Back Update Project

WSO2 Coding Standards & Best Practices Guidelines

Contents

General

Comments

- Doc comments
  - All classes and all methods/functions MUST have doc comments
  - Explain each parameter, return type and assumptions made
- Line comments
  - In case you have complex logic, explain any genius logic, rationale for doing something

Logging

- Log then and there
- With ample local information and context
- Remember logs are for users. Make them meaningful, readable and also make sure you spell check (ispell)
- Use correct log level, e.g do not log errors as warnings or vice versa
- Remember to log the error before throwing an exception
- There shouldn't be useless logs which reveals zero information. For example, if you are creating a datasource for a tenant, log should be something like "Datasource: testds for tenant:abc.com was created sucessfully".

- Error logs should reveal where and why it went wrong such that the user / sys-admin can rectify it. For example, "Element not found in config" is a useless log. It should be something like "Element: ServerName was not found in carbon.xml file"

Logic

- Make your genius code readable
- Use meaningful variable names. Remember, compilers can handle long variable names

---

- Variables declared in locality, as an when required
- The underscore character should be used only when declaring constants, and should not be used anywhere else in Java code
- Make sure the function/method names are self descriptive
- One should be able explain a function/method using a single sentence without conjunctions (that is no and/or in description)
    - Have proper separation of concerns
    - Check if you do multiple things in a function
    - Too many parameters are smelly, indicates that something is wrong
- Use variables to capture status and return at the end whenever possible
- Avoid status returning from multiple places, that makes code less readable
- Be consistent in managing state e.g. Initialize to FALSE and set to TRUE everywhere else
- Where does that if block end, or what block did you end right now? Have a comment at end of a block at }
- Use if statements rationally, ensure the behavior is homogeneous
- In case of returning a collection, must return an empty collection and not null (or NULL)
- Do not use interfaces to declare constants. Use a final class with public static final attributes and a private constructor.
- Always use braces to surround code blocks ({}) even if it is a single line.
- Break code into multiple lines if it exceeds 100 columns
- Align method parameters, exception etc. in order to improve readability. Use the settings in your IDE to do this.
- Be sure to define, who should catch an exception when throwing one
    - Be sure to catch those exceptions that you can handle
- Do not use string literals in the code, instead declare constants and use them, constant names should be self descriptive
    - Use constants already defined whenever possible, check to see if someone already declared one, specially in base libs, like Axis2

Java Specific

    - Coding conventions
      - http://www.oracle.com/technetwork/java/codeconv-138413.html
- Only exception is line length, we use 10ce0
    - Run FindBugs on your code - http://findbugs.sourforge.net/
    - Use CONSTANT_VALUE.equals(variable_name) to avoid null pointer exceptions

IMPORTANT

You should run FindBugs on your new code or modified code, and commit only after fixing any bugs reported by FindBugs. It is recommended to use the IntellijIDEA (FindBugs-IDEA) or Eclipse FindBugs plugin to do this.