WordPress Code Review Procedure

Basic requirements

1. There should be NO custom coding to WordPress Core or to any contributed components (plugins, themes, libraries).
2. Modifications to contributed components or themes should be implemented as custom plugins or subthemes.
3. Disable, uninstall, and remove all plugins, libraries, and themes NOT IN USE.
4. Documentation (README.txt) for custom plugins is required.
5. Custom plugins reside in a **wp-content/plugins/** folder.
6. Custom themes (subthemes and contributed base themes) reside in a **wp-content/themes** folder.
7. Remove any debugging and/or commented out code.
8. Remove development plugins.
9. Remove any source and version control files and folders (ie. .svn .dstore)
10. Stable releases of any contributed plugins are preferred; flag any dev versions and provide explanation for why it was included.

Vendor deliverables

1. Output of **phpinfo** command run at vendor site.
2. List of all plugins and libraries in use.
3. Documentation on any customized components
   a. Include README.txt for custom modules
   b. This documentation should include at a minimum the purpose of the plugin and installation instructions.
4. A single tarball of entire site (core and all)
   a. **wp-content/plugins** folder should include all custom modules
   b. **wp-content/themes** folder should include all custom subthemes and contributed base themes
   c. The site should include a database folder containing the output of the mysql database unload.
   d. The site should include a documentation folder containing the phpinfo command output.
   e. Any version control files (.svn etc) should be excluded.

Internal review process

1. Deploy supplied site in our development virtual environment.
2. Examine phpinfo output for conflicts with Dev environment.
3. Review that the supplied list of plugins is accurate.
4. Verify custom plugins and themes are in proper folders
5. Check for plugins in need of updating.
6. Check for disabled plugins.
7. Verify release levels of any contributed plugins.
8. Check whether core or contributed plugins have been modified.
9. Check for compliance with **WordPress coding standards**.
10. Check for compliance with **theme standards**.
11. Check for **security concerns**.
12. Examine README.txt supplied with custom plugins.

If you wish to discuss a Service Level Agreement, **contact ops-help**.