<!doctype html public "-//w3c//dtd html 4.0 transitional//en">Code Review Checklist

Reviewer: Richard

Code by: Jay

**Files Under Review**

FullscreenActivity.java

**Specifications**

Coding Guidelines/ Coding Style

[Y] Is the functionality described in the specification fully implemented by the code?   
[Y] Is there any excess functionality in the code but not described in the specification?

**Correctness**

1. Initialization and Declarations   
  
[Y] Are all local and global variables initialized before use?   
[Y] Are variables and class members of the correct type and appropriate mode   
[Y] Are variables declared in the proper scope?   
[Y] Is a constructor called when a new object is desired?   
[Y] Are all needed import statements included?  
  
2. Method Calls   
  
[Y] Are parameters presented in the correct order?   
[Y] Are parameters of the proper type for the method being called?  
[Y] Is the correct method being called, or should it be a different method with a similar name?   
[Y] Are method return values used properly? Cast to the needed type?

3. Arrays   
  
[N] Are there any off-by-one errors in array indexing?   
[N] Can array indexes ever go out-of-bounds?   
[N] Is a constructor called when a new array item is desired?

4. Object Comparison   
  
[Y] Are all objects (including Strings) compared with "equals" and not "=="?

5. Output Format   
  
[N] Are there any spelling or grammatical errors in displayed output?   
[Y] Is the output formatted correctly in terms of line stepping and spacing?

6. Computation, Comparisons and Assignments   
  
[Y] Check order of computation/evaluation, operator precedence and parenthesizing   
[N] Can the denominator of a division ever be zero?   
[N] Is integer arithmetic, especially division, ever used inappropriately, causing unexpected truncation/rounding?   
[Y] Check each condition to be sure the proper relational and logical operators are used.   
[Y] If the test is an error-check, can the error condition actually be legitimate in some cases?   
[N] Does the code rely on any implicit type conversions?

8. Exceptions

[Y] Are all relevant exceptions caught?   
[N] Is the appropriate action taken for each catch block?

9. Flow of Control   
  
[N/A] In a switch statement is every case terminated by break or return?   
[N/A] Do all switch statements have a default branch?  
[Y] Check that nested if statements don't have “dangling else” problems.   
[Y] Are all loops correctly formed, with the appropriate initialization, increment and termination expressions?   
[Y] Are open-close parentheses and brace pairs properly situated and matched?

10. Files   
  
[Y] Are all files properly declared and opened?   
[Y] Are all files closed properly, even in the case of an error?   
[Y] Are EOF conditions detected and handled correctly?   
[Y] Are all file exceptions caught?

**Qualityof Documentation**

1. Classes

[N] Are all classes documented with what they do?

[Y] Are constructors documented with a description?

[Y] Are constructor parameters documented?

2. Methods

[Y] Are all methods documented with a description?

[N] Are all method parameters documented?

[N] Is the return value documented for return methods?

3. Instance Variables

[N] Are the variables organized properly?

[Y] Are the variables described if their function is vague?

4. Comments

[Y] Are there comments where code can be unclear or complicated?

**Quality of Testing**

[Y] Overall, is the code easily testable using unit testing?

[N] Is the code formatted enough to make integrating testing easier?

**Comments**

Some minor variable name should be more specific, but overall the code is easy to read, indented well and comment nicely.