Testing

Team 21: 21Direction

Zain Alshaikh

Corin Bertrand

Damian Heaton

Mandy Li

Brandon Oliver

Toms Tafijs

[1] C. Politowski. F. Petrillo. Y. Gueheneuc. (2021, March) *“A Survey of Video Game Testing”* . https://arxiv.org/pdf/2103.06431.pdf

Black Box Testing:

a)

A significant part of our testing was done via black box testing, this decision was made due to the fact that automation of a lot of the tests goes beyond the requirements of the ENG1 assignment. Subsequently, we did research on the best ways to do testing to meet market standards and we concluded that the obvious option was to do a form of end-to-end testing called game-play testing. This decision was made as there are a lot of functionalities in the project that could not be automated such as the HUD which is best seen when playing the game. In addition, the large majority of our requirements are able to be shown using black box testing which allows us to show the game in a practical aspect. One recent relevant study has stated that they "report that manual game-play testing, a form of end to-end testing also called play-testing, is the primary testing technique used by game developers." [1] This was an important factor in our decision making as it allows us to make sure that we are following industry standards.

White Box Testing:

We chose to use unit testing for the parts that we could justify unit testing as being necessary for, such as the automated processes. This was used because we knew that we had to test the assets and the sound files of the game, and because the testing didn't need to test any runtime possibilities. As with all code, as soon as we implemented any features such as the new entities, we tested at runtime whether it worked as it was originally planned. Anything that we found that didn't work would need some revision, so the code was then checked over to see where the issue arose and what needed to be done to fix it or improve it.

Initially, we aimed to use JUnit 5. However, after a lot of time and investigation, we could not get it working. On the other hand, there were no issues with JUnit 4, which proved to be more helpful in future, as JUnit 4 has all of its files packaged into a singular jar file rather than having the 3 separate sub-projects with JUnit 5. In addition, JUnit 4 allows more computers to run the tests as it only needs Java 5 and upwards rather than Java 8 and up, as is the case with JUnit 5. This means that we could allow older computers to run the game and the tests as it doesn't then need newer more updated versions of Java. We used a GdxTestRunner class which you can access at https://github.com/Lyrenhex/ENG1-Project-Part2/blob/main/core/src/test/java/com/lyrenhex/GdxTestsRunner.java

Justification:

We believe that using a joint approach of the white box unit tests for the automated processes and the black box testing for the majority of the rest works perfectly for the assessment due to the scale of the game and the limited time frame that we had.

[1] C. Politowski. F. Petrillo. Y. Gueheneuc. (2021, March) *"A Survey of Video Game Testing"* . https://arxiv.org/pdf/2103.06431.pdf

b)

Black Box Testing:

https://lyrenhex.github.io/ENG1-Project-Part2/v2/testing

Manual testing report: https://lyrenhex.github.io/ENG1-Project-Part2/pdfs/BlackBoxTests.pdf

Tests (on Google Drive):
https://drive.google.com/drive/u/1/folders/1dxT1MbrVHkXZ6CBkQgcMcMzQl8yhtvif

Statistics:

We aimed to cover all of the user requirements using the black box testing mainly as the large portion of the requirements can be shown through manual testing. There are a few exceptions which aren't possible to test with either type of testing architecture.

Tested User Requirements: 21/24

User Requirements not Tested:

UR_PERFORMANCE – This isn't tested as the hardware can vary massively however it has run smoothly on all devices that we have played it on.

UR_FUN – Whether a game is fun is subjective so we cannot possibly test this aspect.

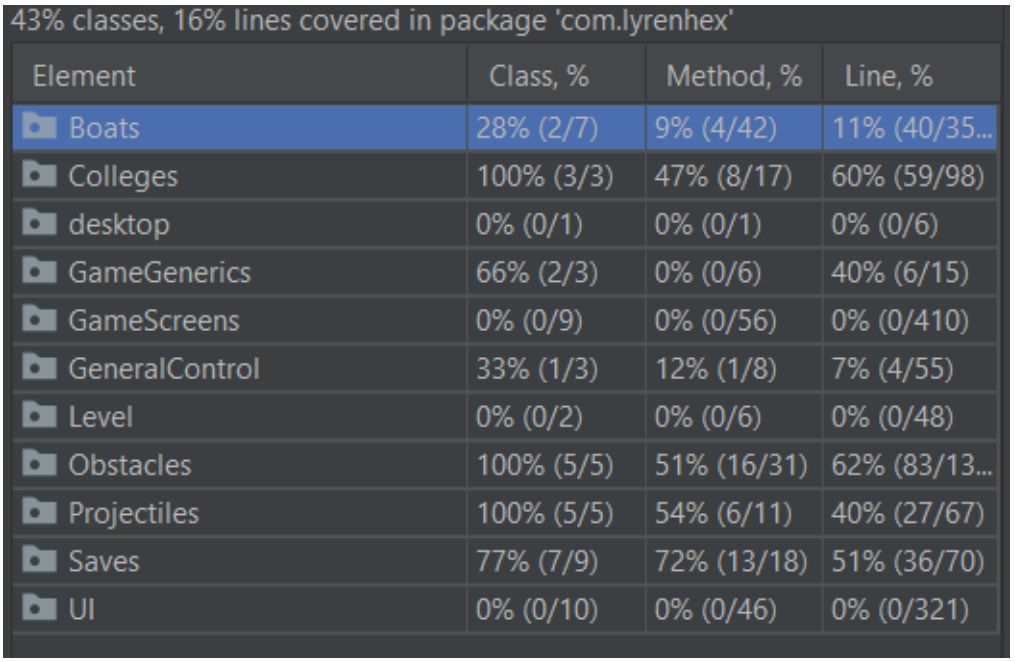
UR_CHILD_FRIENDLY – The game doesn't have any graphic content or gory aspects but regardless this isn't something we could test.

Tested Functional User Requirements: 25/25

Testing Overview:

The testing information that we have coded has shown to be very complete and that the tests are correct for all the given scenarios that we have needed to check over and fix if it was needed. Overall, it is a quite comprehensive set of tests to check for all the assets, make sure the game screen runs and check if the testing infrastructure works and all this combined gives us a solid idea of the function of the game. It also shows us how robust the code is as it works every time that we have run it and there have been no errors or issues that are apparent at this stage. The manual testing is very thorough and covers all of the possible user requirements that can be tested, and the unit tests manage to cover all of the automated processes that we felt like needed to be tested.

[1] C. Politowski. F. Petrillo. Y. Gueheneuc. (2021, March) *"A Survey of Video Game Testing"* . https://arxiv.org/pdf/2103.06431.pdf



43% classes, 16% lines covered in package 'com.lyrenhex'

Element	Class, %	Method, %	Line, %
Boats	28% (2/7)	9% (4/42)	11% (40/35...
Colleges	100% (3/3)	47% (8/17)	60% (59/98)
desktop	0% (0/1)	0% (0/1)	0% (0/6)
GameGenerics	66% (2/3)	0% (0/6)	40% (6/15)
GameScreens	0% (0/9)	0% (0/56)	0% (0/410)
GeneralControl	33% (1/3)	12% (1/8)	7% (4/55)
Level	0% (0/2)	0% (0/6)	0% (0/48)
Obstacles	100% (5/5)	51% (16/31)	62% (83/13...
Projectiles	100% (5/5)	54% (6/11)	40% (27/67)
Saves	77% (7/9)	72% (13/18)	51% (36/70)
UI	0% (0/10)	0% (0/46)	0% (0/321)

Statistics:

This overview shows us that the unit tests that we were able to automate covers 43% of all the classes making up just under half of our testing. As shown in the IDE created report, 19 of the classes in both the UI and the GameScreens haven't been tested, this is because these cannot be automated, and they need to be done with black box manual testing. 16% of the lines of the code are covered by the testing also which is a significant amount of them.

We created tests for the main classes that had many features and the ones that were changed the most, this was to be able to eliminate any bugs or issues that arose.

All of the automated tests have passed with no issues.

[1] C. Politowski. F. Petrillo. Y. Gueheneuc. (2021, March) *"A Survey of Video Game Testing"* . https://arxiv.org/pdf/2103.06431.pdf

C)

Gradle Unit Tests Report: https://lyrenhex.github.io/ENG1-Project-Part2/v2/test/

Black Box Testing: https://lyrenhex.github.io/ENG1-Project-Part2/v2/testing

Manual testing report: https://lyrenhex.github.io/ENG1-Project-Part2/pdfs/BlackBoxTests.pdf

Tests (on Google Drive):
https://drive.google.com/drive/u/1/folders/1dxT1MbrVHkXZ6CBkQgcMcMzQl8yhtvif

[1] C. Politowski. F. Petrillo. Y. Gueheneuc. (2021, March) *"A Survey of Video Game Testing"* .
https://arxiv.org/pdf/2103.06431.pdf