**Author of the Code: \_\_\_\_\_\_Brannon Chan\_\_\_\_\_\_\_\_\_\_ Name of Code Reviewer: Ahmed Sakr**

**Report Reviewed by \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Assessment of Suitability of Code Provided for Review**

* Too small
* A bit too small
* **Good size**
* A bit too large
* Too large

**Tabulation of Errors**

|  |  |  |  |
| --- | --- | --- | --- |
| **File** | **Line #** | **Checklist Item #** | **Description / Note / Suggested Fix** |
| ToolsFragment.java | 18 | D1 | No proper class documentation string with author and date tag |
| ToolsFragment.java | 23,24 | D1 | Unnecessary comment “private”; it’s clear that is comment |
| ToolsFragment.java | 23 | V1 | Variable name is ambiguous and not clear |
| ToolsFragment.java | 41-45 | E7 | No checks if the objects are null before usage |
| ToolsFragment.java | 55 | D1 | I don’t think you intended to keep this comment here |
| ToolsFragment.java | 61, 66, 71 | D1 | Add method javadocs |
| ToolsFragment.java | 51-57 | S6 | Avoid using anonymous classes to avoid complex code |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | Add more rows as needed |

**Summary of Errors (Minimum of 3, Maximum of 5)**

* Perhaps many of the errors were examples of the same kind (e.g. Always hard-coding values?) Or perhaps the errors signal an overall weak approach (e.g. brittle or smelly code). Look at your table of errors and group them together.
* **Documentation issues: No class or method javadocs**
* **Naming convention: variable names should be renamed to be clear**
* **Code simplification: breakup code to make it easier to follow**

**Summary of Three (3) Recommendations**

* You may feel inclined to make simplistic recommendations that merely repeat your Summary of errors (e.g. Don’t hard-code values) but true recommendations go further and provide strategic or informed suggestions (e.g. Please look into command-line argument and property files as a well to flexibly configure your system).
* **Write Javadoc compliant documentation to allow the reader to get more background into your code.**
* **Name your variables with what function they perform**
* **Divide and conquer the structure of your code so that you have submodules that perform the task**

**Overall Evaluation (Beginning, Developing, Accomplished, Exemplary)**

Give your impression along with a rationale.

**Developing**

You have good code, but it could be improved with better documentation, better variable naming, and restructuring of your code.