CENTER FOR TEXTUAL STUDIES AND
DIGITAL HUMANITIES

DIGH 402 - Introduction to Digital Humanities Design and Programming

Spring Semester 2015

Week 10

PHP and MySQL

Week 9

1. any issues with the current framework and SQL?
2. everything now working correctly for latest code from 402framework repo?

JavaScript Quiz - Part 2

- Google account login
- URL is as follows,

  http://goo.gl/qC2ncH

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Next on our developer's to-do list

- add option for plugins for text, images etc...

- add some more error checking and reporting…

- add default content handler for non controller/format/params URI requests including index.php

- sanitise content request and returned content

- update DB tables, content, metadata...

and so on...

Object Oriented Programming

- update framework to v0.5 - available on GitHub
  - https://github.com/dighteach/402framework
  - https://github.com/dighteach/source/tree/master/2015/DIGH402/402framework/v0.5

- update database SQL to week 10 - available on GitHub
  - https://github.com/dighteach/source/tree/master/2015/DIGH402/402framework/sql/week10

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

handling default views - image/text etc

- default handlers for format in user requested route URI
    - content/image, content/text etc
    - eg: frame/view/image.php

- new constant in directory.php for location of images, texts etc

- viewer classes, eg: ImageViewer, extends BuildHTML class
    - allows easy building and output of HTML required for viewer

- viewer classes also need to check any add-on plugins
    - magnify, zoom etc for image
    - collation, hinman viewer etc for text...

GitHub Code - Constants

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

updated /system/library/loader.php (Loader class)

- after checking for returned content, metadata and selected theme
    - we use $format (image/text etc) to load required viewer
    - check viewer class exists
    - instantiate viewer object
    - set any required viewer or format attributes

- we then use the initial returned raw content data

- content is formatted and returned from the applicable format viewer

- content is passed to draw_theme() method

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

handling default views - /frame/view/image.php (ImageViewer class)

- ImageViewer class in image.php

- abstracts rendering of images from DB

- uses constant for media/images directory

- extends BuildHTML class to allow us to easily build required HTML elements

- private static properties

- format_image_view() method called to create required HTML from content
    - 3 parameters for content, viewer attributes, image attributes
    - customised attributes for div parent wrapper relative to image
    - customised attributes for image element

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

handling default views - /frame/view/text.php (TextViewer class)

- uses similar pattern to ImageViewer class
    - format_text_view() is the main method for rendering the content
    - uses BuildHTML to return the required HTML elements

- abstracts rendering of initial text content from DB

- text currently stored in content table in DB, and not a file etc
    - this can be handled in a similar manner to the images

- format of txt in DB will be dependent upon import or edit plugins
    - eg: HTML format, txt, TEI etc...

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

initial plugin implementation

- new plugin directory for user developed and imported plugins
    - stored in root directory for framework
    - directory = /plugins

- new constant in directory.php for new plugin directory and admin plugins

- plugins initially added to DB tables ‘plugins’ and ‘plugins_lookup’

- code is stored in plugin named directory in /plugins directory
    - eg: /plugins/image_zoom
      /plugins/image_magnify

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

new plugins and plugins_lookup table in DB

- plugin table allows us to store basic information on available framework plugins
    - plugin_id, plugin_name, plugin_desc, and plugin_directory

- plugin_name can be used to display plugin title in framework options

- plugin_desc for tooltips etc

- plugin_directory informs the application where to find the files within the framework's plugin directory

- plugins_lookup table references plugin_id to plugin_type and content type
    eg: plugin_id=1, plugin_type=content, and content_type=image

GitHub Code (SQL) | GitHub Code (plugins)

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

updated /system/library/loader.php (Loader class - adding content and format plugins)

- first we check for available plugins relative to current controller and format in router URI
    - eg: content/image

- load plugin file ‘plugin.php’, set plugin_class, and check that class exists

- instantiate object for plugin_class and call get_plugins() method
    - this checks the available plugins for controller and format
    - checks the plugins_lookup and plugins tables
    - returns the details for the available plugins including plugin_id

- outputs plugin from code in defined plugin_directory
    - eg: css and js files

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

check for installed plugins available in new DB table - /frame/controller/plugin.php

- new plugin.php file and PluginController class
    - file stored as a controller in /frame/controller

- requires query_builder.php file and extends parent class BuildQuery

- queries plugins and plugins_lookup tables in database to check for available plugins
    - check is relative to controller and format specified in router URI

- returns results to Loader class

- also method to check plugin details using plugin_id as a specified parameter

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

use and rendering of user installed plugins - viewer (/system/library/loader.php)

- pass returned plugin_ids to load_plugins() method in Loader class

- load_plugins() prepares plugin details and sets plugins array for use with the View class
    - new $plugins property added to Loader class

- plugins array can now be passed to draw_middle() via draw_theme() method

- draw_plugins() method in View class allows us to output available plugins including
    - plugin name, description, directory
    - required files from plugin where applicable

- these files can now be drawn within the framework template as part of the draw_middle() method

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

use and rendering of user installed viewer plugins - updated View class

- new draw_plugins() method
    - accepts $plugins array parameter
    - checks $plugins array is not empty and then builds HTML for plugin_options
    - loops through $plugins array and formats output for plugin_options
    - uses draw_js() methods to output plugin files and code
        - this loads the plugin for the user

- updated draw_middle() method
    - accepts a new parameter for $plugins
    - calls draw_plugins() method with $plugins parameter before draw_main() method
    - hierarchy allows us to ensure that plugin is above content in DOM

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

viewer plugins - how to build

- built around JQuery Javascript library (or plain javascript etc if you prefer)

- allows injection of JS code within framework content

- allows manipulation, editing etc of existing content

- AJAX manipulation and editing of DOM, texts, images…

- quick and easy plugin development

- easily tested outside of framework before deployment as a plugin

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

viewer plugin - v.basic test image_magnify.js

- create a new JQuery function

- associate mouse click event with element id #image_magnify

- associate toggle event with element id #content and image element within

- hide and show image element for each click

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

helpful Javascript - JQuery UI

- added js and css files to constants

- added JQuery UI css and js files to css and js arrays in draw_theme() method in Loader class

- css and js files are now loaded in the head section of our framework template

- JQuery UI is a very useful UI library for quick deployment of JQuery effects, options…

- eg: dialog/modal windows, accordion effects, autocomplete etc...

GitHub Code

Object Oriented Programming

Working framework so far...v0.5

Home Page

Output Image Content

Output Text Content