Testing Document Sprint Week 4

Total branch coverage: 65%

For our project we mainly looked at branch coverage. We did this because a lot of our code is GUI code which cant be tested. Usually GUI code does not consist of a lot of branches so by measuring branch coverage we thought we had the most trustworthy metric.
Obviously 65% is a lot more than our previous 30%. The main reason for this is that we tested all of our Units thoroughly this time. Below is a report of what we tested and what we thought could not be tested

nl.tudelft.sem.group2 74%
Apart from the LauncherApp class and the AreaState Enum (which can not really be tested) we tested this package pretty well. Logger and AreaTracker have 83% and 80% coverage respectively and ScoreCounter even has a 100% branch coverage (only 2 branches though).

nl.tudelft.sem.group2.game.Board 11%
This class only has a meager 11% branch coverage. Is this necessarily a bad thing? Probably not. The Board class mostly contains GUI methods for drawing on the canvas. The only method we could have probably tested more is the collisions() method. We tested the intersection methods behind it but not the collisions() method itself.

nl.tudelft.sem.group2.global.globals 0%
This class is just a list of constants, no need for testing.

nl.tudelft.sem.group2.scenes 0%
The scenes package has a very high probability of being changed in the near future. We plan to move the functionalities of GameScene and ScoreScene to separate classes. Because of this, we decided to start writing these tests later on.

nl.tudelft.sem.group2.units 97%
The units package has been tested for 97% branch coverage. We tested the draw methods using Mockito, this is why the percentage is so high. Some classes even tested with 100%. Nice job!