SYA-React Coding Standard

Professor Jullig
CMPS - 115 - Software Engineering Project 1

1 Formatting:

- **1.1 Braces:**
  Will be used where optional (After if, else if, else statements even when the body is empty or contains only a single element)
- **1.1.1 Non-Empty Blocks:**
  Braces will follow Kernighan and Ritchie style for Empty and Non-Empty blocks with a modification to closing braces.
  - No line break before the opening brace.
  - Line break after the opening brace.
  - Line break before the closing brace.
  - Line break after the closing brace. (Modification)

```
Example:

return new MyClass() {
  @Override public void method() {
    if (condition()) {
      try {
        something();
      }
      catch (ProblemException e) {
        recover();
      }
    }
    else if (otherCondition()) {
      somethingElse();
    }
    else {
      lastThing();
    }
  }
};
```

- **1.1.2 Empty Blocks: may be concise.**

  ```
  Example:

  // This is acceptable
      void doNothing() {}

  // This is equally acceptable
      void doNothingElse() {
      }
  ```

- **1.2 Assignment and Equality**

  - Single space before and after equals during assignment.
  - Single space before and after equality operator.

    ```
    Examples:

    onClick = {function}

    if (variable == anotherVariable) {
      doesSomething();
    }
    ```

- **1.3 One Statement Per Line**
  Each statement is followed by a line break.

- **1.4 Block Indentation: +2 Space**

  Each time a new block or block-like construct is opened, the indent increases by two spaces. When the block ends, the indent returns to the previous indent level. The indent level applies to both code and comments throughout the block.

- **1.5 Standard Indentation: 2 spaces**

- **1.6 Whitespace**

- **1.6.1 Vertical Whitespace**
  A single blank line always appears:

  - Between consecutive members or initializers of a class: fields, constructors, methods, nested classes, static initializers, and instance initializers.
  - As required by other sections such as *1.2* as defined within this document.

A single blank line may also appear anywhere it improves readability, for example between statements to organize the code into logical subsections. A blank line before the first member or initializer, or after the last member or initializer of the class, is neither encouraged nor discouraged.

Multiple consecutive blank lines are permitted, but never required (or encouraged).

- **1.6.2 Horizontal Whitespace**
  Beyond where required by the language or other style rules, and apart from literals, comments, a single ASCII space also appears in the following places only.
  - Separating any reserved word, such as if, for or catch, from an open parenthesis that follows it on that line.
  - Separating any reserved word, such as else or catch, from a closing curly brace that precedes it on that line.
  - On both sides of any binary or ternary operator. This also applies to the following "operator-like" symbols.

  This rule is never interpreted as requiring or forbidding additional space at the start or end of a line; it addresses only interior space.

2 Name Schemes:

- **2.1 Variable names:**
  - camelCase, starting with lowercase first letter.
- **2.2 Class names:**
  - CamelCase, starting with uppercase first letter.
  - Exceptions include classes under 'Reducers' (Due to Redux)
- **2.3 Function names:**
  - camelCase, starting with lowercase first letter.
- **2.4 Files names:**
  - camelCase, starting with lowercase first letter.

3 Commenting Guidelines:

- **3.1 Unused Functions:**
  - Remove unused variables and functions before commiting, if it is debugging code, comment it as such.

- **3.2 General Form:**
  The basic formatting of commenting is as follows:

```
/*
* Multiple lines of comment text are written here,
* wrapped normally...
*/
public int method(String p1) { ... }
```

  Or in this single line example:

```
//Single line of comment text

  OR

/* Single line of comment text */
```

- **3.2.1 Comments before functions:**
  - Comments are required before a function declaration with at minimum a summary of what the function is doing.
- **3.2.2 In-line function comments:**
  - In-line comments are optional, but if used, must be aligned at the same level of indentation throughout that function.

```
Example of in-line comments:

MyFunction(props) {
  @Override public void method() {
    if (condition()) {
      something();                  //does thing
    }
    else if (otherCondition()) {    //notice indentation
      somethingElse();              //edge case thing
    }
    else {
      lastThing();                  //most common thing
    }
  }
};
```