Assignment One Review

Brae

May 9, 2018

CSSE2002: Programming in the Large

Common Mistakes - Abstract Thing

The JavaDocs specified that the `Thing` class was an abstract class.

```
public class Thing {
        ...
}

public abstract class Thing {
        ...
}
```

Common Mistakes - Code Duplication

Code duplication for replacing strings.

Common Mistakes - Code Duplication

Code duplication for replacing strings.

Don't do this

```
public Thing(String shortDescription, String
    longDescription) {
    this.shortDescription = shortDescription.replace('
        \n', '*').replace(';', '*').replace('\r', '*')
        ;
    this.longDescription = longDescription.replace('\n
        ', '*').replace(';', '*').replace('\r', '*');
}

protected void setShort(String shortDescription) {
    this.shortDescription = shortDescription.replace('
        \n', '*').replace(';', '*').replace('\r', '*')
        ;
}
```

Common Mistakes - Code Duplication

Code duplication for replacing strings.

Abstract away repeated code

```
public Thing(String shortDescription, String
    longDescription) {
    this.shortDescription = replaceDescription(
        shortDescription);
    this.longDescription = replaceDescription(
        longDescription);
}

private String replaceDescrition(String description) {
    return description.replace('\n', '*')
            .replace(';', '*')
            .replace('\r', '*');
}
```

Common Mistakes - Member Duplication

```
public class Explorer extends Thing implements Mob {

    private String shortDescription;
    private String longDescription;

    public Explorer(String shortDescription, String
        longDescription, int health) {
        super(shortDescription, longDescription);
        this.shortDescription = shortDescription;
        this.longDescription = longDescription;
        ...
    }
}
```

Common Mistakes - Member Duplication

```
public class Explorer extends Thing implements Mob {

    public Explorer(String shortDescription, String
        longDescription, int health) {
        super(shortDescription, longDescription);
        ...
    }
}
```

Common Mistakes - String Comparison

Strings need to be compared using the .equals method not the comparison operator.

```
String first = "hello";
String second = "hello";

if (first == second) {} // wrong
if (first.equals(second)) {} // right
```

Common Mistakes - Style

Horizontal space is required on both sides of any binary or ternary operator.

Separate any reserved word, such as if, for or catch, from an open parenthesis (() that follows it on that line.

```
if (x) {} // right
if(x){} //wrong
if (x){} //wrong
if(x) {} //wrong
```

Common Mistakes - Style

Name Case

Variable names should be in camelCase

Class names should be in PascalCase

Constants should be in `SCREAMING_SNAKE_CASE`

Method names should be in camelCase

Common Mistakes - Style

Name Case

Variable names should be in camelCase

Class names should be in PascalCase

Constants should be in `SCREAMING_SNAKE_CASE`

Method names should be in camelCase

```
public class MyClassName {
        private static final int MAX_SCORE = 1000;
        private int bestScore = 0;

        public int getBestScore() {
                int myScore = 1;
                return myScore;
        }
}
```

Common Mistakes - Style

There is no line break after a curly brace if it is followed by else or a comma.

Wrong

```
if (x) {

}
else {

}
```

Right

```
if (x) {

} else {

}
```

Common Mistakes - Style

One line if statements

```
if (x) { return y; }
```

Braces are used with if, else, for, do and while statements, even when the body is empty or contains only a single statement.

```
if (x) return y;
```

Common Mistakes - Commenting

Class Comments Used at the top of every file. Explains important details about the class.

```
/*
 * An exception which is thrown when the programmer
     becomes unhappy
 */
 public class UnhappyProgrammerException extends
     Exception {

 }
```

Inline Comments Used when code is not immediantly obvious.

```
// subtracts one from all elements in the array
for (int i = 0; i < numbers.length; i++) {
    numbers[i] = numbers[i] - 1;
}
```

Common Mistakes - Commenting

Method Comments Used to explain the purpose of a method. In assignment two you will be using JavaDcos comments for methods.