JCR Website Developer Documentation

# Account Information

File Server  
Host: [ftp.dur.ac.uk](ftp://ftp.dur.ac.uk)  
Protocol: FTP  
Encryption: Plain FTP  
Logon Type: Normal  
Username: djb8jcr  
Password: SDKoH721  
Remote Directory: /home/stevens/misc/djb8jcr/public\_html

## MySQL

Management: <http://www.dur.ac.uk/php.myadmin/password/phpMyAdmin/>   
Authenticate using file server or personal credentials at first stage  
Butler JCR database login:  
Username: djb8jcr  
Password: VP4GL6dhF3rYdUGB  
Server: mysql.dur.ac.uk  
Durham University Public Database login used for user information import:  
Username: nobody  
Leave password empty

## Google Analytics and JCR YouTube account

Username: [butler.jcr@durham.ac.uk](mailto:butler.jcr@durham.ac.uk)  
Password: butlerjcr8

## Free Parking ([www.butlerjcr.com](http://www.butlerjcr.com) domain)

Management: <https://www.freeparking.co.uk/>   
Due for renewal/change: 20th July 2016  
Instead of renewal, an alternative system should be found which supports subdirectory redirection.  
Username: dcb8bjcr  
Password: gu84Ee

# Hooks – Global Variables

Without hacking the CodeIgniter core the following additional functionality has been provided:

**Global skip controller - $skip\_controller**

This global variable is used by the pre-controller to tell the Controller constructor to skip calling the controller method. This is used for example when a user does not have permission to view a page.

**Global controller json - $controller\_json**

This is used to override all standard output with custom content. For example when providing a small amount of html or text for an ajax request.

# Hacks to CodeIgniter Core

**./index.php**

Set default time zone (line 3)

Check if offline override exists, and redirect if appropriate (line 6)

Define BASE\_PATH as path to the index.php script (as opposed to the system folder in BASEPATH) (line 154)

Remove testing environment case test (line 45)

**./system/core/CodeIgniter.php**

Check for the ‘skip\_controller’ global variable, which might be set in the post\_controller\_constructor hook. Skip running of the method if so. (line 322)

**./system/core/Output.php**

Replace tabs and newlines in the output (line 370)

**./system/libraries/Zip.php**

Line 91: Replace $date = (@filemtime($dir)) ? filemtime($dir) : getdate($this->now); with $date = (file\_exists($dir)) ? filemtime($dir): getdate($this->now); to prevent clogging error logs with filemtime error on non-existent file, see <https://github.com/EllisLab/CodeIgniter/pull/647>

# How the Hooks Work

This is an overview of how the general page loading process works.

* **Controller initialised** - controller instantiated but no methods called
  1. Controller library changed to add a pointer to the ci object to the global variables allowing access in the hooks
* **Post-controller-constructor hook**
  1. Checks for login
  2. Checks user access rights for the page
  3. Loads head and body start, dealing with ajax stuff, and more.
  4. Checks for page access or login errors and displays them using the skip\_controller variable instead of running the controller if required
* **Controller method** – unless skipped, controller method is run
* **Post controller hook** - This loads the end of the body and footer. The method in the controller might set the global variable controller\_json. This is to override any other output that may occur. Only controller\_json is output. This is to allow for small ajax responses, but does not prevent the logout json sent by post\_controller\_constructor.
* **Output the result**

# Frameworks

|  |  |  |
| --- | --- | --- |
| **Name** | **Version** | **Description** |
| CodeIgniter | 2.2.0 | PHP core framework |
| jQuery | 1.10.2 | Javascript core framework (Google CDN) |
| jQueryUI | 1.10.4 | Provides UI elements including icons, dialog boxes and accordions (Google CDN) |
| Smoothbox | Modified | Picture lightbox (embedded in common.js) |
| jQuery History | 1.8b2 | Updates browser URL for ajax page loads (embedded in common.js) |
| TinyMCE | 4.0.26 | Page content editor (main script served on pencil icon click) |
| jCrop | 0.9.12 | Crop tool on the profile picture page (only served on details page) |

# Maintenance Tasks

There are several tasks which need to be performed at intervals:

* Synchronise users database with CIS main list. New students will be added, current students will be updated in the event that their name or email has changed, graduated students will be kept in the database but the value of the ‘current’ column flag will become 0. Keeping graduated students means that items posted or involving past students can still show names. This should be run at the start of August to remove graduates, end of September to import freshers and every couple of months to add Erasmus or students changing college. This is run by selecting synchronise on the admin page.
* JCR Awards in the archive. Once the file listing the winners of JCR awards is available, FTP it to archive/doc/awards/{year}-{year}.pdf. Add the file name to the array in the archive.php view file.
* During a period of housing reviews, open the property table in the database and sort by id to get the latest entries. The longitude and latitude must be geolocated manually for new properties. The best way to do this is to find the property in Bing Maps and right click on it to get the lat/long values. At the time when I developed this system, the server was rejecting the php curl method of acquiring lat/long and the Google Maps javascript API limited it to 10 requests per day per site key. It may now be worth investigating if either of these systems has been improved to now be viable.
* Sports – update and check team and competition ids match with Team Durham website. Do in November when new competitions are all running

# To Do List

* Charities – fix photo uploading
* Invoice system
* Rewrite format\_users function in admin\_model to process entries on admin levels individually so that the list can be cleared if required, it would be best to rewrite it to make changes on an individual basis per role rather than updating the whole page at once which is a time-consuming and potentially volatile method.
* Add international page content – I was asked to add an international page but was never given any content for it.
* Liversout review and property management system for admin and replace the need for the geolocation maintenance task.
* Create 2x resolution images for header and adjust stylesheet for high dpi screens.
* Proposed: ‘Change It’ system used by Nottingham University Students Union to enable students to suggest changes and vote on changes proposed by others.
* Proposed: Facebook and Twitter integration for societies/sports/committees, group member lists and mailing lists avoiding the majordomo system.
* Proposed: Use categorisation of events to allow filtering on calendar
* Markets - Recaptcha or similar on market orders page if not logged in
* Markets – Expand meal ordering options to allow for more than one order or smaller sizes
* Charities – responsive design of album table

# Functions in Utils Helper

* logged\_in() – checks if the session has a logged in user or not. Returns TRUE/FALSE
* has\_level($level) - determines if the user has the level passed. Variable can be 'admin', ‘any’, ‘logged\_in’ level id, or a mixture
* is\_admin() – checks if the logged in user is an administrator and therefore has full access to the site
* admin\_levs() – returns list of levels which have admin access
* back\_link($page, $generate = FALSE) – generates back link button, can be passed a page but will alternatively go to the last location
* print\_link() – generates print text to trigger the print script
* get\_usr\_img\_src($uid, $type) – returns the link to a user image which has been uploaded. $uid contains the unique user id and $type can be set to a keyword: ‘large’, ‘small’, ‘tiny’. If the image doesn’t exist, the default of the same size is returned
* user\_profile\_a\_open($id) – generates the opening link to a user profile page where $id is the id of the target user
* build\_options($levels, $user\_level) – constructs the list of user levels and can optionally select the level which matches the second parameter
* cshow\_error($message, $status\_code = 500, $heading = ‘An Error was Encountered’) – following a login or site error, this generates the error message. To show it, the skip controller global must also be set
* user\_pref\_name($first, $pref = ‘’, $last = ‘’) – displays a user’s name using the pref name if it’s available. Their surname is optional
* contact\_wm($inner\_text) – creates a link to the admin contact page with the text specified by $inner\_text
* email\_link($id, $inner\_text) – creates a link to contact the user with id matching $id, the text which is displayed on the link is $inner\_text
* success\_code() – eval this code to display success messages in the default style
* error\_code() – eval this code to display error messages in the default style
* token\_ip($page) – create a CSRF token as a hidden field for a form. $page must match that set for validiate\_form\_token in the corresponding controller
* validate\_form\_token($page) – checks the CSRF token has been set and that it matches the one coming from the form submit
* generate\_form\_token($page) – called by token\_ip to create the random token
* rand\_alphanumeric($quantity) – creates a string of length $quantity using random numbers, lowercase & uppercase letters
* rand\_num($min = 0, $max) – generates a random number between min and max
* rand\_uppercase($quantity) – returns a string of length $quantity using uppercase characters
* return\_array\_value($key, $array) – if $array contains $key then its value will be returned
* deleteAll($directory, $empty = false) – deletes everything from the path specified by $directory, if $empty is set to false, it will also delete the directory itself
* get\_last\_location() – returns the url of the last location visited in the session
* tick\_img() – inserts tick icon
* cross\_img() – inserts alert icon
* textarea\_to\_db($v) – strips html breaks from textarea specified in $v
* db\_to\_textarea($v) – strips line breaks from database cell specified in $v so that text can be displayed in a html textarea
* time\_elapsed\_string($ptime) – returns a nice timespan string such as ’47 minutes ago’ approximating the difference between the timestamp in the variable and the current time.
* wotw\_box($colour, $uri, $uri\_text, $time, $description, $facebook, $twitter) – prepares the what’s on this week box using the parameters and url links given
* wotw\_open($heading) – header for a wotw style box. Close the box with a closing div tag
* editable\_area($page, $path, $access\_rights) – create an editable area with $page being the controller that the page is within, $path being the save path of the editable area and $access\_rights being the result of the page\_edit\_rights for the page.