output value = "address" format = "jsonFormat" filter = "true"/>";
    vFrom = "<t:output value = "directionsFrom" format = "jsonFormat" filter = "true"/>";
    showNearMarker = "${hasGlobalPosition}";
    showMapItems = "${showMapItems}";
    zoom = null;
    siteMarkers = new Array();
    gMapSites = ${sitesArray};
    mapLoaded=false;

    mapLoad('gmapCanvas', gMapSites, gMapOptions);
   }

   /**
    * the function makes async call to load google map
    */
   function loadMap() {
    var script = document.createElement('script');
    script.type = 'text/javascript';
    //we use this hard code to convert &amp; to real amp.
    script.src = jQuery('<div></div>').html('https://maps.googleapis.com/maps/api/js?
v=3.exp&amp;sensor=false&amp;callback=initialize').text();
    document.body.appendChild(script);
   }

   jQuery(document).ready(function() {
    if (jQuery('#gmapCanvas').length) {
     if (typeof google === 'object') {
      if (typeof google.maps === 'object') {
       initialize();
      }
     }
     else {
      loadMap();
     }
    }
   });
  </script>

  <div id = "directions" class = "directions-wrapper">
   <t:if t:test = "hideDirections" negate = "true">
    <form id = "getdir" method = "post" onsubmit = "return false;">
     <label>Directions from:</label>
     <input type = "text" size = "25" id = "from" name = "from"
      value = "${directionsFrom}" />
     <input name = "submit" type = "button" value = "Get Directions!"
      class = "getDirections" onclick = "dirLoadForId('gmapCanvas');" />
    </form>
    <div id = "dirtxt" class = "" />
   </t:if>
  </div>
 </t:if>
</t:container>
```

Hint Component

Filename HintComponent.tml

Provides validation text (showing data entry errors) for forms throughout the application, but particularly within the enrolment process.

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <span class = "validate-text">
  <t:if test = "validationMessage">
   <span class = "reason hidden-text"> ${validationMessage} <span class = "reason-pointer"
 />
   </span>
  </t:if>
  <t:if test = "hint">
   <span class = "hint hidden-text" id = "givenName-hint"> ${hint} <span class = "hint-
pointer" />
   </span>
  </t:if>
 </span>
</t:container>
```

Menu

Filename Menu.tml

Renders a Menu display. Called by "BodyStructure".

Course Class

```
<ul xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <t:loop source = "children" value = "currentMenu">
  <t:delegate to = "menuItem" />
 </t:loop>
 <t:block>
  <div t:id = "menuItem">
   <t:delegate to = "menuItem" />
  </div>
 </t:block>
</ul>
```

Menu Item

Filename MenuItem.tml

Renders a specific menu. Called by "Menu"

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <li class = "${menuItemClass}">
  <a href = "${itemHref}">${menu.name}</a>
  <ul><t:body /></ul>
 </li>
</t:container>
```

Page Head

Filename PageHead.tml

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <meta http-equiv = "content-type" content = "text/html; charset=utf-8" />
 <t:any element = "meta" name = "generator" content = "prop:metaGeneratorContent" />

 <t:if test = "canonicalLinkPath">
     <link rel = "canonical" href = "${canonicalLinkPath}"></link>
  <meta property = "og:url" content = "${canonicalLinkPath}" />
 </t:if>

 <t:if test = "canonicalRelativeLinkPath">
  <link rel = "canonical" href = "${canonicalRelativeLinkPath}"></link>
 </t:if>

 <meta property = "og:type" content = "website"/>
 <meta property = "og:image" content = "" />

 <meta content = "${metaDescription}" name = "description" property = "og:description"/>

 <meta http-equiv = "X-UA-Compatible" content = "IE=edge,chrome=1" />
 <meta name = "viewport" content = "width=device-width, initial-scale=1.0" />

 <div t:type = "ui/courseClassMetaInfo"/>

 <link rel = "shortcut icon" type = "image/ico" href = "/s/img/favicon.ico" />

    <span t:type = "ui/title" pageName = "pageName" webNode = "webNode"/>

 <meta name = "author" content = "" />
 <!--[if lt IE 9]><script src="/s/js/html5.js"></script><![endif]-->
 <link href = "/s/stylesheets/css/site.css?v=${ciVersion}" media = "screen, print" rel =
 "stylesheet" />
    <script type = "text/javascript" src = "/s/js/all.js?v=${ciVersion}"></script>

</t:container>
```

Payment Agreement

Filename PaymentAgreement.tml

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
    <t:if test = "preferenceController.featureEnrolmentDisclosure">
        I have read the <a class = "nyromodal" title = "Student information" href =
 "${preferenceController.featureEnrolmentDisclosure}?wrap=false">Student Information</a>
 or have had it explained to me, and I agree to accept these  conditions.
    </t:if>
    <t:if test = "preferenceController.refundPolicyUrl">
        I understand the <a href = "${preferenceController.refundPolicyUrl}" target =
 "_blank">enrolment, sale and refund policy</a>.
       <p:else>
           I understand the enrolment, sale and refund policy.
       </p:else>
    </t:if>
</t:container>
```

Promo Codes View

Filename PromoCodesView.tml

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <t:if t:test = "paymentGatewayEnabled">
  <div id = "discounts_box" class = "discount-block box">
   <h3 class = "title-block">
    <span>My Discounts</span>
   </h3>

   <t:if t:test = "promotions.empty" negate = "true">
    <ul class = "menu">
     <t:loop source = "promotions" value = "promotion">
      <li class = "onshortlist-x">
```

```
      <a class = "cutitem" id = "${promotion.id}" href = "/removeFromCookies?
key=promotions&amp;removeItemId=${promotion.id}">
       <span class = "hide" title = "Remove item">X</span>
      </a>
      <a class = "viewitem" title = "${promotion.code}" href = "#" disabled =
 "true">${promotion.name}</a>
     </li>
    </t:loop>
   </ul>
  </t:if>
  <a class = "nyromodalreload actionLink" target = "_blank" href = "http://${serverName}/
addDiscount">Add a discount code</a>
  <a class = "actionLink" target = "_blank" href = "https://skillsoncourse.com.au/
login">Manage your classes</a>
 </div>
</t:if>
</t:container>
```

Quick Search View

Filename QuickSearchView.tml

Produces specific parameters for the display of the page masthead

Called by "PageStructure"

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <div>
  <t:if t:test = "hasResults" negate = "true">
   <b>
    <a href = "courses?s=${searchString}">Search...</a>
   </b>
   <p:else>
    <t:if t:test = "hasLocationDetailList">
     <ul class = "suburb_list">
      <li class = "title"> Show courses near... </li>
      <t:loop source = "locationDetailList" value = "location">
       <li class = "location hoverable">
        <a href = "/courses?near=${location.displayName}">${location.displayName}</a>
       </li>
      </t:loop>
     </ul>
    </t:if>
    <t:if test = "hasMatchingCourseList">
     <ul class = "course_list">
      <li class = "title">Matching courses</li>
      <t:loop source = "matchingCourseList" value = "course">
       <li class = "course hoverable">
        <a href = "/course/${course.code}"> ${course.name}- <em>${course.code}</em>
        </a>
       </li>
      </t:loop>
      <li class = "quicksearch-all">
       <a href = "/courses?s=${searchString}">Show all results...</a>
      </li>
     </ul>
     <p:else>
      <t:if test = "hasCourseList">
       <ul class = "course_list">
        <li class = "title">Courses</li>
        <t:loop source = "courseList" value = "course">
         <li class = "course hoverable">
          <a href = "/course/${course.code}">
           <em>${course.name}</em>
          </a>
         </li>
        </t:loop>
        <li class = "quicksearch-all">
         <a href = "/courses?s=${searchString}">Show all results...</a>
        </li>
       </ul>
      </t:if>
     </p:else>
    </t:if>
    <t:if test = "hasTagGroupResultsList">
     <ul class = "tag_list">
      <li class = "title">Subjects</li>
      <t:loop source = "tags" value = "tag">
       <li>
        <a href = "${tag.link}">${tag.name}</a>
       </li>
      </t:loop>
     </ul>
    </t:if>
   </p:else>
  </t:if>
 </div>
</t:container>
```

Room Location

Filename RoomLocation.tml

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <t:if test = "room.site.isWebVisible">
  <t:unless test = "disabledLink">
   <t:if test = "redirect">
    <a href = "${mapLink}">
     <span t:type = "ui/RoomLocationText" room = "room" withRoomName =
 "withRoomName" withSiteAddress = "withSiteAddress" />
    </a>
    <p:else>
     <a title = "Show venue details" class = "tooltip" onclick =
 "zoomMapForSite(${room.site.angelId});" href = "#">
      <span t:type = "ui/RoomLocationText" room = "room" withRoomName =
 "withRoomName" withSiteAddress = "withSiteAddress" />
     </a>
    </p:else>
   </t:if>
   <p:else>
    <span t:type = "ui/RoomLocationText" room = "room" withRoomName =
 "withRoomName" withSiteAddress = "withSiteAddress" />
   </p:else>
  </t:unless>
 </t:if>
</t:container>
```

Room Location Text

Filename RoomLocationText.tml

Produces specific parameters for the display of the page masthead

Called by "PageStructure"

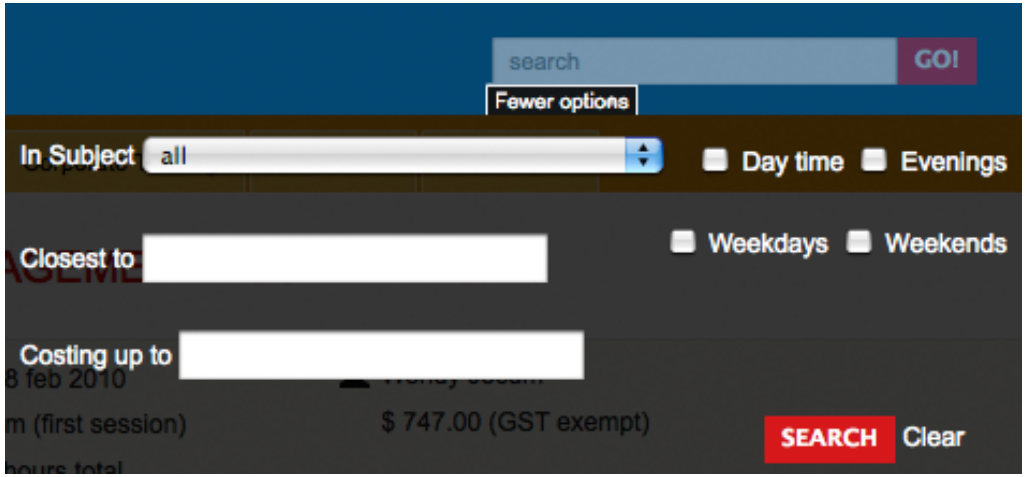
```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <t:if test = "withRoomName">
  <t:if test = "hasRoomName">${room.name}</t:if>
 </t:if>
 <t:if test = "hasSiteName">
  <t:if test = "withRoomName"> - </t:if> ${room.site.name} <t:if t:test =
 "withSiteAddress">
   <span> ${room.site.street} ${room.site.suburb} ${room.site.postcode} </span>
  </t:if>
 </t:if>
</t:container>
```

Search Criteria

Filename SearchCriteria.tml

Produces specific parameters for the display of the page masthead

Called by "PageStructure"



Search Options

Produces specific parameters for the display of the page masthead

Called by "PageStructure"

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <div class = "search-terms">
  <div id = "search_query"> Results for <t:if t:test = "browseTagPath">
   <t:loop source = "browseTagPath" value = "tag" index = "tagIndex">
    <div class = "tag-crumb-title search-value no-highlight">
     <t:if t:test = "showTagRaquo"><a href = "${tag.link}">${tag.name}</a>
```

```
      <p:else>${tag.name}</p:else>
     </t:if>
    </div>
    <t:if t:test = "showTagRaquo">
     <span>»</span>
    </t:if>
   </t:loop>
  </t:if>
   <t:if t:test = "searchString">
    <span class = "search-value no-highlight">${searchString}</span>
   </t:if>
   <t:if t:test = "hasSearchDays"> on </t:if>
   <t:if t:test = "searchingWeekday">
    <span class = "match-8 search-value">Week day</span>
    <t:remove> <!-- TODO seems to be unreachable code, sort out
     <p:else>
     <t:if t:test="$hasSearchDaysDuringWeek">
     <t:loop source="$searchDaysDuringWeek" value="$listItem">
     <span class="~dayCssClass + ' search-value'">
     <wo:string value="$listItem" />
     </span>
     </t:loop>
     </t:if>
     </p:else>
    -->
    </t:remove>
   </t:if>
   <t:if t:test = "searchingWeekend">
    <span class = "match-9 search-value">Weekend</span>
    <t:remove>
     <!-- TODO seems to be unreachable code, sort out
      <p:else>
      <t:if t:test="$hasSearchDaysDuringWeekend">
      <t:loop source="$searchDaysDuringWeekend" value="$listItem">
      <span class="~dayCssClass + ' search-value'">
      <wo:string value="$listItem" />
      </span>
      </t:loop>
      </t:if>
      </p:else>
     -->
    </t:remove>
   </t:if>
   <t:if t:test = "searchTime">in the <span class = "match-10 search-
value">${searchTime}</span></t:if>
   <t:if t:test = "searchPrice">up to <span class = "match-12 search-
value">${searchPrice}</span></t:if>
   <t:if t:test = "searchNear"><span class = "match-11 search-value"> near ${searchNear}</
span></t:if>
            <t:if t:test = "tagNames">
                <t:loop source = "tagNames" value = "tagName" index = "tagIndex">
                    <t:if t:test = "addTagComma">
                        <span>;</span>
                    </t:if>
                    <span class = "tag-crumb-title search-value no-highlight">${tagName}</
span>
                </t:loop>
            </t:if>
  </div>
  <t:if t:test = "hasMapItemList">
   <div id = "toggle-results-map">
    <a href = "#" class = "toggle_locations">locations</a>
   </div>
  </t:if>
 </div>
</t:container>
```

Search Inputs

Filename SearchInputs.tml

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <div id = "advanced_search_container">
  <div class = "" id = "advanced_search">
   <!--clientValidation should be NONE to exclude usage Tapestry java script injection-->
   <t:form t:id = "search2" class = "advanced-search-form" method = "get" name =
 "search" secure = "request.isSecure()" clientValidation = "literal:NONE">
    <div id = "adv_search_keyword">
     <label title = "Search keywords and functions" for = "adv_keyword">I'm Looking For</
label>
     <input t:type = "TextField" clientId = "adv_keyword" t:value = "advKeyword" default =
 "keyword" name = "s" />
    </div>
    <div id = "adv_search_tag">
                    <t:loop source = "tagGroups" value = "tagGroup">
                        <label for = "tagGroup">In ${tagGroup}</label>
                        <t:select id = "tagGroup" name = "subject" model =
 "tagGroupModel" blankLabel = "all" value = "selectedTagValue" />
                    </t:loop>
```

```
    </div>
    <div id = "adv_search_location">
     <label title = "Enter a suburb or postcode near where you'd like to do the
 course" for = "suburb-autocomplete">Closest to</label>
     <input t:type = "TextField" name = "near" t:value = "searchNear" clientId = "suburb-
autocomplete" class = "suburb-autocomplete"
      default = "postcode or suburb" size = "18" autocomplete = "off" />
    </div>
    <div id = "adv_search_price">
     <label title = "Select the maximum price you'd like to pay for a course" for =
 "adv_price">Costing up to</label>
     <input t:type = "TextField" clientId = "adv_price" t:value = "searchPrice" name =
 "price" size = "8" />
    </div>
    <div id = "adv_search_time">
     <label class = "time checkbox">
      <input t:type = "Checkbox" t:value = "daytime" name = "time" /> Day time </label>
     <label class = "time checkbox">
      <input t:type = "Checkbox" t:value = "evening" name = "time" /> Evenings </label>
    </div>
    <div id = "adv_search_day">
     <label class = "day checkbox">
      <input t:type = "Checkbox" t:value = "weekday" name = "day" class =
 "parent" clientId = "weekday-parent" /> Weekdays </label>
     <label class = "day checkbox">
      <input t:type = "Checkbox" t:value = "weekend" name = "day" class =
 "parent" clientId = "weekend-parent" /> Weekends </label>
    </div>
    <div id = "adv_search_submit">
     <button type = "submit" id = "searcher" class = "btn btn-primary">Search</button>
     <a id = "cancel-search" onClick = "clearAdvancedSearch();" class = "btn">Clear</a>
    </div>
   </t:form>
  </div>
 </div>
</t:container>
```

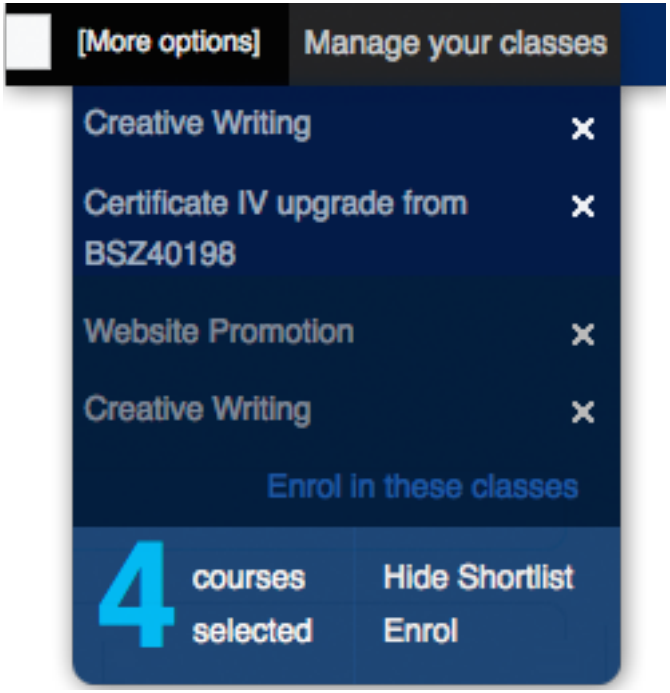
Search Terms Clarification

Filename SearchTermsClarification.tml

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <t:if test = "paramsInError">
  <t:if test = "searchNear">
   <p class = "invalid_search_near">Sorry, the location &quot;${searchNear}&quot; is not
 recognised.</p>
   <p>Please try a different location.</p>
  </t:if>
  <t:if test = "searchDay">
   <p class = "invalid_search_day">Sorry, the day &quot;${searchDay}&quot; is not
 recognised.</p>
   <p>Please try a different day.</p>
  </t:if>
  <t:if test = "searchTime">
   <p class = "invalid_search_time">Sorry, the time of day &quot;${searchTime}&quot; is
 not recognised.</p>
   <p>Please try a different time of day.</p>
  </t:if>
  <t:if test = "searchPrice">
   <p class = "invalid_search_price">Sorry, the price &quot;${searchPrice}&quot; is not
 recognised.</p>
   <p>Please try a different price.</p>
  </t:if>
  <t:if test = "searchTag">
   <p class = "invalid_search_tag">Sorry, the subject with path &quot;${searchTag}&quot;
 is not recognised.</p>
   <p>Please try a different subject.</p>
  </t:if>
  <t:if test = "km">
   <p class = "invalid_km">Sorry, the km distance with value &quot;${km}&quot; is not
 recognised.</p>
   <p>Please try a different digits only km value.</p>
  </t:if>
        <t:if test = "after">
            <p class = "invalid_after">Sorry, the after date with value &quot;
${after}&quot; is not recognised.</p>
            <p>Please try a different date with format &quot;YYYYMMDD&quot;.</p>
        </t:if>
        <t:if test = "before">
            <p class = "invalid_before">Sorry, the before date with value &quot;
${before}&quot; is not recognised.</p>
            <p>Please try a different date with format &quot;YYYYMMDD&quot;.</p>
        </t:if>
    </t:if>
</t:container>
```

Shortlist

Filename ShortList.tml



Short List Manager

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <div id = "shortlist" class = "short-list">
  <div class = "shortlistInfo" id = "info">
   <span>${totatItemsCount}</span>
   <p>items</p>
  </div>
  <t:if test = "hasAnyItems">
   <h3 class = "title-block">
    <span>My Courses Order</span>
   </h3>
   <div class = "${classForList}">
    <ul class = "shortListOrder shortlistChoices">
     <t:loop source = "items" value = "courseClass">
      <li data-classid = "${courseClass.id}">
       <a href = "/class/${courseClass.uniqueIdentifier}">${courseClass.course.name}</a>
       <span class = "deleteItem" title = "Remove item">
        <a href = "/removeFromCookies?key=shortlist&amp;removeItemId=
${courseClass.id}">X</a>
       </span>
       <div class = "shortListOrderClasses">
        <t:if t:test = "courseClass.firstSession">
         <abbr class = "dtstart" title = "">
          <t:output format = "dateFormat" value = "courseClass.firstSession.startDate" />
         </abbr>
        </t:if>
        <t:if t:test = "courseClass.endDate"> - <abbr class = "dtend" title =
 ""><t:output format = "dateFormat"
           value = "courseClass.endDate" /></abbr>
        </t:if>
       </div>
      </li>
     </t:loop>
     <t:loop source = "productItems" value = "product">
      <li data-classid = "${product.id}">
       <a href = "/product/${product.sku}">${product.name}</a>
       <span class = "deleteItem" title = "Remove item">
        <a href = "/removeFromCookies?key=productShortList&amp;removeItemId=
${product.id}">X</a>
       </span>
       <div class = "shortListOrderClasses">
        <t:if t:test = "product.created">
         <abbr class = "dtstart" title = "">
          <t:output format = "productDateFormat" value = "product.created" />
         </abbr>
        </t:if>
        <t:if t:test = "product.modified"> - <abbr class = "dtend" title =
 ""><t:output format = "productDateFormat"
           value = "product.modified" /></abbr>
        </t:if>
       </div>
      </li>
     </t:loop>
     <li class = "shortListOrderEnrol">
      <t:if test = "hasItems">
       <a href = "/enrol/" class = "shortlistLinkEnrol">Enrol</a>
       <p:else>
        <a href = "/enrol/" class = "shortlistLinkEnrol">Purchase</a>
       </p:else>
      </t:if>
```

```
    </li>
   </ul>
   <t:if test = "expandComponent" negate = "true">
    <div class = "closeButton">X</div>
   </t:if>
  </div>
  <div class = "shortlistAction">
   <ul class = "shortlistControls">
    <t:if test = "expandComponent" negate = "true">
     <li class = "shortlistActionShow">
      <a href = "#">Show Shortlist</a>
     </li>
     <li class = "shortlistActionHide">
      <a href = "#">Hide Shortlist</a>
     </li>
    </t:if>
    <li class = "shortlistActionEnrol">
     <t:if test = "hasItems">
      <a href = "/enrol/">Enrol</a>
      <p:else>
       <a href = "/enrol/">Purchase</a>
      </p:else>
     </t:if>
    </li>
   </ul>
  </div>
 </t:if>
</div>
</t:container>
```

Site Details Component

Filename SiteDetailsComponent.tml

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <div class = "vcard locationInfo">
  <h1>${site.name}</h1>
  <div class = "adr" id = "">
   <t:if t:test = "hasAddress">
    <p>
     <div class = "street-address">${site.street}</div>
     <span class = "locality">${site.suburb}</span>
     <t:if t:test = "hasPostCode">
      <span class = "postal-code">${site.postcode}</span>
     </t:if>
    </p>
   </t:if>

   <t:if t:test = "hasDrivingDirections">
    <h4 class = "locationInfo_driving">Site Directions</h4>
    <p>
     <t:outputRaw value = "drivingDirections" />
    </p>
   </t:if>
   <t:if t:test = "hasPublicTransportDirections">
    <h4 class = "locationInfo_publicTransport">Public Transport</h4>
    <p>
     <t:outputRaw value = "publicTransportDirections" />
    </p>
   </t:if>
   <t:if t:test = "hasSpecialInstructions">
    <h4 class = "locationInfo_instructions">Other Information</h4>
    <p>
     <t:outputRaw value = "specialInstructions" />
    </p>
   </t:if>
  </div>
 </div>
 <t:if t:test = "site.hasCoordinates">
  <span t:type = "ui/GoogleDirections" sites = "mapSites" showDirections =
 "true" collapseLocationMap = "collapseLocationMap"
   showLocationMap = "true" />
 </t:if>

 <div id = "gmapCanvas"></div>
</t:container>
```

Timeline Event Detail

Filename TimelineEventDetail.tml

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <dl class = "details">
  <dt>Course</dt>
  <dd>${record.courseClass.course.name}</dd>
  <dt>Class code</dt>
  <dd>${record.courseClass.uniqueIdentifier}</dd>
```

```
  <dt>Map &amp;amp; Address</dt>
  <t:if test = "${record.room}">
   <dd>
    <span t:type = "ui/RoomLocation" room = "record.room" withRoomName = "true" />
   </dd>
   <p:else>
    <dd></dd>
   </p:else>
  </t:if>
  <dt>Start</dt>
  <dd>
   <t:output value = "record.startDate" format = "timestampFormatter" />
  </dd>
  <dt>End</dt>
  <dd>
   <t:output value = "record.endDate" format = "timestampFormatter" />
  </dd>
 </dl>
</t:container>
```

Timetable Events

Filename TimetableEvents.tml

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd">
 <table class = "${cssTableClass}">
  <thead>
   <tr>
    <t:loop source = "headerLabels" value = "label">
     <th>${label}</th>
    </t:loop>
   </tr>
  </thead>
  <t:loop source = "displayedObjects" value = "event" index = "listIndex">
   <tr class = "${cssRowClass}">
    <td>
     <t:output value = "event.startDate" format = "itemDateFormatter" />
    </td>
    <td>
     <t:output value = "event.startDate" format = "itemTimeFormatter" />
     <t:if test = "hasItemEndDate"> - <t:output value = "event.endDate" format =
 "itemTimeFormatterWithTimeZone" />
     </t:if>
    </td>
    <td>
     <t:if t:test = "event.room">
      <span t:type = "ui/RoomLocation" room = "event.room" withRoomName = "true" />
     </t:if>
    </td>
                <td>
                    <t:if t:test = "event.publicNotes">
      <t:outputraw value = "${detail}" />
                    </t:if>
                </td>
   </tr>
  </t:loop>
 </table>
</t:container>
```

Social Media

Filename SocialMedia.tml

Allows the user to order the preferred buttons for the Add This social media links which are placed against course and static pages.

```
<t:container xmlns:t = "http://tapestry.apache.org/schema/tapestry_5_1_0.xsd" xmlns:p =
 "tapestry:parameter">
 <div class = "addthis_toolbox addthis_default_style ">
  <a class = "addthis_button_email"></a>
  <a class = "addthis_button_preferred_1"></a>
  <a class = "addthis_button_preferred_2"></a>
  <a class = "addthis_button_preferred_3"></a>
  <a class = "addthis_button_compact"></a>
 </div>
 <script type = "text/javascript" src = "//s7.addthis.com/js/300/addthis_widget.js#pubid=
${addThisProfileId}"></script>
</t:container>
```