Understanding Page Template Components

2016 OmniUpdate User Training Conference Activity Guide

OmniUpdate, Inc.
1320 Flynn Road, Suite 100
Camarillo, CA 93012

Contents

Understanding Page Template Components

To get started, first log into your workshop site: workshop[#].outc16.com

1. Navigate to the homepage by using the personalized link on your workshop insert
2. Replace **[#]** with the number on your workshop insert.
3. Click on the **DirectEdit** link found on the page, which for the conference is the words "Last Updated" followed by a date.
4. Log into OU Campus using the administrator username and password.

Converting a PCF into a TMPL

Understanding the PCF

Purpose: To understand the key components of a PCF before converting it into a TMPL.

Objective: Identify the prolog, editable regions, and page parameters in the PCF.

1. Navigate to the */workshop-understanding-page-templates* directory.
2. **Edit** the *starter.pcf* file via the **Source Editor**.
3. Note the location of the two `<ouc:properties>` nodes and the `<parameter>` nodes inside them.
4. Note the location of the `<ouc:div>` node with `label="maincontent"`, as well as the `<ouc:editor />` node within the `<ouc:div>`.
5. Read the existing prolog elements and navigate to the directory that contains the declared XSL file.
6. Navigate back to *starter.pcf*
7. Add an additional prolog statement on the line below the existing `<?pcf-stylesheet>` declaration:
   a. `<?pcf-stylesheet path="/_resources/xsl/mobile-preview.xsl" title="Mobile Preview" extension=".m.html" alternate="yes" publish="no" ?>`
8. **Save** the changes.
9. **Preview** the changes by switching from **Interior Page** to **Mobile Preview** in the drop-down menu found on the **Preview** screen.

Saving as a TMPL and Removing Content

Purpose: To understand how to turn a single webpage into a new template file.

Objective: Remove page-specific content from the PCF file and turn it into a TMPL file.

1. While still viewing *starter.pcf* in the */workshop-understanding-page-templates* directory, enter the **Source Editor** once more.
2. Click **Save As** in the Source Editor toolbar.
3. In the resulting filechooser, navigate to */_resources/ou/templates/workshop-understanding-page-templates* and save the file with the name **newtemplate.tmpl**
4. Delete the content inside the `<parameter>` node with the `name` of `heading`.
5. Delete the two `<option>` nodes within the `<parameter>` with the `name` of `left-enable`.

6. Delete the two `<option>` nodes within the `<parameter>` with the `name` of `right-enable`.
7. Delete the content inside of the `<title>` node (found in `<ouc:properties label="metadata">`).
8. Inside `<meta name="Description">`, delete the string of text inside the `content` attribute.
9. Finally, delete all content inside the `<ouc:div>` node - keep the `<ouc:editor>` node, though!
10. **Save** the TMPL file.

Creating the TCF

Create a TCF from a Starter File

Purpose: To understand the basic structure of TCF files.

Objective: Create a new TCF for a page.

1. Navigate to the */_resources/ou/templates/workshop-understanding-page-templates* directory.
2. Locate the *newtemplate.tcf* file (alternatively, download the helper files from the OUTC16 website and upload *newtemplate.tcf* into this directory).
3. **Edit** *newtemplate.tcf* via the **Source Editor**.
4. Inside the `<title>` node, add a name for the template.
5. Between the opening and closing `<variable-list>` nodes, add a `<variable>` node for the page title
   a. The `name` attribute should be `name="title"`, the `type` attribute should be `type="text"`, and then there should also be a `prompt` and an `alt` attribute (the values of which can be whatever you want).
6. Download the helper files from the OUTC16 website if you haven't already and open the *variable-nodes.txt* file. Alternatively, the file can be found inside the */workshop-understanding-page-templates* directory.
7. Paste the remaining variables from *variable-nodes.txt* before the closing `<variable-list>` tag. Alternatively, for a greater challenge, add the following variables manually, with the correct attributes (refer to the Support Site or *end-result.tcf* from the helper files for more information):
   a. `<variable name="description">`
   b. `<variable name="body">`
   c. `<variable name="leftcol">` with two `<option>` nodes inside it with `value` attributes of `1` and `0`, respectively
   d. `<variable name="rightcol">` with two `<option>` nodes inside it with `value` attributes of `1` and `0`, respectively
   e. `<variable name="pcf-filename">`
8. Between the opening and closing `<template-list>` tags, configure the template creation for creating a new page using the *newtemplate.tmpl* file. Refer to *end-result.tcf* if you need help.

9. Note the information present in the `<navigation-list>` node and the `<navigation>` nodes within it – this is how links are automatically added to a navigation file upon new page creation.
10. **Save** the changes.

Adding Echo Directives to the TMPL

Purpose: To understand the relationship between TCF and TMPL files.

Objective: Add echo directives to the TMPL that pull information from our TCF variables.

1. Navigate to the */_resources/ou/templates/workshop-understanding-page-templates* directory.
2. Open *newtemplate.tmpl* in the **Source Editor**.
3. Add echo directives in the following locations:
   a. Inside `<parameter name="heading">`, add `<!--%echo var="title" -->`.
   b. Inside `<parameter name="left-enable">`, add `<!--%echo var="leftcol" encoding="none" -->`.
   c. Inside `<parameter name="right-enable">`, add `<!--%echo var="rightcol" encoding="none" -->`.
   d. Inside `<title>` (which resides in `<ouc:properties label="metadata">`), add `<!--%echo var="title" -->`.
   e. Add `<!--%echo var="description" -->` inside the content attribute of the `<meta name="Description">` tag.
   f. Inside the `<ouc:div label="maincontent">` node and after the `<ouc:editor>` node, add `<!--%echo var="body" encoding="none" -->`.
4. **Save** the file and return to the */_resources/ou/templates/workshop-understanding-page-templates* directory.

Final Template Creation Steps

Adding an Image

Purpose: To understand which files are necessary to create a template in OU Campus.

Objective: Upload an image to */_resources/ou/templates/workshop-understanding-page-templates* with the appropriate filename.

1. If not there already, navigate to */_resources/ou/templates/workshop-understanding-page-templates*.
2. Find an image with file extension JPEG, PNG, or GIF and **Upload** it to the directory.
3. Name it *newtemplate.gif* (or *.jpg*, *.png*, etc., depending on file type)
   a. The image name must have the same filename as the TCF being used. If you have named your TCF/TMPL something other than *newtemplate*, name your image accordingly.
4. If desired, **Publish** the file.

Setting Access Settings for the Template

Purpose: To prepare the template for use in OU Campus.

Objective: Configure access settings for the new template and assign it to a template group.

1. In a new tab, navigate to **Setup > Templates**.
2. The new template that you created following this guide should appear in the list of templates shown on this screen. Its name will be the string of text in the `<title>` node of the TCF.
3. If desired, change the template **Title**, alter the **Access Group** for the template, or override the default image by pasting a URL in the **Thumbnail URL** field.
   a. Click **Save** if any of these settings have been changed.
4. Navigate to the **Template Groups** screen and create a new template group.
5. Name the new template group and click the checkbox next to your newly-created template.
6. Click **Save**.

Testing our New Template

Purpose: To check our work and see the new template in action.

Objective: Assign the template group to a directory and create a new page using the template.

1. In the Pages list view, hover over a directory and click **Edit > Access**.
2. In the **Access Settings** dialog box, scroll to **Template Group** and choose your newly-created template group from the drop-down menu.
3. **Save** your changes.
4. Click on the directory to open it in the Pages list view.
5. In the directory, click **New**. Your newly-created template should appear in the **New Content** modal.
6. Select your template.
7. Fill out the New Page Wizard for your template, adding a page title, description, sample body text, choosing which columns to show, and adding a web-friendly filename.
8. Click **Create**.
9. Your new page has been created. Check to ensure that the content is added into the appropriate areas of the page.

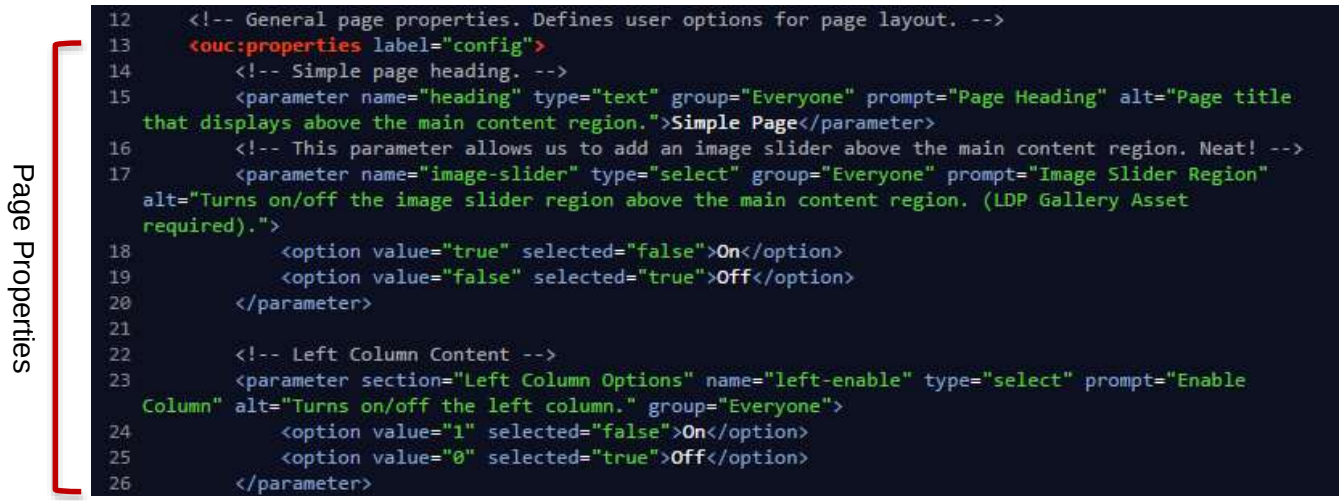
PCF Breakdown

Use this guide to understand the components of a PCF:

Prolog and Root Node

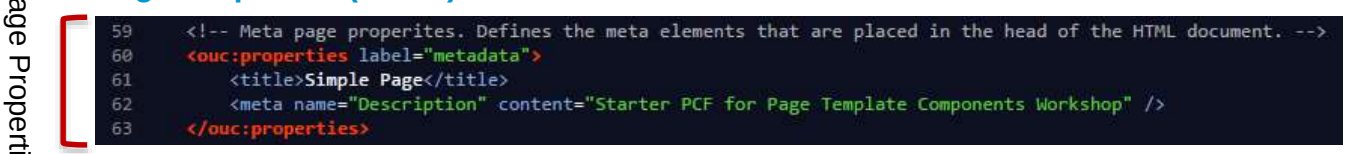
All XML documents require an XML Prolog. For PCFs (which are XML documents), the prolog will also include the PCF stylesheet declaration with the correct path to the specific xsl used to style the document. They also require a root node. For PCF documents, the root node will be `<document>`. All the children nodes within the parent `<document>` node are used to markup page properties, metadata or editable regions for the PCF.

Page Properties (Part 1)



An `<ouc:properties>` node with the `label` of `config` is where custom parameters reside. These parameters can do a variety of things, and generally they're used to alter the appearance of a page. Whether that means configuring a page heading or banner image, turning columns on or off, or even adding additional editable regions, they'll all reside within `<parameter>` nodes inside `<ouc:properties label="config">`.

Page Properties (Part 2)



Another `<ouc:properties>` node with the `label` of `metadata` will contain any metadata for the page. This information is usually not displayed on the page itself, but is used by web

browsers and search engines. Notice that the nodes within `<ouc:properties label="metadata">` aren't the `<parameter>` nodes we saw above. The nodes for metadata have a specific syntax that's unique to them.

Editable Regions

```
65      <!--    Editable region. An ouc:div defines an editable region so a user can edit content within the WYSIWYG editor. Every ouc:div must have a unique label. -->
66      <ouc:div label="maincontent" group="Everyone" button-text="Edit Me!">
67          <ouc:editor cssmenu="/_resources/ou/editor/styles.txt"/>
68          <p><img class="" style="float: left; margin-right: 5px;" title="What a hipster!" src="http://demos.creative-tim.com/hipster-as-f-cards/assets/img/beard-1.jpg" alt="Bearded Guy" width="400" height="317" />VHS biodiesel before they sold out sartorial. Food truck scenester kogi next level. Trust fund godard scenester bicycle rights. Pork belly keffiyeh echo park VHS. Cronut poutine flannel dreamcatcher keytar, synth mustache craft beer pug lomo tacos cardigan. Semiotics selfies kinfolk lo-fi +1 craft beer, mumblecore four loko banh mi. Leggings cardigan selfies, typewriter bushwick hoodie tote bag brunch.</p>
69          <p>You probably haven't heard of them PBR&amp;amp;B mustache, helvetica DIY yr semiotics freegan. Cold-pressed DIY letterpress, chillwave twee four loko cliche neutra tattooed chicharrones sustainable PBR&amp;amp;B four dollar toast. Asymmetrical shabby chic literally bushwick waistcoat four dollar toast. Raw denim lo-fi retro fingerstache messenger bag. YOLO bushwick +1 pug gastropub trust fund. Tumblr venmo fixie wayfarers meggings keytar, squid marfa tofu ennui chillwave cray. Austin vinyl irony, cold-pressed literally tumblr trust fund plaid polaroid kombucha chia church-key fap everyday carry.</p>
70          <p>Sriracha marfa affogato shabby chic, pitchfork tote bag chia street art. Etsy before they sold out quinoa, hella vinyl butcher flexitarian hashtag tofu banjo bespoke schlitz master cleanse craft beer. Seitan tattooed chambray waistcoat, forage DIY whatever blue bottle austin lomo synth slow-carb literally. Skateboard cornhole direct trade, trust fund tote bag knausgaard lumbersexual godard keffiyeh biodiesel freegan letterpress bushwick mustache. Viral schlitz pork belly chicharrones tattooed affogato kickstarter. Wolf 90's godard fixie, hammock hoodie cliche knausgaard kinfolk. Pabst pop-up master cleanse lomo +1 chartreuse.</p>
71          <p>&amp;nbsp;</p>
72      </ouc:div>
```

Editable Region

An `<ouc:div>` node denotes an editable region on a PCF page, where a user can edit content using JustEdit. Every `<ouc:div>` must have its own unique `label` attribute.

The self-closing `<ouc:editor>` node that you see directly under the opening `ouc:div` tag is where you can define custom CSS styles that can be applied to content in this region. This is done using the `cssmenu` attribute.

Extra Regions and Closing Document Node

```
77      <!-- Extra regions that will show up when we have the appropriate column turned on. -->
78      <ouc:div label="left-quicklinks" group="Everyone" button-text="Quick Links">
79          <h4>Quick Links</h4>
80          <ul>
81              <li><a href="#">Link 1</a></li>
82              <li><a href="#">Link 2</a></li>
83              <li><a href="#">Link 3</a></li>
84          </ul>
85      </ouc:div>
86
87      <ouc:div label="right-quicklinks" group="Everyone" button-text="Quick Links">
88          <h4>Quick Links</h4>
89          <ul>
90              <li><a href="#">Link 1</a></li>
91              <li><a href="#">Link 2</a></li>
92              <li><a href="#">Link 3</a></li>
93          </ul>
94      </ouc:div>
95
96  </document>
97
```

Editable Regions

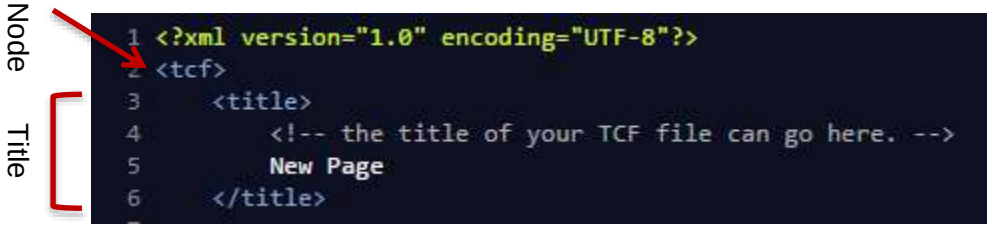
Closing Root Node

Some pages may have additional, secondary editable regions, each denoted by another `<ouc:div>` node with a unique `label` attribute. Finally, when all else has been defined, the PCF will end with the closing root node of `</document>`.

TCF Breakdown

Use this guide to understand the components of a TCF.

Prolog, Root Node, and Title



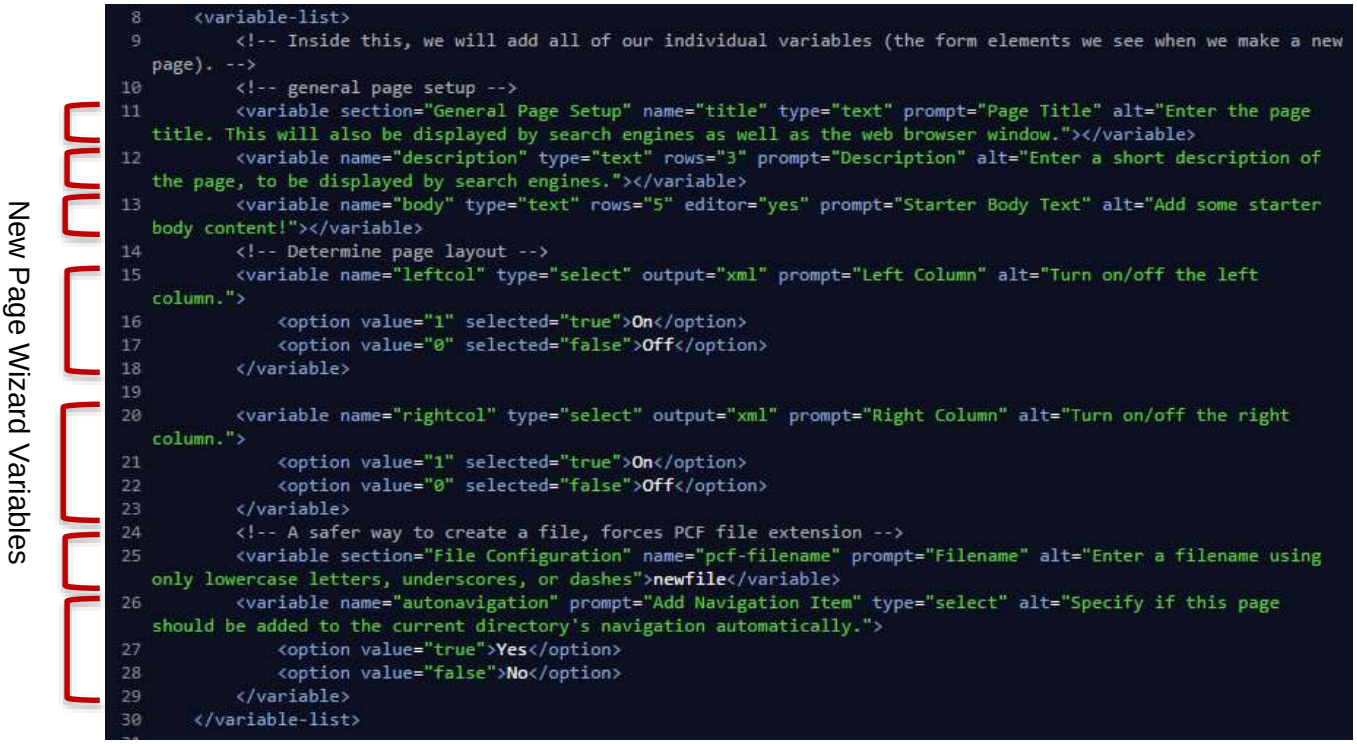

```
<?xml version="1.0" encoding="UTF-8"?>
<tcf>
    <title>
        <!-- the title of your TCF file can go here. -->
        New Page
    </title>
```

Just like PCFs, TCFs are XML files at heart, and thus need a prolog. Since PCF stylesheets are unnecessary for a TCF, all that needs to be added is an XML declaration providing the version number and encoding type.

The root node for a TCF is simply `<tcf>`. Every other node inside a TCF must live inside of it.

The `<title>` node is the name of your new template. It will show up in the **Template Options** screen found in **Setup > Templates**.

Variable Nodes




```
    <variable-list>
        <!-- Inside this, we will add all of our individual variables (the form elements we see when we make a new page). -->
        <!-- general page setup -->
        <variable section="General Page Setup" name="title" type="text" prompt="Page Title" alt="Enter the page title. This will also be displayed by search engines as well as the web browser window."></variable>
        <variable name="description" type="text" rows="3" prompt="Description" alt="Enter a short description of the page, to be displayed by search engines."></variable>
        <variable name="body" type="text" rows="5" editor="yes" prompt="Starter Body Text" alt="Add some starter body content!"></variable>
        <!-- Determine page layout -->
        <variable name="leftcol" type="select" output="xml" prompt="Left Column" alt="Turn on/off the left column.">
            <option value="1" selected="true">On</option>
            <option value="0" selected="false">Off</option>
        </variable>

        <variable name="rightcol" type="select" output="xml" prompt="Right Column" alt="Turn on/off the right column.">
            <option value="1" selected="true">On</option>
            <option value="0" selected="false">Off</option>
        </variable>
        <!-- A safer way to create a file, forces PCF file extension -->
        <variable section="File Configuration" name="pcf-filename" prompt="Filename" alt="Enter a filename using only lowercase letters, underscores, or dashes">newfile</variable>
        <variable name="autonavigation" prompt="Add Navigation Item" type="select" alt="Specify if this page should be added to the current directory's navigation automatically.">
            <option value="true">Yes</option>
            <option value="false">No</option>
        </variable>
    </variable-list>
```

The `<variable-list>` node in the TCF is the “container” for `<variable>` nodes. These `<variable>` nodes are what make up the fields in the New Page Wizard when a user creates a new page from a template. Each `<variable>` must have a unique `name` attribute.

Template Nodes

Template

```
32    <template-list>
33        <!-- Using a template node, this is where we tell the TCF which TMPL file to send the data to. -->
34        <template
35        filename="{pcf-filename}"
36        display-filename="no"
37        extension="pcf"
38        preferred-redirect="yes"
39        autonav="{autonavigation}">end-result.tmpl</template>
40    </template-list>
```

In order for OU Campus to know where to send all the data in the TCF's `<variable>` nodes, we must define which TMPL file (or files) is associated with the TCF. This is where `<template-list>` comes into play. Inside `<template-list>`, users can add `<template>` nodes to "link" the TCF with various TMPL files – one `<template>` node for each TMPL file. That way, the echo directives in the TMPL will look for `<variable>` nodes with the defined `name` attributes in our TCF.

Navigation and Closing Root Node

Auto-navigation Options

Closing Root Node

```
42    <navigation-list>
43        <!-- This is the code that adds our new page to the side navigation automatically. It's already good to
   go. -->
44        <navigation name="true" path="_sidenav.inc" group="Everyone" publish="no"><li><a href="
   {ox_autonav:shorturl}">{title}</a></li></navigation>
45        <navigation name="false" path="_sidenav.inc" group="Everyone" publish="no"></navigation>
46    </navigation-list>
47 </tcf>
```

The final component to a standard New Page TCF is the `<navigation-list>` node, which deals with auto-navigation. Inside `<navigation-list>`, we add `<navigation>` nodes to define where we want links to be automatically created when we make a new page. Here, I have one `<navigation>` node telling OU Campus to create a link automatically in the *_sidenav.inc* file that "lives" in the same directory as where I'm making my new page. If I don't want a link to be auto-added, the second `<navigation>` node adds nothing to the *_sidenav.inc* file.

Immediately underneath `<navigation-list>`, we see the closing root node `</tcf>`, which signifies the end of our TCF.