$LOGOIMAGE

Customizing the Trac Interface

Introduction

This page is meant to give users suggestions on how they can customize the look of Trac. Topics on this page cover editing the HTML templates and CSS files, but not the program code itself. The topics are intended to show users how they can modify the look of Trac to meet their specific needs. Suggestions for changes to Trac's interface applicable to all users should be filed as tickets, not listed on this page.

Project Logo and Icon

The easiest parts of the Trac interface to customize are the logo and the site icon. Both of these can be configured with settings in trac.ini.

The logo or icon image should be put in a folder named "htdocs" in your project's environment folder. (*Note: in projects created with a Trac version prior to 0.9 you will need to create this folder*)

> *Note: you can actually put the logo and icon anywhere on your server (as long as it's accessible through the web server), and use their absolute or server-relative URLs in the configuration.*

Now configure the appropriate section of your trac.ini:

Logo

Change the `src` setting to `site/` followed by the name of your image file. The `width` and `height` settings should be modified to match your image's dimensions (the Trac chrome handler uses "`site/`" for files within the project directory `htdocs` and "`common/`" for the common ones).

```
[header_logo]
src = site/my_logo.gif
alt = My Project
width = 300
height = 100
```

Icon

Icons should be a 16x16 image in `.gif` or `.ico` format. Change the `icon` setting to `site/` followed by the name of your icon file. Icons will typically be displayed by your web browser next to the site's URL and in the `Bookmarks` menu.

```
[project]
icon = site/my_icon.ico
```

Note though that this icon is ignored by Internet Explorer, which only accepts a file named favicon.ico at the root of the host. To make the project icon work in both IE and other browsers, you can store the icon in the document root of the host, and reference it from trac.ini as follows:

```
[project]
icon = /favicon.ico
```

$LOGOIMAGE

Site Header & Footer

In the environment folder for each Trac project there should be a directory called `templates`. This folder contains files `site_header.cs` and `site_footer.cs`. Users can customize their Trac site by adding the required HTML markup to these files. The content of these two files will be placed immediately following the opening `<body>` tag and immediately preceding the closing `</body>` tag of each page in the site, respectively.

These files may contain static HTML, though if users desire to have dynamically generated content they can make use of the ClearSilver templating language from within the pages as well. When you need to see what variables are available to the template, append the query string `?hdfdump=1` to the URL of your Trac site. This will display a structured view of the template data.

Site CSS

The primary means to adjust the layout of a Trac site is to add CSS style rules that overlay the default rules. This is best done by editing the `site_css.cs` file in the enviroment's `templates` directory. The content of that template gets inserted into a `<style type="text/css"></style>` element on every HTML page generated by Trac.

While you can add your custom style rules directly to the `site_css.cs` file, it is recommended that you simply reference an external style sheet, thereby enabling browsers to cache the CSS file instead of transmitting the rules with every response.

The following example would import a style sheet located in the `style` root directory of your host:

```
@import url(/style/mytrac.css);
```

You can use a ClearSilver variable to reference a style sheet stored in the project environment's `htdocs` directory:

```
@import url(<?cs var:chrome.href ?>/site/style.css);
```

Project List

You can use a custom ClearSilver? template to display the list of projects if you are using Trac with multiple projects.

The following is the basic template used by Trac to display a list of links to the projects. For projects that could not be loaded it displays an error message. You can use this as a starting point for your own index template.

```
<html>
<head><title>Available Projects</title></head>
<body>
 <h1>Available Projects</h1>
 <ul><?cs
 each:project = projects ?><li><?cs
  if:project.href ?>
   <a href="<?cs var:project.href ?>" title="<?cs var:project.description ?>">
    <?cs var:project.name ?></a><?cs
  else ?>
   <small><?cs var:project.name ?>: <em>Error</em> <br />
   (<?cs var:project.description ?>)</small><?cs
  /if ?>
  </li><?cs
 /each ?>
 </ul>
```

$LOGOIMAGE

```
</body>
</html>
```

Once you've created your custom template you will need to configure the webserver to tell Trac where the template is located:

For FastCGI:

```
FastCgiConfig -initial-env TRAC_ENV_PARENT_DIR=/parent/dir/of/projects \
              -initial-env TRAC_ENV_INDEX_TEMPLATE=/path/to/template
```

For mod_python:

```
PythonOption TracEnvIndexTemplate /path/to/template
```

For CGI:

```
SetEnv TRAC_ENV_INDEX_TEMPLATE /path/to/template
```

Main Templates

It is also possible to use your own modified versions of the Trac ClearSilver templates. Note though that this technique is not recommended because it makes upgrading Trac rather problematic: there are unfortunately several dependencies between the templates and the application code, such as the name of form fields and the structure of the template data, and these are likely to change between different versions of Trac.

If you absolutely need to use modified templates, copy the template files from the default templates directory (usually in found in `$prefix/share/trac/templates`) into the `templates` directory of the project environment. Then modify those copies to get the desired results.

See also TracGuide, TracIni