Sprint Retrospective, Iteration # 4

Group: 8008

User Story	Task	Member responsible for the task	Estimated Effort per Task *(in hours)*	Actual Effort per Task *(in hours)*	Done *(yes / no)*	Notes
As a developer I want to implement design patterns in the code so that the code becomes more clear and easier to implement new features.	Write out what design patterns to use and why they are important.	Nick	1	1	Yes	-
	Create a class diagram of the design patterns	Nick	2	1	Yes	-
	Create a sequence diagram of how the design patterns go through the code	Nick	2	1	Yes	-
As a player I want to have a multiplayer mode so that I can play the game with a friend.	Implement the Singleton	Nick	2	1	Yes	-
	Implement the Decorator pattern	Nick	6	6	Yes	Also implemented the darkness decorator and the darkness mode
	Create a requirements document of the new feature	Eric	2	2	Yes	-
	Create an UML of this new feature	Eric	3	1	Yes	-
	Implement the new feature	Eric	8	8.5	Yes	-

As a developer I want to have clearer code in the class BlockFactory and a better structure of how platforms are created so that we can easily expand these classes.	Refactor the BlockFactory and the Platform class	Nick	5	0	No	Platform class already refactored by implementing the decorator. Refactoring the BlockFactory was determined non-critical.
As a player I want to have another gaming experience than the 'normal' Doodle Jump so that this game is special.	Add a new feature that is not in the normal Doodle Jump, that is, add missions that get you coins that you can use to upgrade/buy new powerups Create the requirements for this new feature. Create an UML of this new feature. Create an UML of the ProgressionObserver hierarchy Make new sprites to draw the missions Add a ProgressionManager that manages all progression (i.e. missions and coins)	Joost, Michael	Michael: 14 Joost: 20	Covered by accumulation of the others 1 ¼ ½ ¾ 10	Yes for all	Code for the upgrades for the trampoline are done. Upgrades for the jetpack still lack sprites. Both lack sound. UML and requirements have been uploaded in git. The ProgressionManager is fully implemented and the mission / coins system works.

	Implement the observer pattern to communicate between progression observables and corresponding observers Draw the missions and coins in the PauseScreen			5 5		
As a player I want to have enemies so that the game becomes more challenging.	Implement at least 1 enemy Create tests Implement one more enemy	Cornel Cornel Cornel	8 3 2	10 4 1	Yes Yes No	Made a start on another enemy, but did not fully implement it because it is not important at this moment. My problem, also with the other stuff I make, is that it takes half of the time creating the code itself so that it works, but the other half fixing errors and things that come up after a code review. My pull request are open a long time and that is very annoying. It also takes me a lot more time than I would like to.
As a developer I want to be sure the AudioManager and the Constants class work as they supposed to so that there are no bugs in this system.	Create tests for the AudioManager Create tests for the Constants class	Cornel Cornel	3 2	0 3	No Yes	For the Audiomanager we discussed that we would not test this, because Travis would always let the program fail because of this. The constants class is tested!
As a player I want to have the new powerups spawned in the game so that I can play the game with the new powerups.	Spawn the new powerups randomly Refactor the powerup creation with weighted sets	Cornel Cornel	2 2	1 6	Yes Yes	The refactoring and discussing the best way to do it took a lot of time. After the refactoring the spawning of new

Main Problems Encountered

Problem 1

Description: The LoganSquare JSON (de)-serializer needs additional sources to serialize classes to JSON and write them to files. It was relatively easy to let everything compile successfully locally, but very difficult to get Travis compiling these new sources. As a result Travis failed very often until a solution was found.

Reaction: Normally it's unacceptable if Travis fails many times in succession, but because of the issues with LoganSquare we tolerated it this time.

Problem 2

Description: We found out that there is a bit of a gap in programming skills in our team. Some people are very good and make almost perfect, effective and efficient code, but some don't do this automatically. This way it takes very long for these last people to add new features and they are doing longer to write the same amount of lines code.

Reaction: In our team this is tolerated and the ones who are very good help the others create better code, and they learn from it. But the Teaching Assistant is probably not happy that some write more code than others.

Problem 3

Description: Testing sounds was very annoying. It worked fine when just testing in IntelliJ, but when we tried to push it and have it check by Travis, it would always fail. We tried some 'hacks' like Thread.sleep after every sound but it would not do it.

Reaction: We decided not to test the classes that have to do with playing sounds. We will now keep Travis happy, but our testing coverage is a bit lower.

Adjustments for the next Sprint

Until now every sprint we work very hard on new features, but not hard enough on the features which have to be done by the assignments every week. This way we get a low grade sometimes for the assignments. It's of course a pity that we can't implement features that we would love to add ourselves, but the assignments are more important and have to be done.
The missions and upgradables are a very nice feature to have and we really want that feature in the game, but it is too much to implement perfectly in one sprint in the game. The current version works, but it doesn't look very well. Next sprint we planned to finish the feature completely.