

# Debugging

## Debug Mode

To turn on Debug mode:

- Login to the administration area e.g. `http://YOURSITE/administrator/`
- At the top under the **Site** menu click **Global Configuration**.
- Click the **System** tab.
- Under the **Debug Settings** section change **Debug System** to **Yes**.
- Click the **Save** button.

Debug mode will output a list of all queries that were executed in order to generate the page. This will also turn on a stack trace output for error and warning pages. Hubzero components will also have PHP error reporting turned on, allowing one to see any PHP errors that may be present.

**Note:** Turning on debugging mode for production (live) sites is strongly discouraged and it is recommended to be avoided if at all possible.

## Restricting who sees debug output

Since debug mode can contain potentially sensitive, it is **strongly** recommended that access to debug output is restricted to the administrator or super administrator user access levels and/or a defined list of users.

To restrict:

- Login to the administration area e.g. `http://YOURSITE/administrator/`
- At the top under the **Extensions** menu click **Plugin Manager**.
- Select **System** from the "Select Type" drop-down.
- Find the debug plugin, typically titled "System - Debug", and click to edit.
- Under the **Parameters** section select the **Allowed Groups** and/or enter a comma-separated list of usernames into the Allows Users box.
- Click the **Save** button.

## Inspecting Variables

Hubzero provides the utility class `HubzeroUtilityDebug` for dumping variables.

`dump()`

This will perform a `print_r` on the variable passed, wrapping the output in HTML `<pre>` tags.

`ddie()`

Short for "dump and die", this will perform a `print_r` on the variable passed, wrapping the output in HTML `<pre>` tags and `die()`;

`dlog()`

This method allows developers to dump variables to the debug toolbar, allowing data to be inspected without interrupting the flow or process of the code or output. **Note:** This feature requires the global Debug mode and system debug plugin to be enabled.

### Example

```
$myvar = array(  
    'one' => 'foo',  
    'two' => 'bar',  
);
```

```
HubzeroUtilityDebug::dump($myvar);
```

### Illegal variable ... passed to script.

One encounters the following error:

Illegal variable `_files` or `_env` or `_get` or `_post` or `_cookie` or `_server` or `_session` or globals passed to script.

This error is generated when the key of a key-value pair is numeric in one of the following variables: `_files` or `_env` or `_get` or `_post` or `_cookie` or `_server` or `_session` or globals. An example of this would be `$_POST[5] = 'value'`. This is most often generated by having form elements with numeric values as names. For example:

```
<input type="text" name="5" />
```

As the error indicates, this is not allowed. Element names must include at least one non-numeric character. Examples:

```
<input type="text" name="n5" />
```

```
<input type="text" name="n_5" />
```