# Assignment 2

## Game: Temple Run

by

## Group 9

Course: TI2206

Software Engineering Methods

Maarten Sijm
Robin van Heukelum
Mathias Meuleman
Mitchell Dingjan
Maikel Kerkhof

TA: Jurgen van Schagen

Teacher: Alberto Bacchelli

# Exercise 1: 20-Time

Requirements:

- The user should be able to save the progress of a game.
- The user should be able to load the progress of a game.
- The data storage file will be encrypted/decrypted upon read/write actions.

All data will be saved in an encrypted JSON file.
List of data that can be saved:

- Player name
- Number of coins collected
- Settings
    - Sound value
    - Other settings (future features)
- Bought items (future features)
- Personal highscore(s)

Some of the bullet points in this requirement are impossible to implement because the back-end does not yet support these items. They are marked with '(future features)'.
Two classes were created to implement these features, one for the parsing of the JSON and one for writing JSON to a file. We think that it is overkill to show the CRC cards here, as their only collaboration is with the GUI.

We got most of this feature implemented over the week. Only the encryption/decryption of the JSON files has not yet been implemented. The implementation of the JSON parser and writer took more time than expected, so the encryption/decryption will be passed on to the next iteration. We were able to surface though, we didn't go down like a dead dolphin [red: reference to first guest lecture about Scrum].

# Exercise 2: Your wish is my command

## Take a good look at the textures

The textures feature (*#53*) was almost done last week. However, there were a few problems, which we had to overcome this week.

- Travis' console log became larger than 4MB because of unsupported 3D features on Travis' display. This made builds fail.
    - We have overcome this failure by adding a condition to rendering the objects, namely:
      `if (Platform.isSupported(ConditionalFeature.SCENE3D)) {`
      There are still some warnings in creating the SubScenes and the Camera, but these warnings can be happily ignored and do not create the log to overflow.
- The textures generate huge amount of lag.
    - By adding cache hints to the object rendering, this lag has been partially fixed. There is still some lag, but on a proper PC the game is playable.

## Discuss how to improve the design of the GUI implementation

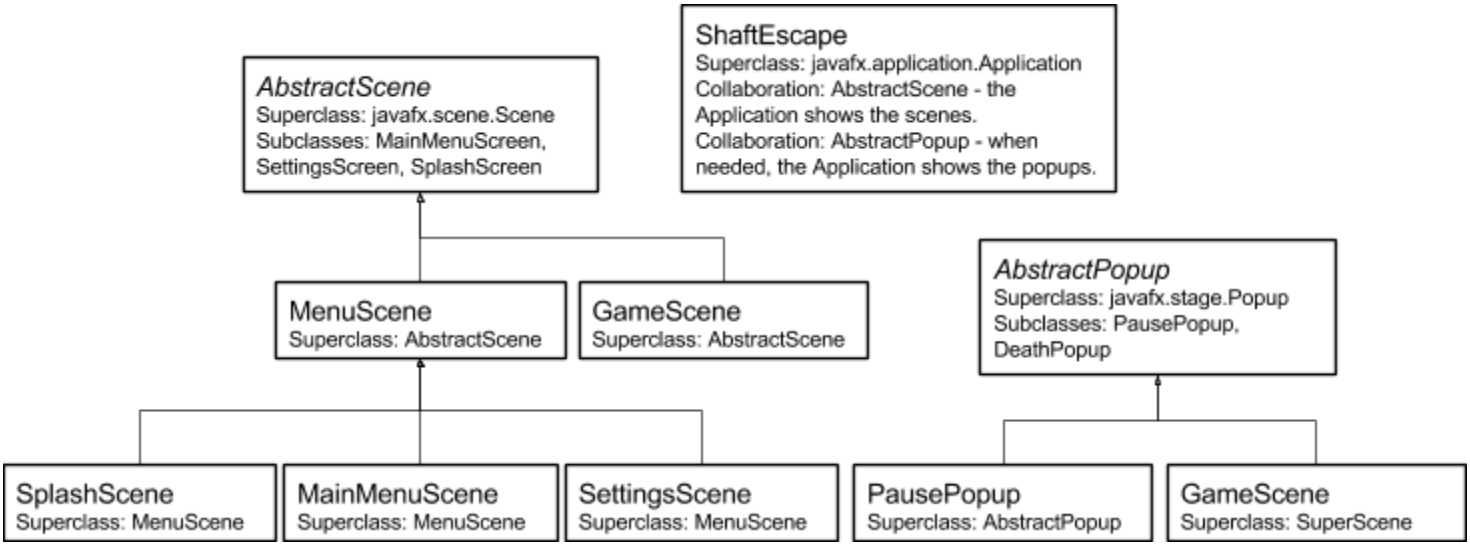
### Discussion

Our assignment was to rethink our GUI implementation (*#95*). There was no hierarchy yet in our scenes, but we had to create that.
We were thinking about creating only one Application (before, SplashScreen extended Application and called the application start methods from the other Screens) which controls which scene is viewed.
All scenes will extend from a new class, AbstractScene, which sets some standards on how to show the scene.
From AbstractScene, a MenuScene and GameScene will extend.
The MenuScene will be the ancestors for the menus, which are currently SplashScreen, StartScreen and SettingsScreen.
The GameScene will display the game in combined 3D and 2D, as GameScreen is currently doing.

Also, the Popups needed some rethinking (*#92*). There was only one PopupMenu class, which had two static methods that created the Popups. The methods were really long and needed refactoring.
We are thinking about an AbstractPopup superclass, which extends from the JavaFX Popup, from which two subclasses extend that represent the two different types of Popups in our game, PausePopup and DeathPopup.

During this discussion, we have used CRC cards. Because last time a photo was not very visible, we have drawn our CRC cards similar to a class diagram.

### CRC cards



# Short documentation on our extra implemented features

As extra features, we only had bugfixes and configuration updates. We will list the most important ones:

- *#81 Pressing Esc twice will crash the game* has been fixed by adding an extra condition to the input handler.
- *#82 On high speed, Player doesn't collect all coins*
    - This fix took a bit more complex method to solve. We added a small loop to the collision handler, which takes the Entity that is moving and checks for collision on every position on their path. In this way, coins will always be picked up.
- *#89 Player name* is a feature in the GUI that makes it able to enter a name for your Player character. Also, previously used names can be chosen from a list in the LoadGameScene screen.
- *#93 Add diversity in track* was done by creating more and smaller TrackParts. The TrackParts were already used in a random order, but because there are more TrackParts, the track is now more diverse.
- *#94 Personal highscore* was also fairly simple. All that was needed was an extra field in the State class, and some methods for updating/retrieving it.
- *#99 Static Analysis Tools review* was quite a hassle.
    - It was hard to get to get our definitions for checkstyle in the two different IDEs that we use. We had created a separate checkstyle for the tests folder, because we do not need JavaDoc in our tests. In Eclipse, this goes smoothly, but in IntelliJ, no two separate checkstyle files can be active. In our pom.xml, we could create the two separate definitions, luckily.
    - Also the PMD was a bit of work to transport from Eclipse to IntelliJ, but eventually we had created a PMD definitions file that was usable for both IDEs.
    - In this branch, all remaining issues were directly resolved, so that the master would become clean, to be kept clean by everyone.

There is one feature that we still did not yet have implemented yet because of an issue, namely soundtrack. The feature (still) works, but as Travis' builds still fail we did not yet want to merge the branch into the master. We researched the source of this problem again, just like last week. We applied various more strategies in order to make the build pass, but unfortunately all in vain. At the beginning of the sprint, we asked our TA to have a look at it and one of our team members sent a mail which contained information about the issue. We did not yet receive a reaction, therefore we decided to shift this feature to the next iteration again.