Multiple profile types for foaf:Person

Introduction

VIVO now supports multiple profile pages for foaf:Persons. This feature, which is optional so installations can continue to use just the individual--foaf-person.ftl template, currently consists of two profile page types: a standard view, which is a redesigned version of the foaf:Person template in previous releases; and a quick view, which emphasizes the individual's own web page presence while providing summary VIVO information, such as current positions and research areas. The profile quick view requires the use of a web service that captures images of web pages. This web service is not included with the VIVO software, so an installation will either have to develop their own service or use a third-party service, usually for a small fee depending on the number of images served. Examples of these services include WebShotsPro, Thumbalizr and Websnapr.
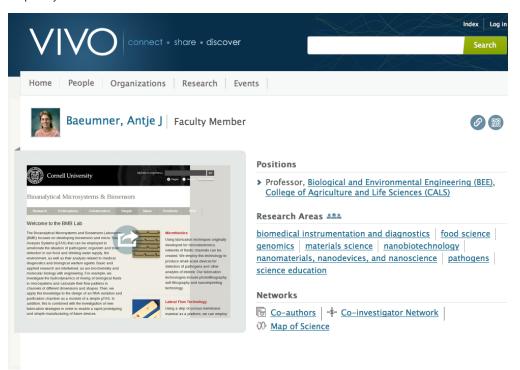
The Profile Page Types

As noted above, there are currently two supported profile page types. Here are examples of those two views
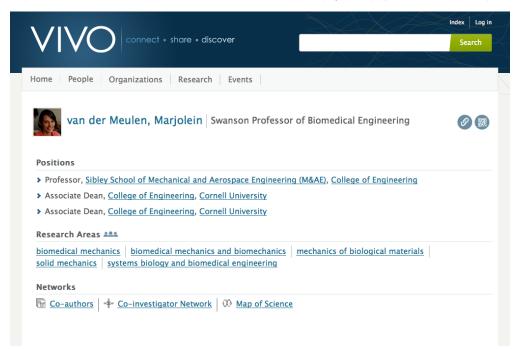
The Standard View

The standard view is similar to the default foaf:Person template except that the information displayed at the top of the page is divided into only two primary columns instead of three. The actual template name for this page type is individual--foaf-person-2column.ftl.

The Quick View

As illustrated below, the quick view puts a visual emphasis on the individual's own web presence. In this case, the person only has one web page displayed. When there is more than one, the primary web page is displayed as shown and any additional web pages are displayed as thumbnails beneath the primary one.

It's possible that there will be some individuals who do not have a web page to display. In that situation the quick view will display as follows.

Implementing Multiple Profile Pages

Here are the steps required to implement the multiple profile pages feature.

1. Develop or a website image capture service
2. Update the runtime.properties file
3. Override the default foaf:Person template
4. Update the webpage quick view template
5. Set the Profile Page Type for your foaf:Persons

Step 1. Develop or a Website Image Capture Service

Since there are currently only two page views, and one of those emphasizes the individual's own web site, to implement the multiple profile pages feature requires that an installation either develop its own web service for capturing images of web sites or select a third-party service for this purpose. As noted in the introduction, these services include WebShotsPro, Thumbalizr and Websnapr.

A third option, however, would be to modify the quick view template (individual--foaf-person-quickview.ftl) so that it does not display a web page image (as in the third screen shot above). This template file is located in the `productMods/templates/freemarker/body/individual` directory.

Step 2. Update the runtime.properties File

Set the multiViews.profilePageTypes to "enabled" and ensure that it is not commented out.

Step 3. Override the Default foaf:Person Template

There are two ways to override the default individual--foaf-person.ftl template, which is located in the `themes/wilma/templates` directory: (1) rename the file, or (2) remove it from that directory.
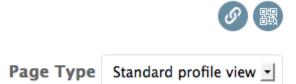
Step 4. Update the Webpage Quick View Template

The template that displays the web page image in the quick view is named propStatement-webpage-quickview.ftl. As delivered, this template uses a placeholder link (or links) to display the individual's web page (or pages), while the code that calls the web service is currently commented out. Here is that section of the template:

```
40  <#-- This section commented out until the web service for the web page snapshot is implemented. -->
41  <#-- The assumption is made that the service will require the url of the web page and possibly   -->
42  <#-- an image size as well. Delete the placeholder link once the web service is implemented.     -->
43  <#--
44  <span id="span-${identifier}" class="webpage-indicator-qv">
45      ${strings.loading_website_image}. . .   <img  src="${urls.images}/indicatorWhite.gif">
46  </span>
47  <a title="${i18n().click_to_view_web_page(linkText)}" href="${statement.url}">
48      <img id="img-${identifier}" class="org-webThumbnail" src="http://your.web.service/getsTheImage?url=${statement.url}${imgSize}"
    alt="${i18n().screenshot_of_webpage(statement.url)}" style="display:none"/>
49  </a>
50  <#if imgSize == "" >
51      </li>
52      <li class="weblinkLarge">
53          <a title="${i18n().click_to_view_web_page(linkText)}" href="${statement.url}">
54              <img id="icon-${identifier}" src="${urls.images}/individual/weblinkIconLarge.png"  alt="${i18n().click_webpage_icon}"
    style="display:none"/>
55          </a>
56  <#else>
57      </li>
58      <li class="weblinkSmall">
59          <a title="${i18n().click_to_view_web_page(linkText)}" href="${statement.url}">
60              <img id="icon-${identifier}" src="${urls.images}/individual/weblinkIconSmall.png"  alt="${i18n().click_webpage_icon}"
    style="display:none"/>
61          </a>
62  </#if>
63  -->
64  <#-- Here is the placeholder link, 4 lines long -->
65      <a href="${statement.url}" title="${i18n().link_text}">
66          ${linkText}
67      </a>
68      <script>$("a[title='${i18n().link_text}']").parent('li').css("float","none");</script>
69  <#else>
70      <a href="${profileUrl(statement.uri("link"))}" title="${i18n().link_name}">${statement.linkName}</a> (${i18n().no_url_provided})
71  </#if>
72
```

Note the highlighted text on line 48. The URL in the src attribute is where you call either the web service you developed or the third-party service. The APIs for these services are fairly standard. Besides the URL of the web site that will be the source of the screen shot, the code in this template assumes that the API also takes an image size. For example, some services can provide small, medium and large images; others may only provide a large image and a thumbnail image. Once you've updated line 48 to call your web service, remember to comment out or remove the placeholder link, lines 65-68.

Step 5. Set the Profile Page Type for your foaf:Persons

When multiple profile pages are implemented, the default view is the standard profile view. You can change an individual's profile page type through the GUI by accessing the Page Type drop down:

Page Type Standard profile view ▾

You can also set the profile page type by ingesting RDF. An N3 triple, for example would consist of the following parts:

- the subject would be the URI of the individual, such as

    <http://localhost:8080/individual/n7829>;

- the predicate would be the hasDefaultProfilePageType object property,

    <http://vitro.mannlib.cornell.edu/ontologies/display/1.1#hasDefaultProfilePageType>;

- and the object would be the type of profile,

    <http://vitro.mannlib.cornell.edu/ontologies/display/1.1#quickView> (or #standard).

The ProfilePageType class is defined in the display model. Refer to the profilePageType.n3 file for details.

Using the Standard View Without Implementing Multiple Profile Pages

It's possible that an installation may want to use the standard view instead of the default foaf:Person template, but does not want to implement multiple profile pages. This can be done by simply (1) overriding the default foaf:Person template (just as in Step 3 above) and (2) ensuring that the multiViews. profilePageTypes properties in the runtime.properties file is either commented out or set to "disabled."