**Coding Standard Template**

|  |  |
| --- | --- |
| Purpose | To guide the development of JAVA programs |
| Program Headers | Begin all programs with a descriptive header. |
| Header Format | /\*\*  \* Count lines of code!  \* The Prog2-LOC-Counter program implements an application that  \* counts lines of codes, methods and classes of programs.  \* Note: Giving proper comments in your program makes it more  \* user friendly and it is assumed as a high quality code.  \*  \* @author Felipe Marín  \* @version 1.0  \* @since 2016-02-29  \*/ |
| Listing Contents | Provide a summary of the listing contents. |
| Contents  Example | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* Listing Contents: \*/  /\* Reuse instructions \*/  /\* Modification instructions \*/  /\* Compilation instructions \*/  /\* Includes \*/  /\* Class declarations: \*/  /\* FileHandler \*/  /\* Counter \*/  /\* Prog2LOCCounter \*/  /\* Source code in /home/felipe/NetBeansProjects/Prog2-LOC-Counter/src \*/  /\* Counter \*/  /\* countLOC() \*/  /\* getLOCCounter() \*/  /\* setLOCCounter(int LOCCounter) \*/  /\* getMethodCounter() \*/  /\* setMethodCounter(int methodCounter) \*/  /\* getPathArray() \*/  /\* setPathArray(ArrayList<String> pathArray) \*/  /\* FilesHandler \*/  /\* actionPerformed(ActionEvent e) \*/  /\* createImageIcon(String path) \*/  /\* getPathsArray(String path) \*/  /\* createAndShowGUI() \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ |
| Reuse Instructions | * Describe how the program is used. Provide the declaration format, parameter values and types, and parameter limits. * Provide warnings of illegal values, overflow conditions, or other conditions that could potentially result in improper operation. * Make use of the annotation tags that JAVA provides to define parameters, return types, etc. |
| Reuse Example | //\*\*  \* This method is used to count the lines of code of a set of files.  \* @return int This returns the lines of code of the files.  \*/  public int countLOC() {  .  .  .  }  /\*\*  \* Returns an ImageIcon, or null if the path was invalid.  \*  \* @param path  \* @return  \*/  protected static ImageIcon createImageIcon(String path) {  .  .  .  } |
| Identifiers | Use descriptive names for all variables, function names, constants, and other identifiers. Avoid abbreviations or single letter variables. |
| Identifier Example | Int LOCCounter; /\* This is GOOD \*/  String x4, j, ftave; /\* This is BAD \*/ |

(continued)

**Coding Standard Template (continued)**

|  |  |
| --- | --- |
| Comments | * Document the code so that the reader can understand its operation. * Comments should explain both the purpose and behavior of the code. * Comment variable declarations to indicate their purpose. |
| Good Comment | if (sCurrentLine.endsWith("{")) //Does the current line end with “{“? |
| Bad Comment | if (sCurrentLine.endsWith("{")) //Check if current line ends with “{“ |
| Major Sections | Precede major program sections by a block comment that describes the processing that is done in the next section |
| Example |  |
| Blank Spaces | * Write programs with sufficient spacing so they do not appear crowded. * Separate every program construct with at least one space. |
| Indenting | * Indent every level of brace from the previous one. * Open and closing braces should be on lines by themselves and aligned with each other. |
| Indenting  Example | public int countLOC() {  for (String path : pathArray) {  try (BufferedReader br = new BufferedReader(new FileReader(path))) {  String sCurrentLine;  while ((sCurrentLine = br.readLine()) != null) {  if (sCurrentLine.endsWith("{")) {  LOCCounter++;  } else if (sCurrentLine.endsWith(";")) {  LOCCounter++;  }  System.out.println(sCurrentLine);  }  System.out.println("LOC para clase " + path + " = " + LOCCounter);  } catch (IOException e) {  e.printStackTrace();  }  }  return LOCCounter;  } |
| Capitalization | * Capitalize all classes. * Lowercase all methods and variables, capitalizing each following words. * All capitalize final variables. * To make them readable, user messages may use mixed case. |
| Capitalization Example | public class Counter  int methodCounter = 0;  static private final String NEW\_LINE = "\n"; |