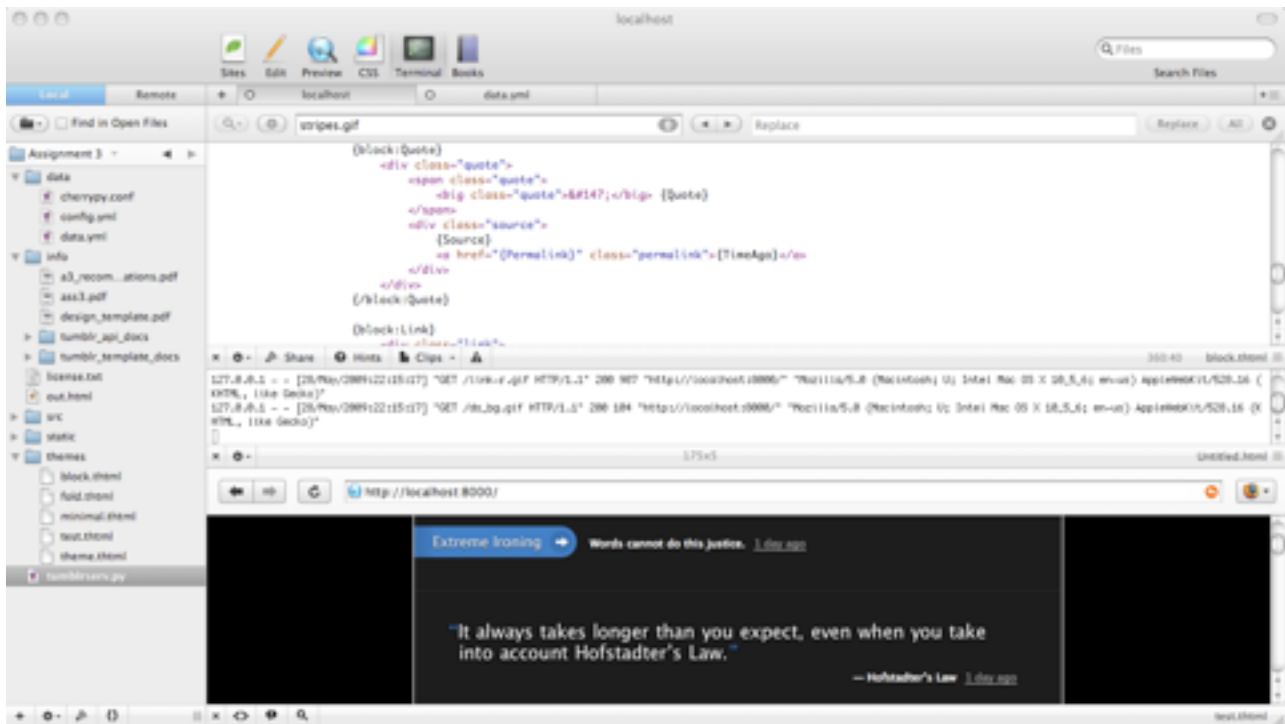


CSSE1001

28/05/2009

Assignment 3 - TumblrServ - Design Document



TumblrServ - Copyright (C) 2009 Jeremy Herbert (UQ Student Number: s42049522)

1. Description

TumblrServ is primarily a tool to help web developers create themes for the popular blogging platform Tumblr (see <http://www.tumblr.com>). Currently, designers must enter their code into a dialog on a webpage, save it and then attempt to reload the page with the new code.

There are many problems with this approach:

- Images must be hosted on a different server, making testing dependent on the configuration of another server
- Developers often use their own unique development process, Tumblr forces you to use theirs
- Updating the code can take as long as 30 seconds; this is far too long to wait in order to receive feedback
- Developers cannot select the data they wish to use in the test environment
- Offline development is impossible

TumblrServ solves all of these problems by running a small server that listens on the local machine. It parses all theme data and performs operations given data from a flat file to generate a page in the same manner as the parsing engine on the Tumblr servers. This allows developers to immediately see the changes as they work in their preferred development environment. Images are also hosted locally in order to keep complexities to a minimum.

2. User Interface/Usage

TumblrServ has no graphical user interface. It is a command line application that is controlled by the switches passed to the application. It must be run from a specific directory structure:

CSSE1001

05/09/2009

Assignment 3 - Design Document

tumblrserver/
tumblrserver.py
data/
src/
static/
themes/

The following table lists the currently available command line switches.

Switch	Description
--theme <theme>	Tells the server to look for a theme named <theme>.html in the themes folder and start a server using it.
--pull-data <username>	Attempts to pull data from the tumblr page associated with <username>
--publish	Uses credentials taken from the configuration file to upload the theme to a tumblr page. Can be used in conjunction with the theme switch
--do-nothing	Render the page as-is with no modifications or parsing

The server will run on a port specified in the CherryPy configuration file located in *config/*. Other configuration is done in *config.yml*; the options available are listed in the following tables.

Parent option	Configuration Option	Type	Description
-	publishing_info	parent	Contains publishing credentials
publishing_info	url	string	The url
publishing_info	username	string	The username used to upload new theme data
publishing_info	password	string	The password used to upload new theme data
-	defaults	parent	contains default-related information
defaults	theme_name	string	The name of the default theme file (no extension)
defaults	data_name	string	Data file name
defaults	rss_url	string	A url to the RSS feed for the tumblr page
defaults	autoreload	parent	Contains options related to the autoreload feature
autoreload	enabled	yes/no	Whether or not the autoreload feature is enabled

CSSE1001

28/05/2009

Assignment 3 - TumblrServ - Design Document

Parent option	Configuration Option	Type	Description
autoreload	interval	integer	The number of seconds between reloads
-	optimisations	parent	Contains optimisation related configuration options
optimisations	do_nothing	yes/no	Whether or not the pages should be parsed
optimisations	display_embeds	yes/no	Whether or not embedded items should be displayed in posts
optimisations	display_images	yes/no	Whether or not images should be displayed in posts

An example configuration:

publishing_info:

```
url: http://example.tumblr.com
username: example@example.com
password: example_password
```

defaults:

```
theme_name: theme
data_name: data # the extension will be automatically appended
rss_url: http://example.tumblr.com/rss
autoreload:
  enabled: no
  interval: 5
```

optimisations:

```
do_nothing: no
display_embeds: yes
display_images: yes
```

TumblrServ follows the principles of Don't Repeat Yourself (DRY) and Convention Before Configuration (CBC), meaning that it will attempt to guess a configuration if none is supplied.

After the server has started, the developer simply needs to point their browser at <http://localhost:8000> (the port number could potentially be different) and they will see their theme.

3. Design

The basic functionality of TumblrServ can be broken down into a procedural list:

1. Parse command line switches
2. Load configuration files
3. Load local data or retrieve data from remote location
4. Parse configuration and data into useable format
5. Assign data to classes
6. Load theme data

CSSE1001

05/09/2009

Assignment 3 - Design Document

7. Adjust theme data slightly to increase compatibility
8. Parse theme data and insert transformed data
9. Generate HTML page and serve via HTTP

Tumblr posts are always given classes inside the web interface, so it makes sense to abstract this data into a heirachy of classes. TumblrServ implements the following classes to do this:

- Post
 - RegularPost
 - PhotoPost
 - QuotePost
 - LinkPost
 - ConversationPost
 - AudioPost
 - VideoPost

These classes are needed because each post type has a specific way of displaying itself and to attempt to parse all of these types in one function would almost definitely make the code unwieldy and difficult to understand. The majority of the work is done by the Post class with specific parsing done in the subclasses

4. Supporting Modules

TumblrServ uses Yaml to store all of its configuration data and uses the PyYaml module to parse the data into sets of dictionaries and lists. Similarly, PyJSON is used to extract data from the Tumblr servers. For the webserver component it uses CherryPy, a well known Python server framework. In order to simplify the upload process, TumblrServ uses the ClientForm HTML form spider library.