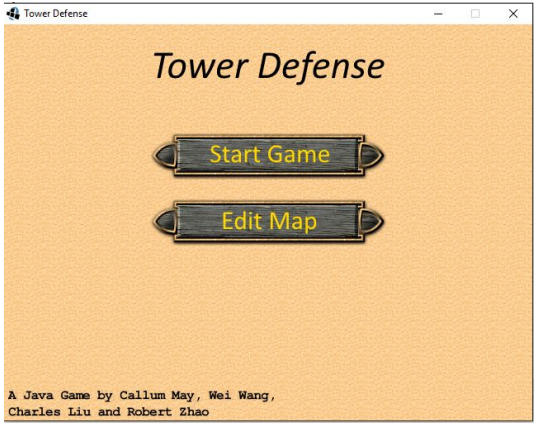
Project for Software Testing and Debugging: Testing on Tower Defense Game

Group Members: Zayn Zaidi, Zejun Ma, Raul Arevalo

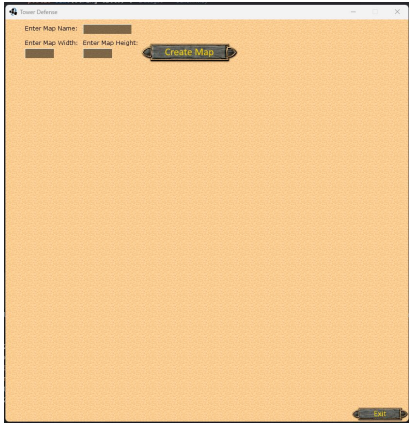
Our Software Under Test: Java Tower Defense Game

- Project Link: https://github.com/callumdmay/java-tower-defense/tree/master
- Java Tower Defense is a classic tower defense game implemented in Java. In this game, players place towers along a path to prevent waves of enemies from reaching the end.



Start Page

A Level of Game



Map Editor

Project Statistics & Structure

Total Files: 59 Java files

Lines of Code

There are 3345 lines of code in java files.

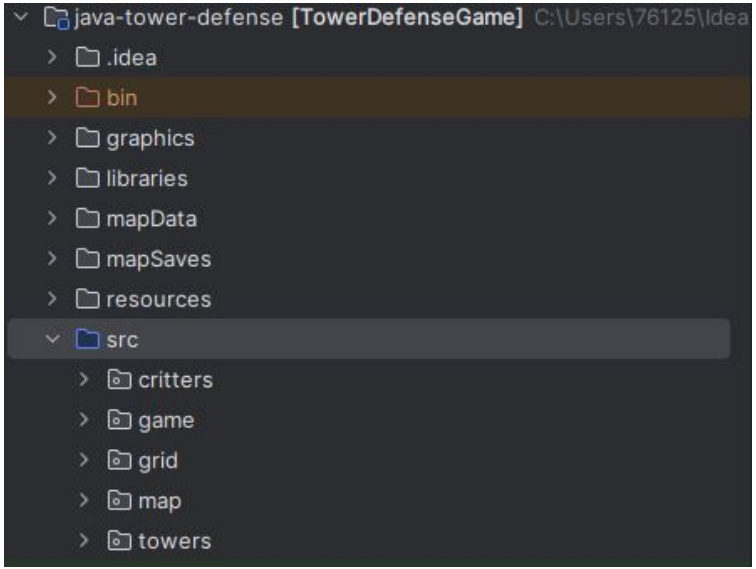
Number of Classes

There are 27 classes.

Number of Methods

There are approximately 210 methods.

- Developed using Eclipse IDE
- Uses Lightweight Java Game Library (LWJGL)
- Uses Slick2D for simple 2D graphics



Project Statistics & Structure

- Critter Classes
  - ArmoredCritter, BossCritter, CritterGenerator, GruntCritter, ScoutCritter, TankCritter
- Map
  - Create Map, Edit Map, Large Map, Small Map
- Towers
  - Basic Tower, Freeze Tower, Sniper Tower
- Games
  - Screens: Map Screen, Menu Screen, Edit Screen, Player Screen ……
  - Player Class
  - Launching Game

Testing

- What We Achieved on Testing
  - WhiteBox Testing (Unit Testing/Integration Testing)
  - BlackBox Testing
  - Mock Testing
  - Usability Testing
- Testing Goal
  - A report of revealed faults
  - Branch coverage/ Statement coverage/ Condition coverage
  - A comprehensive validation of what the game should do and not should do

Whitebox Testing

- Know and rely on internal behavior of classes (tower, critter, map, player)
- Test direct state changes
  - ex: BuyingCost, UpgradeCost, Health
- Test time-based logic
  - Cooldown of tower attacks
- Test edge cases
  - Attempting to attack too soon after previous attack
  - Attempting to upgrade without enough credits
  - Attempting to upgrade beyond max level
- Integration Tests
  - Test interactions between Tower and Player classes, Tower and Critter classes, Tower and Map classes, etc.

Statement Coverage

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
critters	93.6 %	1,020	70	1,090
Critter.java	86.3 %	428	68	496
CritterGenerator.java	99.6 %	487	2	489
ArmoredCritter.java	100.0 %	21	0	21
BossCritter.java	100.0 %	21	0	21
GruntCritter.java	100.0 %	21	0	21
ScoutCritter.java	100.0 %	21	0	21
TankCritter.java	100.0 %	21	0	21
towers	91.5 %	518	48	566
Tower.java	88.6 %	203	26	229
Projectile.java	89.7 %	191	22	213
BasicTower.java	100.0 %	39	0	39
FreezeTower.java	100.0 %	44	0	44
SniperTower.java	100.0 %	41	0	41
grid	100.0 %	60	0	60
MapTile.java	100.0 %	8	0	8
PathTile.java	100.0 %	18	0	18
Tile.java	100.0 %	34	0	34

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
map	88.4 %	1,295	170	1,465
MapEditor.java	33.9 %	58	113	171
LoadFile.java	77.9 %	152	43	195
Map.java	98.6 %	992	14	1,006
LargeMap.java	100.0 %	31	0	31
MediumMap.java	100.0 %	31	0	31
SmallMap.java	100.0 %	31	0	31

Branch Coverage

Element	Coverage	Covered Branches	Missed Branches	Total Branches
map	81.2 %	104	24	128
Map.java	85.2 %	92	16	108
MapEditor.java	0.0 %	0	6	6
LoadFile.java	85.7 %	12	2	14
LargeMap.java		0	0	0
MediumMap.java		0	0	0
SmallMap.java		0	0	0

Element	Coverage	Covered Branches	Missed Branches	Total Branches
critters	80.0 %	40	10	50
Critter.java	65.4 %	17	9	26
CritterGenerator.java	95.8 %	23	1	24
ArmoredCritter.java		0	0	0
BossCritter.java		0	0	0
GruntCritter.java		0	0	0
ScoutCritter.java		0	0	0
TankCritter.java		0	0	0

Element	Coverage	Covered Branches	Missed Branches	Total Branches
towers	87.5 %	14	2	16
Projectile.java	87.5 %	7	1	8
Tower.java	87.5 %	7	1	8
BasicTower.java		0	0	0
FreezeTower.java		0	0	0
SniperTower.java		0	0	0

Faults Found - Whitebox

- The takeDamage function of critter does not properly handle negative damage (adds health)
- Does not properly handle loading of invalid files

BlackBox Testing

- Code was all visible from the start, so had to act as if it wasn’t there for blackbox
- Some functions were simple and had to be put together for testing like setters and getters
- Two Faults Found
  - Constructor for EditMap doesn’t assign the value of userInput to its respective variable
  - Taking negative damage does not throw an exception

Usability Testing

- 6 users (3 novice, 3 experienced with Tower Defense games)
- This usability test was conducted to evaluate how intuitively players could navigate and play the Java Tower Defense game. We focused on six key interaction tasks and observed participants' behavior, confusion points, and success rates. The majority of participants were able to perform core tasks but struggled with locating tower upgrades and interpreting the enemy attributes.

Mock Testing

- GameMockTest.java
  - Tests Critter initialization, movement, damage, freezing/unfreezing
  - Covers behavior at end points and movement direction updates
- MapMockTest.java
  - Verifies tile retrieval, entry/exit placement
  - Validates path construction (linkTwoPoints), conversion to binary map
  - Confirms overall map validity via ValidityOfMap
- TowerMockTest.java
  - Tests property initialization and upgrade logic
  - Validates refund mechanism, cooldown-based attacks
  - Uses Mockito to simulate Critter for angle rotation tests

Conclusion

- Main sections under src were grid, grame, critter, tower, and map for testing
- Whitebox, blackbox, usability, and mock testing carried out
- Faults were found, which gives us a good validation on the game.
- Usability indicates improvement on user experience, UI design, game feedback
- Could do GUI testing in the future

Q&A

Thanks