# Creating a custom theme



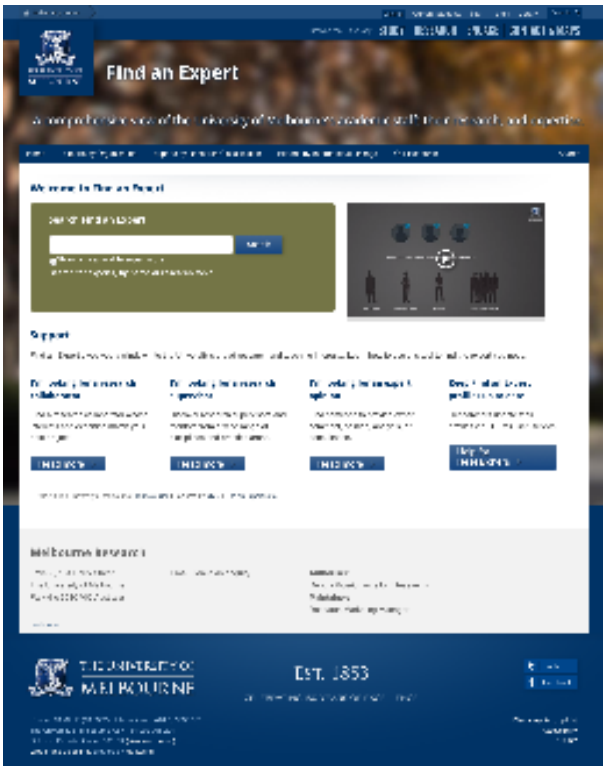
Create your own "brand" for VIVO. Change colors, logo, headings, footers, and more.

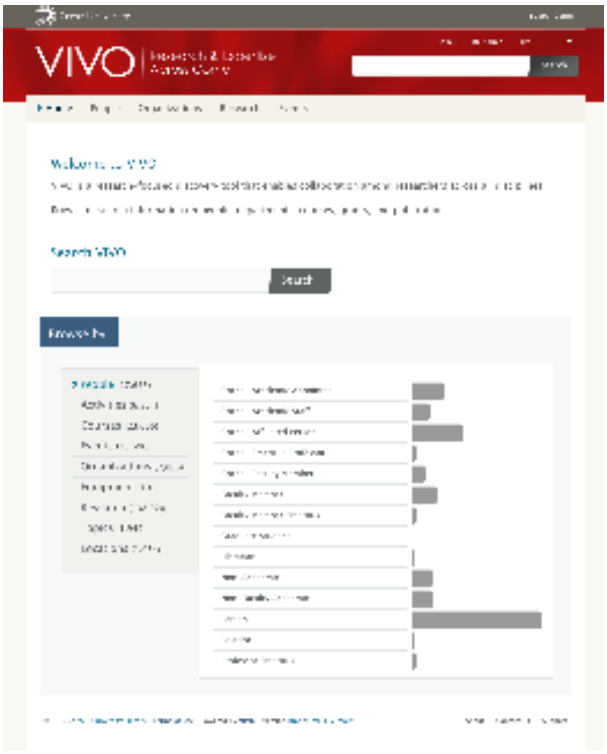
## Overview

### What can it do for you?

Change the "look and feel" of your VIVO installation. Change the styling, the images, the layout, the text, and more. Modify the header and footer on all pages.

### Before and After

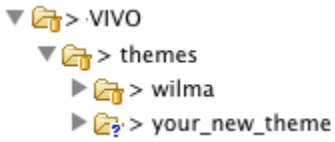




## What do you need to know?

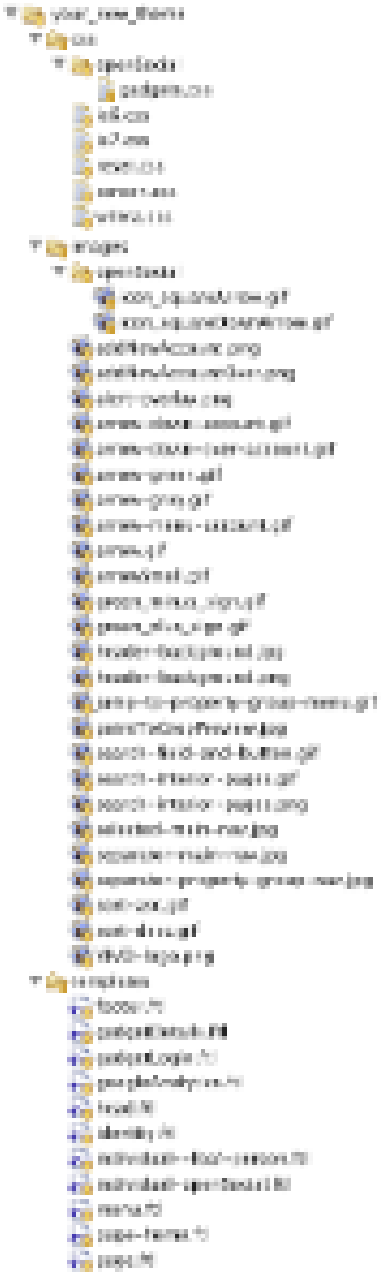
- Standard web-site technologies: HTML, CSS and maybe JavaScript.
- Something about the Freemarker template engine.
- Where the theme files are stored in VIVO, and how to reference them.

## Getting started

VIVO comes with a standard theme, called Wilma. Copy the theme into a new directory in the VIVO source tree, and give it a name.

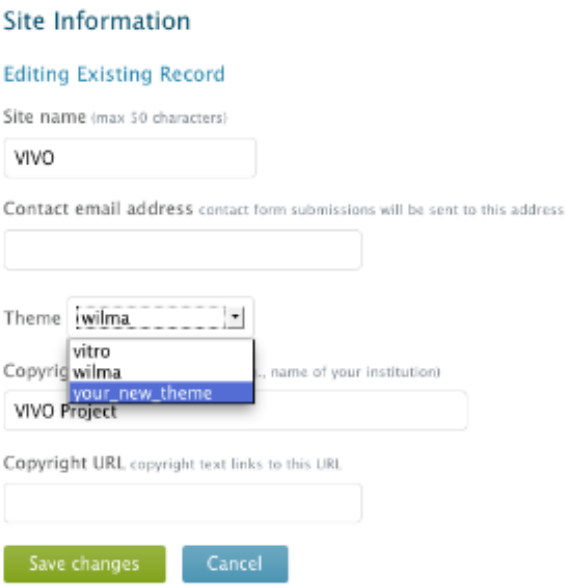



Your new theme will contain CSS files, image files, and Freemarker templates.



This image is from VIVO release 1.5.1 – the exact contents of your theme may be different.

Run the Ant build to deploy your new theme to the Tomcat container. Restart the VIVO Tomcat process. You can then go to the ***Site Admin*** page and choose ***Site Information***, to select your theme as the current one.




# The structure of pages in VIVO

The pages in VIVO are built around three different frameworks. Each of these uses the same header and footer, to provide consistency. In addition to including the header and footer, the pages frequently include smaller templates to provide detail.

These are the basic frameworks:

| | |
|---|---|
| The home page | As the point of entry for VIVO, the home page is special. It is based on the Freemarker template `page-home.ftl` |
| All other public pages | Based on the Freemarker template `page.ftl` |
| "back-end" pages | Pages used for editing the ontology, or manipulating the raw data of VIVO are based on a JSP named `basicPage.jsp` |

# Some significant templates

**page.ftl**

`page.ftl` is the default base template. The rest of the theme templates listed are components of `page.ftl` (included either directly or indirectly). Closer inspection of `page.ftl` reveals a stripped down file that declares minimal markup itself and instead reads as a list of includes for the component templates.

| |
|---|
| There are still page in VIVO which are not constructed entirely in FreeMarker. Breaking down the base template into components allows consistent markup to be used for any page rendered by the application, no matter how the page is constructed. |
| On the VIVO home page, `page-home.ftl` is used instead of `page.ftl`. It serves much the same purpose, but allows you to create a different layout for your home page than for the other pages in VIVO. |

Once the transition is complete in a future VIVO release and all pages are rendered entirely using FreeMarker templates, the restrictions on `page.ftl` will be lifted and the preference on whether the base template should be broken down into smaller component templates will be left to the theme developer. Until that time, it is critical that the following components be maintained:

**head.ftl**

This component template is responsible for everything within the `<head>` element. Note that the open and closing tags for the `<head>` element are defined in `page.ftl` and wrap the include for `head.ftl`. There are several includes within `head.ftl` that should be carried over to any new theme to maintain expected functionality:

- `<#include "stylesheets.ftl">` - ensures that the necessary stylesheets called by templates downstream will be added to the page via <link> elements
- `<#include "headscripts.ftl">` - ensures that the scripts called by templates which must be in the <head> will be added to the page via <script> elements

**identity.ftl**

This component template is responsible for rendering the VIVO logo, secondary navigation and search input field at the top of the page. There are no mandatory includes from `identity.ftl` that need to be carried over but there are 2 template variables that are of particular interest (`${user}` and `${urls}`).

**menu.ftl**

This component template is responsible for rendering the primary navigational menu for the site. In wilma, it also happens to declare the open tag for the main content container. There are no mandatory includes from `menu.ftl`. The `${menu}` template variable is crucial since it contains an array of menu items needed to build the primary navigational menu.

**footer.ftl**

This component template is responsible for rendering the copyright notice, revision information, secondary navigation, and link for the contact form. There is a single include that should be maintained:

- `<#include "scripts.ftl">` - ensures that the non head scripts (those that don't need to be placed in the <head>) called by the templates will be added to the page via `<script>` elements

Several template variables of interest include `${copyright}`, `${user}`, and `${version}`.

**googleAnalytics.ftl**

This component template is primarily intended for the 7 partner institutions on the NIH grant, but it is available for anyone who is interested in using Google Analytics to track visits to a VIVO installation. It is included by `footer.ftl`. Simply uncomment the `<script>` element and provide your Google Analytics Tracking Code.

Adjust the markup as necessary in `page.ftl` and these component templates to achieve the desired content structure, and modify the stylesheets to meet layout needs and style your site. Remember that changes should be made in the source directory and that you will need to redeploy the project before the changes are reflected in the live website.

For more information about VIVO and web analytics, see VIVO Web Analytics.

You can find more information about the structure of the VIVO theme in How VIVO creates a page.

# Making changes

## Modify files in the theme

You can edit the Freemarker templates and the CSS files in the theme with any text editor. You can replace the image files with images that you choose.

## Add files to the theme

### Add CSS, JavaScript, or image files

As you modify the templates, you may want to use additional images, CSS files, or JavaScript files. When your templates refer to these files, they will use the Freemarker variable `urls.theme`, as shown in these examples:

```
<!-- an image file -->
<img src="${urls.theme}/images/arrow-green.gif"/>

<!-- a CSS file -->
<link rel="stylesheet" href="${urls.theme}/css/screen.css" />

<!-- a JavaScript file (create a js directory in your theme) -->
<script type="text/javascript" src="${urls.theme}/js/my.js"></script>
```

### Add Freemarker templates

If your modifications use new Freemarker templates, you can refer to them more simply. Freemarker already knows where your theme directory is located.

```
<#include "my-new-template.ftl">
```

## Override files that are not in the theme directory

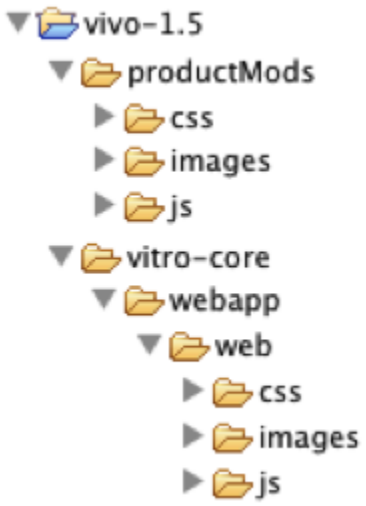
In order to keep the theme directory uncluttered, VIVO keeps most of the front-end files in a separate location. Changes to the theme usually involve the files in the theme directory, but you can override other files as well.

### Override CSS, JavaScript or image files that are not in the theme directory

You may notice that templates refer to files that are not in the theme directory. They use references based on the Freemarker variable `urls.base` instead of `urls.theme`, like this:

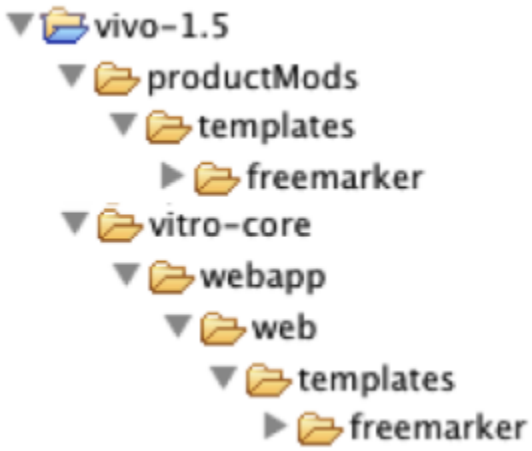
```
<!-- an image file -->
<img src="${urls.base}/images/arrowIcon.gif"/>

<!-- a CSS file -->
<link rel="stylesheet" href="${urls.base}/css/login.css" />

<!-- a JavaScript file -->
<script type="text/javascript" src="${urls.base}/js/browserUtils.js"></script>
```

These refer to files in the `vitro-core/webapp/web` directory, which you can override in the `vivo/productMods` directory. If you look, you will see that `vivo/productMods` already contains some files that override those in `vitro/webapp/web`. This is because VIVO itself is a customization of Vitro.




### Override Freemarker templates that are not in the theme directory

Like adding templates, overriding templates is simplified. You can override a file in `vitro-core/webapp/web/templates/freemarker`, or one of its sub-directories, by creating a file with the same name and path under `vivo/productMods/templates/freemarker`.




But VIVO treats all available Freemarker templates as belonging to the same flat namespace, whether they are in the theme directory or in the `templates/freemarker` directory, or one of its sub-directories. So a file named `vitro-core/webapp/web/templates/freemarker/page/partials/footer.ftl` can be overridden by a file called `footer.ftl` in the theme directory.

And it is.

## Working on the theme

When you make changes to VIVO, you should make the changes in your VIVO distribution directory, run the build script, restart Tomcat, and test the changes. If you are doing full customizing of VIVO, this cycle might be best. If you are only working on the theme, you can speed things up.

- Tell the build script to skip the unit tests: they don't test the theme.
  - `ant deploy -Dskiptests=true`
- Don't restart Tomcat.
  - VIVO always serves the most recent version of CSS files, image files, and JavaScript files. You don't need to restart Tomcat to make that happen.
  - However, your browser may cache these files so you won't see the most recent version. Here are some suggestions for bypassing your browser cache.
- Tell VIVO to reload Freemarker templates each time they are requested.
  - By default, VIVO will wait one minute before checking for a change in a Freemarker template.
  - In VIVO 1.5.2 or before, add this to deploy.properties: `Environment.build=development`
  - In VIVO 1.6, add this to build.properties: `developer.defeatFreemarkerCache=true`
  - In VIVO 1.6.1 or later, use The Developer Panel to defeat the Freemarker cache.
  - This is not a good idea for your production VIVO instance - only for developing your theme.

Some developers prefer to make theme changes inside the `tomcat/webapp/vivo` directory. This eliminates the need to run the build script, but opens the threat of having the changes over-written the next time the build script runs.

## When to restart Tomcat

If you make changes to any of the `runtime.properties` or to any of the RDF files in your VIVO home directory, you must restart Tomcat in order to see the effect of the changes.

If you make changes to any of the Java code, you must run the build script and restart Tomcat.

If you make changes to any of the source files in the theme, including images, CSS, JavaScript or Freemarker templates, you must run the build script, but you do not need to restart Tomcat.