Application Development Standards

Contents

[General Standards 1](#_Toc119424452)

[Privileged, Sensitive, and Dynamic Information 1](#_Toc119424453)

[Prohibited Components and Frameworks 1](#_Toc119424454)

[Third-Party Components 1](#_Toc119424455)

[Third-Party Services 2](#_Toc119424456)

[Style and Conventions 2](#_Toc119424457)

[Naming: Packages and Directories 2](#_Toc119424458)

[Naming: Variables, Constants, Files, Classes (“Elements”) 2](#_Toc119424459)

[Other Conventions 3](#_Toc119424460)

[Definitions 3](#_Toc119424461)

# General Standards

## Privileged, Sensitive, and Dynamic Information

Credentials, sensitive or privileged information, and data that is subject to frequent change shall not be stored in the Source. This information includes, but is not limited to:

1. Usernames
2. Passwords
3. Private or public keys
4. Certificates
5. Tokens
6. Client Identifiers
7. E-mail addresses
8. Server addresses (hostnames, IP addresses, aliases)
9. File or directory paths
10. Phone numbers
11. Individual or business names

If such information is required for the operation of the Application, the Department of Environmental Protection (DEP) will provide a means by which the information may be obtained from a secure location by the Application at startup and/or at runtime.

## Prohibited Components and Frameworks

No Application shall make use of any Third-Party Component or Framework which has passed its End of Life or Support as declared by its author, will reach its End of Life or Support as declared by its author within two (2) years of its time of inclusion in the Application, or has become defunct by absence of maintenance by its author within six (6) months of its time of inclusion in the Application. Maintenance may include responding to issue reports, communications of intent to release updates to the product, or updates to the product.

No Application shall make use of any Third-Party Component, Framework, or Service that has not reached Maturity or has been properly licensed without prior approval from DEP.

## Third-Party Components

The usage of any Third-Party Components shall require approval by DEP prior to the deployment of the Application to any DEP service, server, or device, and prior to the connection of the Application to any DEP service, server, or device.

At no time shall any Component licensed under the GNU Affero General Public License (AGPL) or equivalent “copyleft” licenses be used in any Application.

It is understood that the usage of a Component with known security vulnerabilities may occasionally be necessary. If such usage is unavoidable, DEP shall, in its sole discretion, determine whether the usage of the Component shall be permitted.

It is understood that the usage of non-free Components may occasionally be necessary. The usage of non-free Components shall require approval by DEP prior to their inclusion in the Application at any point in the Application’s development.

A full manifest of all Third-Party Components shall be provided to DEP for review upon request. The manifest shall include:

1. The common name of the Component
2. The version of the Component
3. The author of the Component
4. The name of the license under which the Component is distributed for the version of the component provided in the manifest
5. The inclusion mechanism of the Component (e.g., Maven, NPM, Composer, etc.)

## Third-Party Services

No Application shall make any connections to any Third-Party Service by any mechanism without prior approval by DEP.

## Style and Conventions

### Naming: Packages and Directories

For applications written in languages that make use of namespaces, packages, or other hierarchical grouping methodologies, the hierarchy of names must follow this convention:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| gov |  |  |  |  |
|  | floridadep |  |  |  |
|  |  | Division acronym |  |  |
|  |  |  | Application name |  |
|  |  |  |  | services |
|  |  |  |  | daos |
|  |  |  |  | models |
|  |  |  |  | controllers |
|  |  |  |  | etc. |

A fully qualified class name (i.e., including its package name or namespace) for a:

1. Service class named “UserService”
2. In an application named “baz”
3. Developed for the Office of Technology and Information Services (OTIS)

is:

gov.floridadep.otis.baz.services.UserService

No Application Source shall be permitted to bear the name of its vendor in filenames, directory names, package names, class names, project names, or in any other place that may imply the vendor’s retention of ownership or intellectual property rights without prior approval by DEP.

### Naming: Variables, Constants, Files, Classes (“Elements”)

1. All Elements of the Application Source shall be named in a descriptive manner
2. Abbreviation of words in these named items is only acceptable when necessary to satisfy compiler limitations or language syntax rules
3. All Element names must be correctly spelled American English words, or phrases thereof, from the Oxford English Dictionary except when the name is an established acronym, initialism, abbreviation, proper noun, trademark, or service mark
4. Elements shall be named as singular nouns or verbs except when they represent a collection of other elements or operators

### Other Conventions

In addition to the standards defined herein, all Application code shall adhere to the appropriate style guide listed below. For Application code written in a programming, scripting, or markup language that is not identified below, the vendor shall request a style guide from DEP.

|  |  |  |
| --- | --- | --- |
| Language | Style Guide | Reference |
| Java | Google Style Guide | <https://google.github.io/styleguide/javaguide.html> |
| PHP | WordPress PHP Coding Standards | <https://developer.wordpress.org/coding-standards/wordpress-coding-standards/php/> |
| HTML and CSS | Google HTML/CSS Style Guide | <https://google.github.io/styleguide/htmlcssguide.html> |
| JavaScript | Google JavaScript Style Guide | <https://google.github.io/styleguide/jsguide.html> |
| TypeScript | Google TypeScript Style Guide | <https://google.github.io/styleguide/tsguide.html> |

# Definitions

Application: The binary package(s) that may be executed or deployed

Component: Any work that may be included in another work to enable or enhance its functionality

Framework: Any set of Components that, together, provide a means of constructing an Application or larger/composite Component

Maturity (as it applies to Components, Frameworks, and Services): Having had at least one (1) viable release that is not labeled as “Beta,” “Test,” “Alpha,” “Release Candidate (RC),” or any other marker to indicate that it is not stable

Service: Any discrete source or consumer of data with which another Service or Application may communicate

Source: The entirety of the uncompiled code which, once compiled and packaged, constitutes the Application

Third-Party Component: Any Component produced by an entity other that DEP, or Component that transitively includes such a Component

Third-Party Framework: Any Framework produced by an entity other than DEP

Third-Party Service: Any Service provided by an entity other than DEP