1 Directory Structure

1.1 Motivation

Essential directory structure is vital to a successful programming projects. As projects expand and consist of hundreds, thousands, tens of thousands, even hundreds of thousands of files, the only hope for large scale projects is clean directory organization.

1.2 Directory & File Naming

In the web world, one will be dealing with multiple operating systems and multiple browsers, each of which is **NOT** guaranteed to behave identically given the same circumstances. To that end, strictly conforming to a determined directory naming convention is vital. The following rules ***MUST*** be adhered to when naming directories:

1. Use **NO SPACES** under any circumstances for **ANY** reason.
2. Directories will **NOT** contain upper case letters. All letters will be lower case.
3. Files also will **NOT** contain upper case letters **EXCEPT** when it contains a class definition. There are no other exceptions.
4. Directory and file names may only contain the following symbols: _ and -. A dot (.) will be used to separate the base name from the file extension (e.g., .js, .php).

These rules will be strictly enforced and complied with at all times. Close adherence to these rules will serve to prevent many cross-platform problems before they even occur, saving valuable time and effort.

1.3 Daily Directory Naming

Each day will consist of various activities based around a theme. At the start of each activity, create a new directory to house the files and subdirectories necessary to complete the activity, combined with the ISO 8601 date. [1] For instance, if the theme is JavaScript Functions and today's date is August 11, 2014, the resulting directory name is `js-functions-20140811`.

Figure 1: ISO 8601 Date

2 Command Line

2.1 Motivation

There are many graphical tools that developers use to automate certain processes such as uploading, compiling, etc. The advantage of these tools is the automation expedites routine, day-to-day operations. What most of these tools actually do is use templated commands and execute the command line tools transparently. However, there are two reasons the graphical tools will not be used:

1. Graphical tools will not be available for all jobs. This can be because of site restrictions, software incompatibilities, etc.

2. When an unexpected condition occurs, graphical tools often are unable to gracefully handle the situation.

Since the command line can handle all situations, normal and abnormal, and are always available, the command line will be the standard for this class.

2.2 Common Commands

Each operating system handles commands differently, so syntax will vary. However, each operating system does have a fundamental set of commands that handle basic directory and file operations.

Mac OS X	**Windows**	**Comment**
`cd foo`	`cd foo`	change to directory *foo*
`cd ../foo`	`cd ..\foo`	change to directory *foo* (one up)
`cp foo bar`	`copy foo bar`	Copy file *foo* to *bar*
`mv foo bar`	`ren foo bar`	Rename file *foo* to *bar*
`ls`	`dir`	List files in working directory
`ls -l`	`dir /a`	List files in working directory (detailed)
`pwd`	`pwd`	Print working directory
`mkdir foo`	`mkdir foo`	Create directory *foo*
`rm foo`	`del foo`	Delete file *foo*
`rm -Rf foo`	`rmdir /q /s foo`	Delete directory *foo* (**USE CAUTION**)

Table 1: Common Commands

Table 1 contains common commands and their Mac OS X and Windows equivalents. This table is for basic commands and is by no means exhaustive. Please note the directory deletion command will delete **ALL** files from that directory (and the directory itself) **WITHOUT** asking for confirmation. Do **NOT** use it unless absolutely sure the directory can be safely deleted.

2.3 Useful Shortcuts

2.3.1 Arrow Keys

By pressing the up and down arrow keys, one can access the command history. To see the next least recent command, press the up arrow. As one cycles through the command history, the next more recent commands can be accessed by pressing the down arrow. On Mac OS X, the command history is saved in a system file and persists after closing the Terminal. In Windows, the command history is deleted when the Command Prompt is closed.

2.3.2 Drag & Drop

If a directory or file needs to be worked with, it can be typed using tab completion, as seen in Section 2.3.3. Alternatively, one can use the Finder (Mac OS X) or Explorer (Windows) to find the directory or file in question and drag and drop the item into the command line. For instance, to change to a directory using this technique, type "`cd `" (note the space) and drag and drop the directory into the command line. The full path to the directory will be automatically inserted.

2.3.3 Tab

When typing directory and file names, the tab key can be used to "autocomplete" names. For instance, say the following directories exist:

- `js-functions-20140812`
- `js-canvas-20140813`
- `css-selectors-20140811`

At the prompt, if one were to type `cd css<TAB>`, the directory name `css-selectors-20140811` would automatically appear in the original `cd` command. On the other hand, if one were to type `cd js<TAB>`:

- **Mac OS X**: A list of possible candidates will be displayed.
- **Windows**: Each time one presses the tab key, the next candidate is selected.

In any case, the operating system will assist in the completion of the directory or file name.

References

[1] World Wide Web Consortium. Date and time formats. `http://www.w3.org/TR/NOTE-datetime`.