Modeling Mahjong on an Android System

Grayson Cash & Migao Wu
BS in Computer Science

05/03/2014

Advisor: Kenneth Blaha

http://en.wikipedia.org/wiki/File:Android_robot.svg

Introduction

- Traditional Chinese Mahjong game (not matching version)
- Targeted for Android 4.0+
- Programmed in Java and XML

http://project-hongkong.blogspot.com/2010/05/game-night.html

Why we chose this?

- Individual interests
- Proliferation of Android devices
- Practical problem:
  - User group was too limited
  - Chinese vs English-speaking

http://yourstory.com//wp-content/uploads/2013/06/China.jpg

“Real” Mahjong?

Traditional Chinese Mahjong:

- 4 players required
- 124 tiles consist of 4 different suits which are “Bamboo”, “Money”, “Circle” and “Wind”
- Winning hand pattern: at least 1 pair

http://project-hongkong.blogspot.com/2010/05/game-night.html

- Each player takes turn starting from the dealer to draw and discard a tile.
- 5 functions in the game
  - eat

  - double

  - triple

  - win, skip

Requirements

Functional:

- All five functions
  - eat, double, triple, win and skip
- Allows user to play against three “bot” players
- Allows user to win the game once they have a winning hand
- Representing the game in a 2D environment

Requirements

Non-functional:

- Targeted for Android 4.0+
- Different hardware speeds, OS versions, screen sizes and densities
  - Focusing on our own personal devices

http://www.android.com/versions/kit-kat-4-4/

Use Case: DoubleButton

Name:	double button
Actor(s):	Bots and human player
Precondition(s):	The human player has two same tiles in hands.
Description:	If the bot to the left of the human player discards a tile which is the same as the two same tiles in the human player's hands, then the human player could choose to "double" that tile or "skip".
Basic Flow:	1. The human player got two same tiles in hand. 2. The bot to the left of the human player discards a tile which is the same as the two same tiles in the human player's hands. 3. Then the human player could choose to chick on the "double" button to "double" that tile or click on the "skip" button to "skip".
Output:	"double" with that tile, "doubled" tiles will be showing on the very right or "skip".

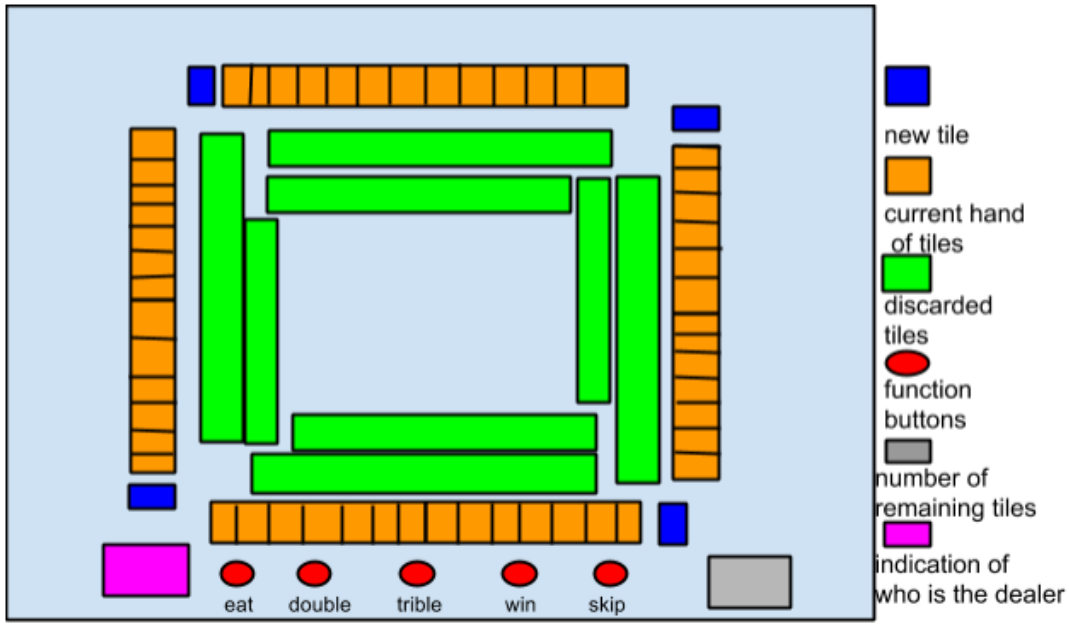
Use Case: WinningButton

Name:	winning button
Actor(s):	Bots and human player
Precondition(s):	1. The human player has a pre-winning hand of tiles. 2. Either the bot to the left of the human player discards a tile that the human player can win with or the human player picks up a winning tile by himself/herself.
Description:	The human player clicks on the winning button to declare winning once the word on the button shows up or use the skip button to skip.
Basic Flow:	1. The human player has a pre-winning hand of tiles. 2. Either the bot to the left of the human player discards a tile that the human player can win with or the human player picks up a winning tile by himself/herself. 3. The human player clicks on the winning button to declare winning once the word on the button shows up or use the skip button to skip.
Output:	The human player wins the current round with a message displaying on the screen.

Design

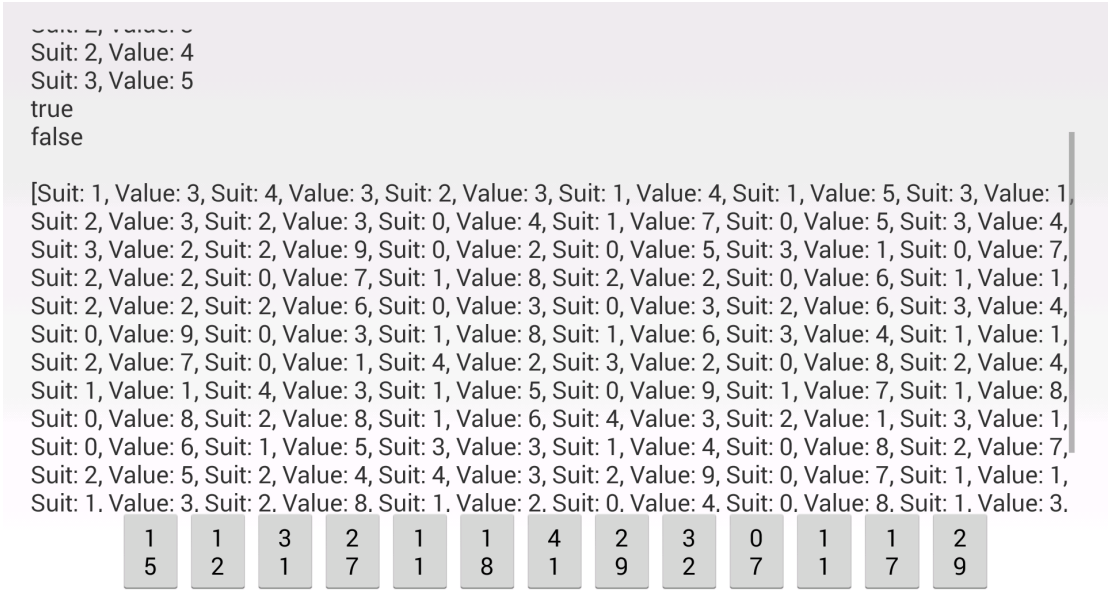
- MVC design
- Interface design

Original Interface Mockup



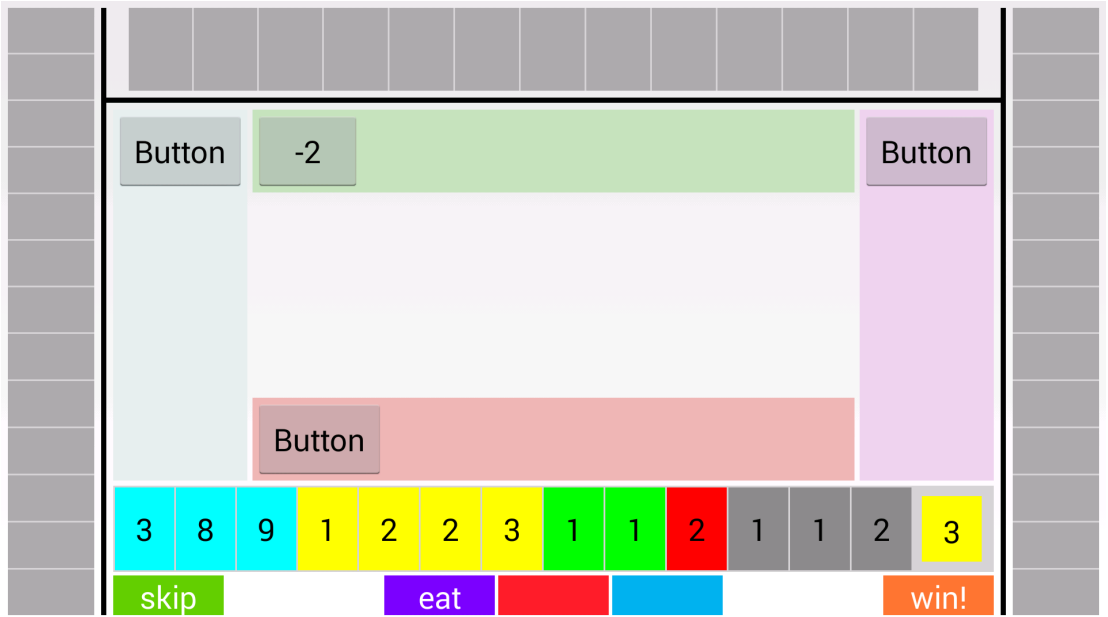
Design - UI

- After 1st tech presentation:



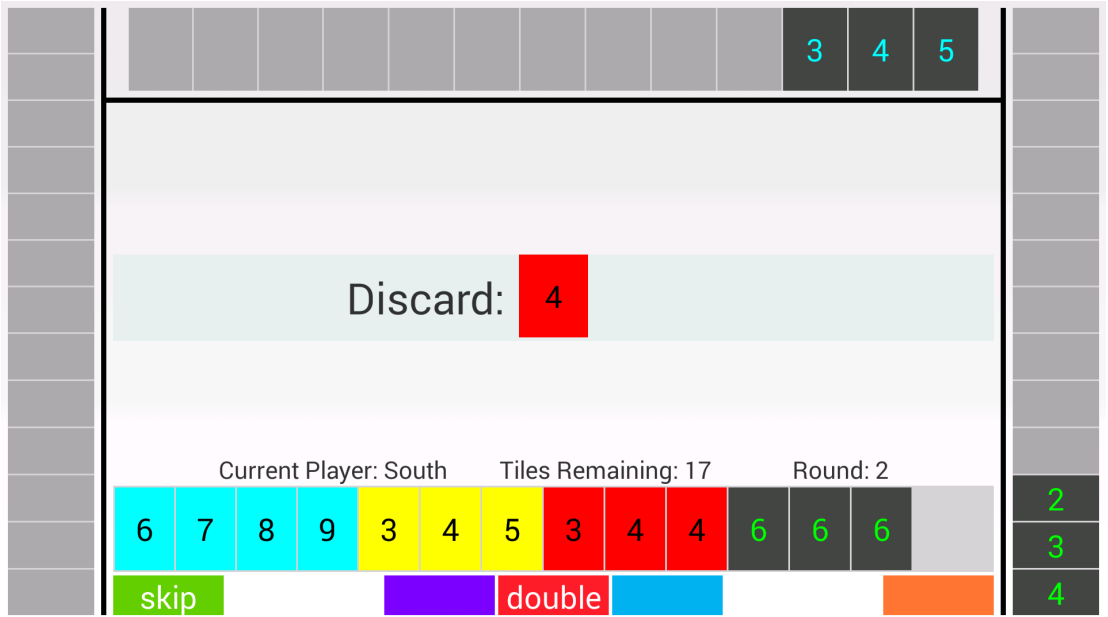
Design - UI

- After 2nd tech presentation:

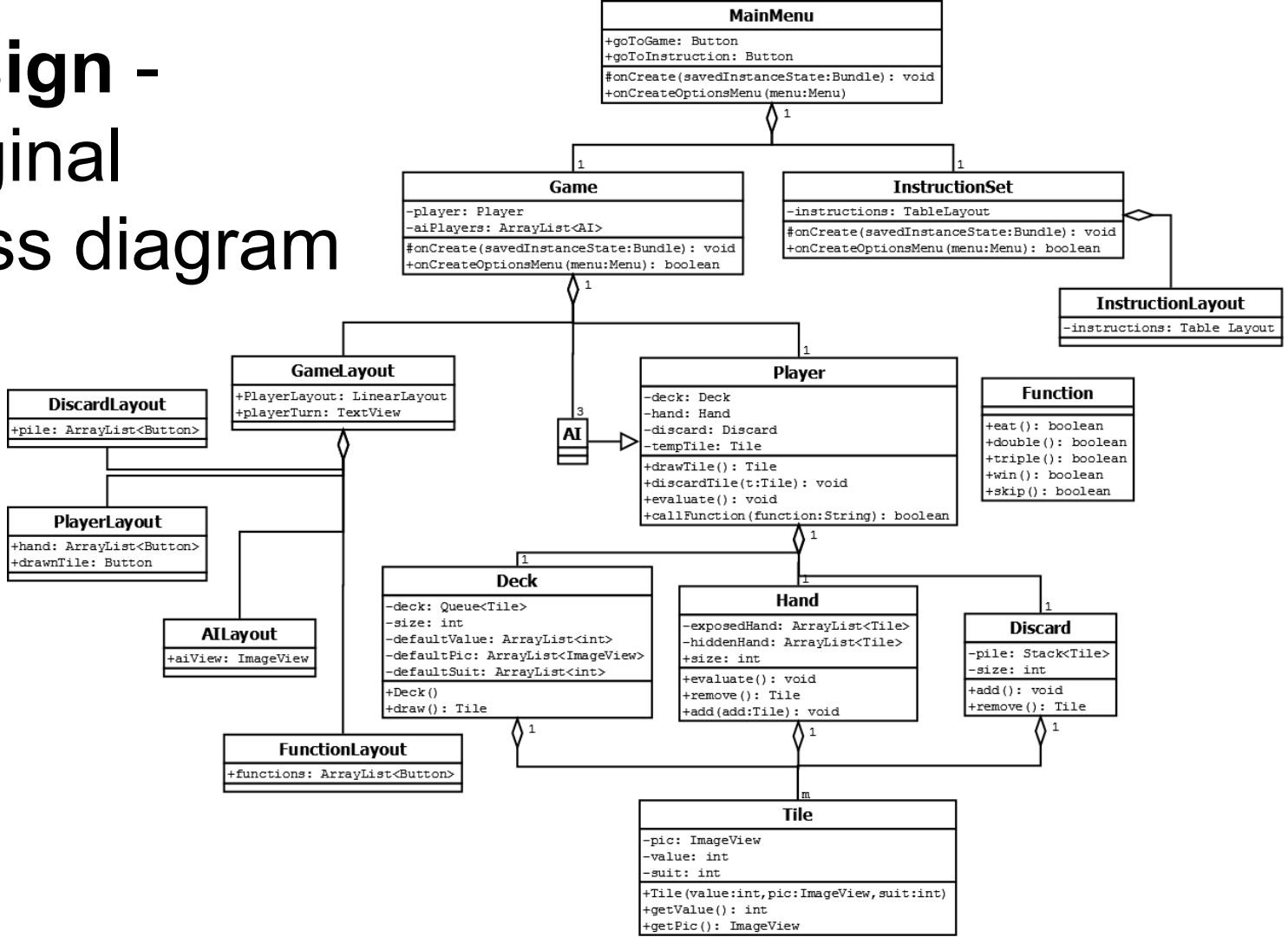


Design - UI

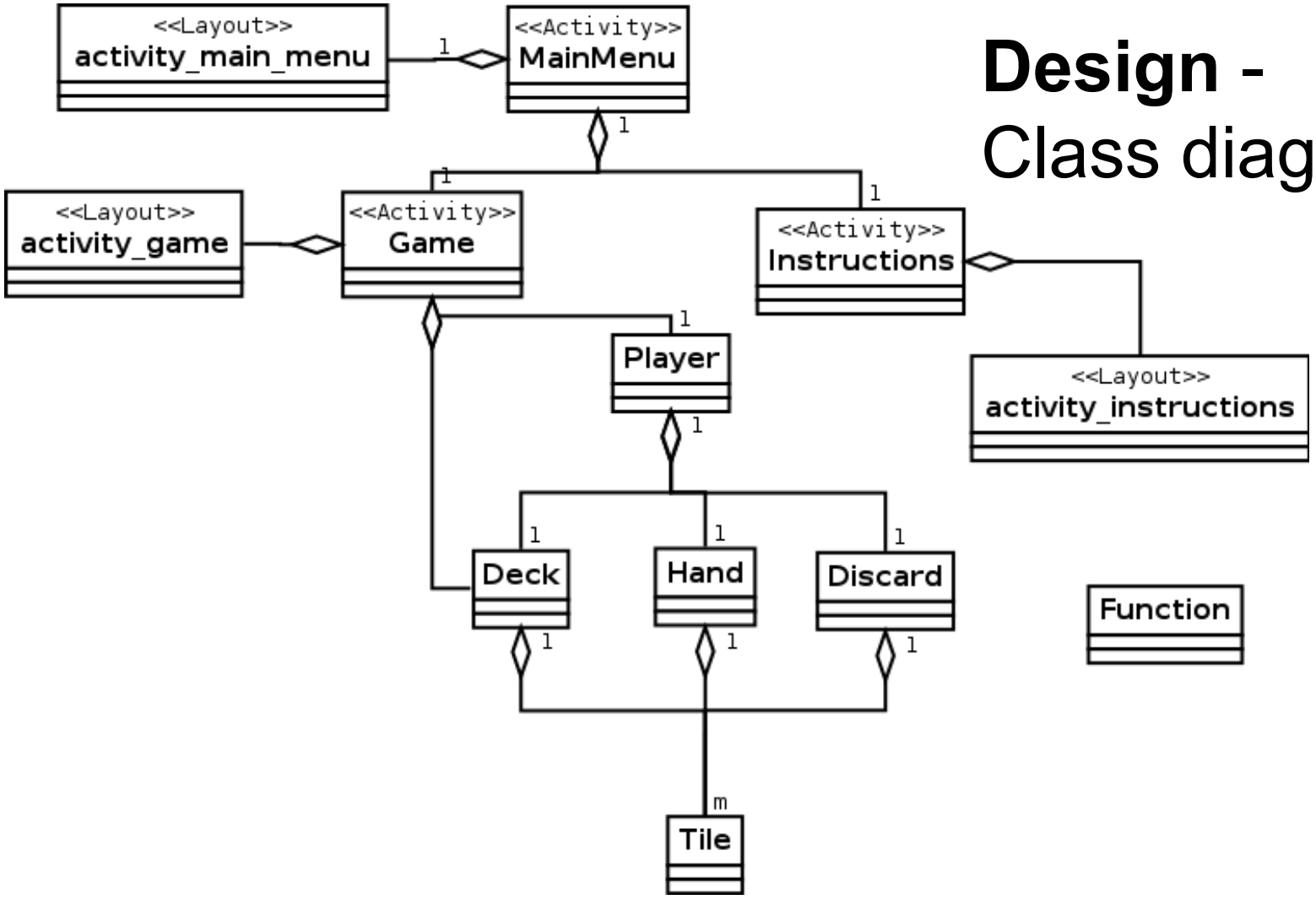
- Final game interface:



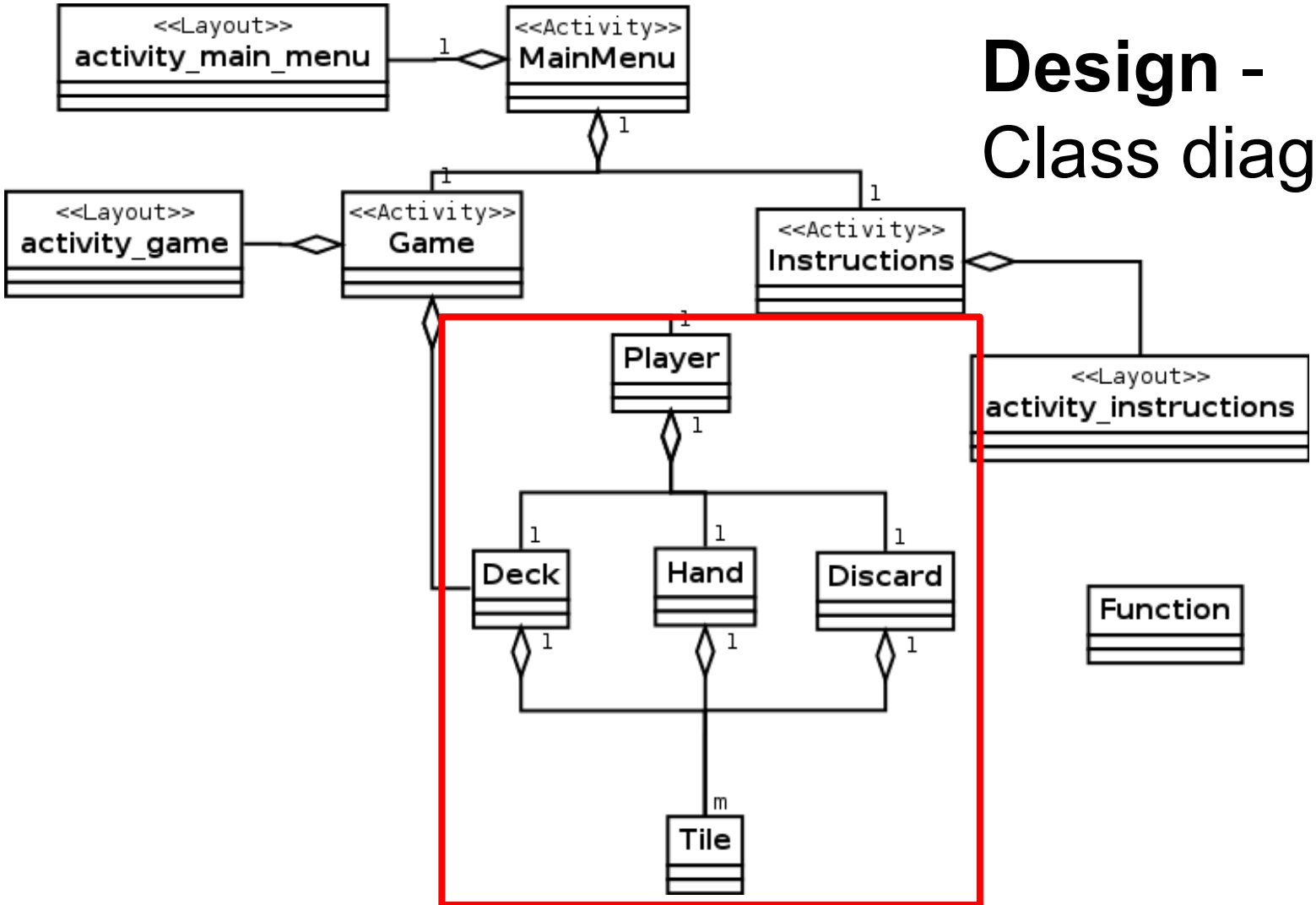
Design - Original Class diagram



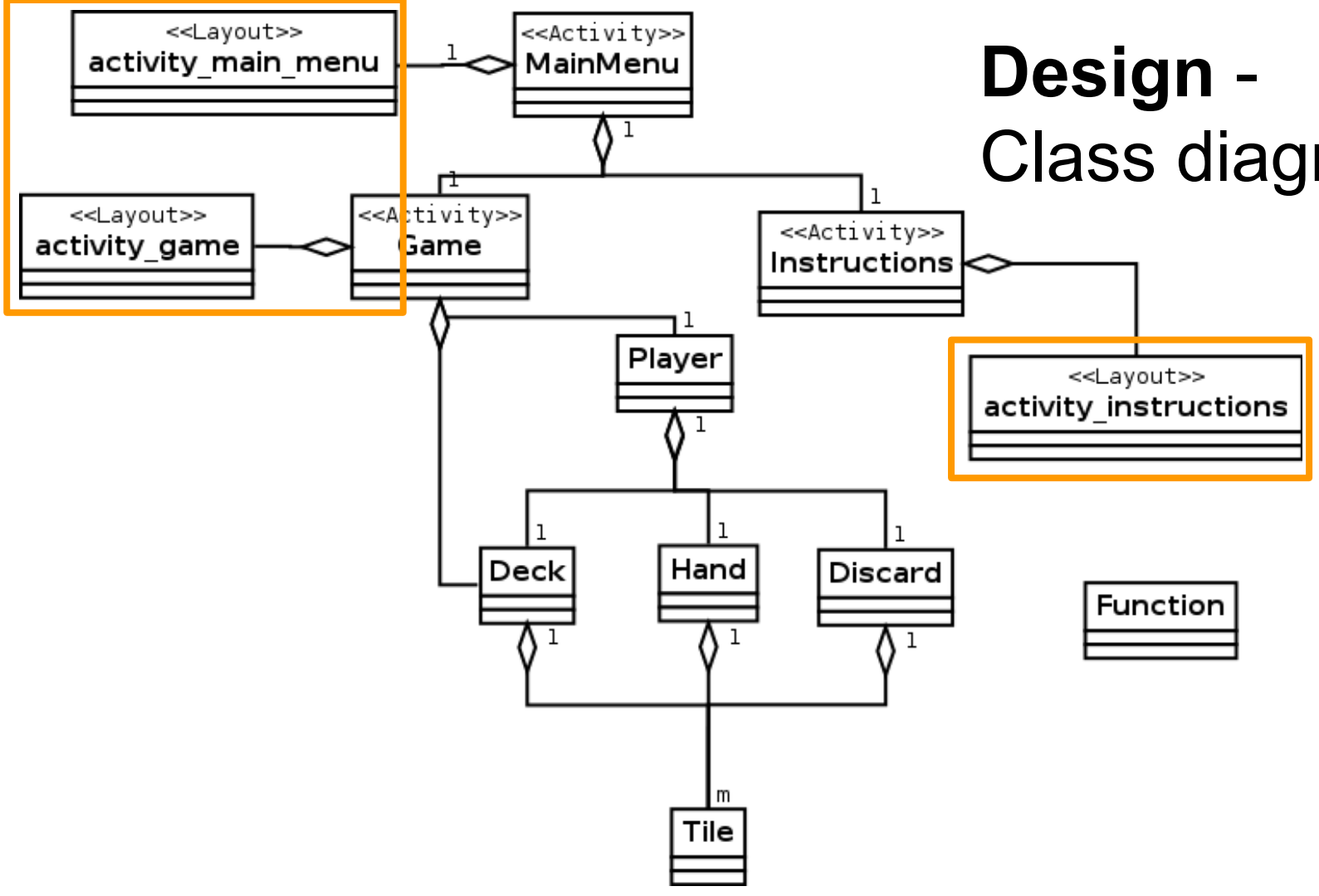
Design - Class diagram



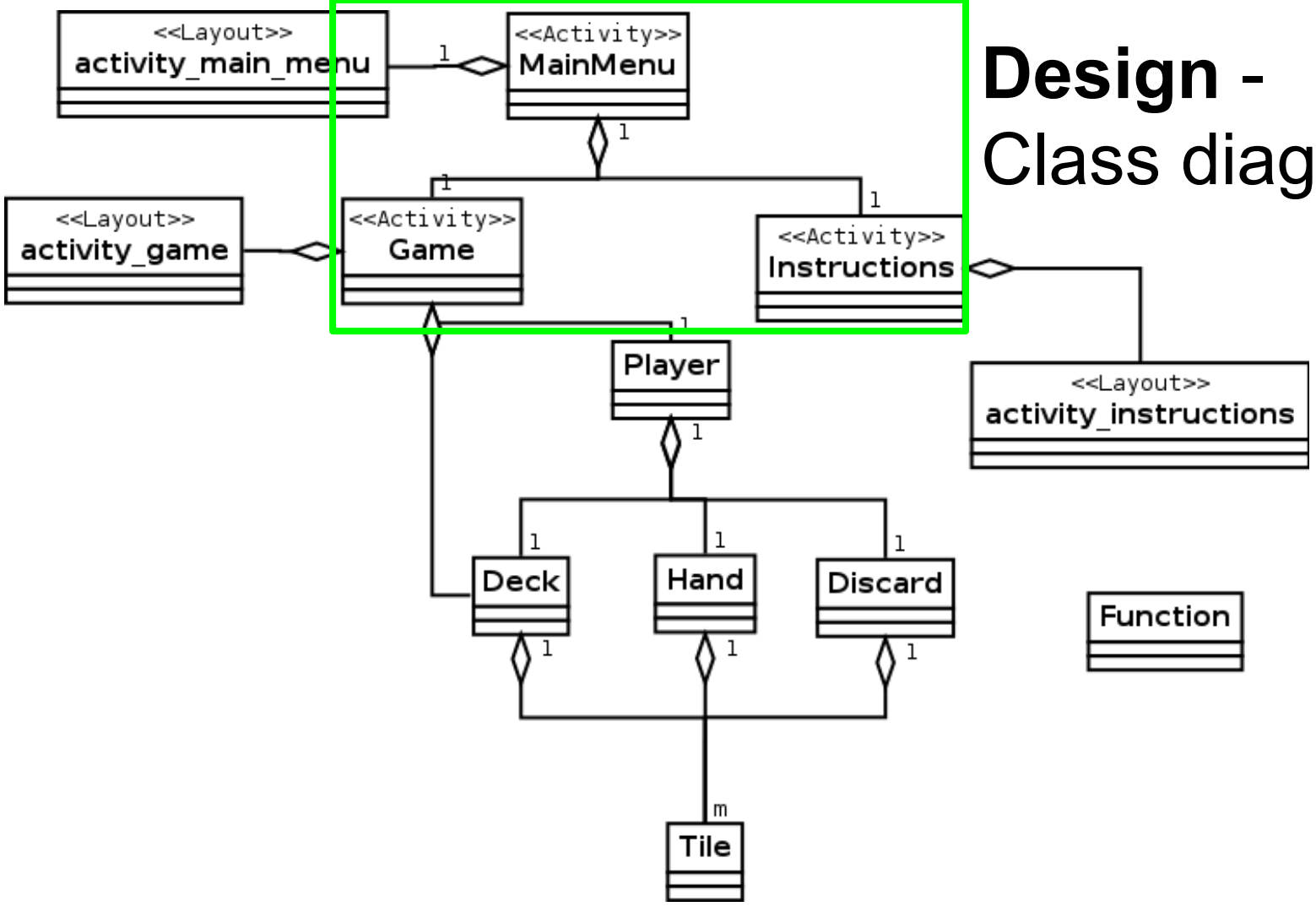
Design - Class diagram



Design - Class diagram

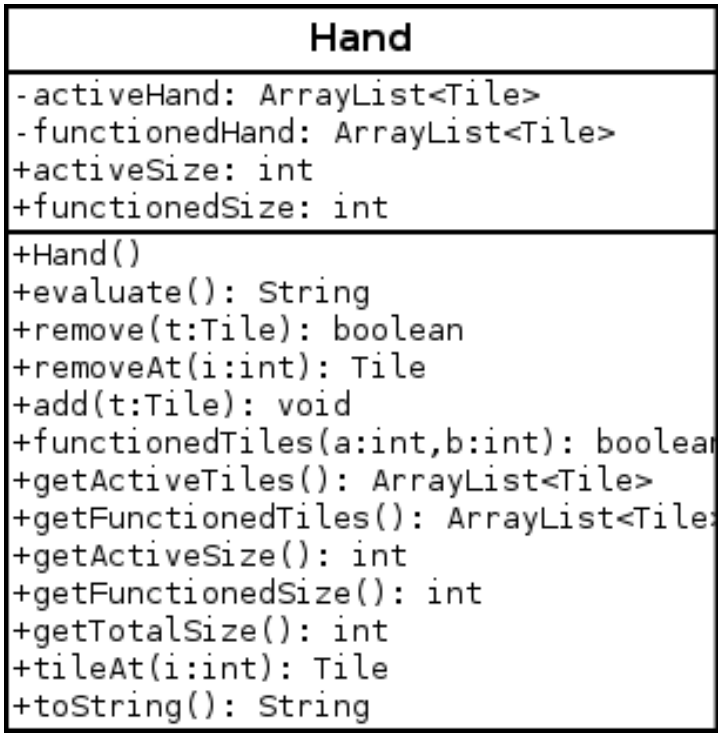


Design -
Class diagram



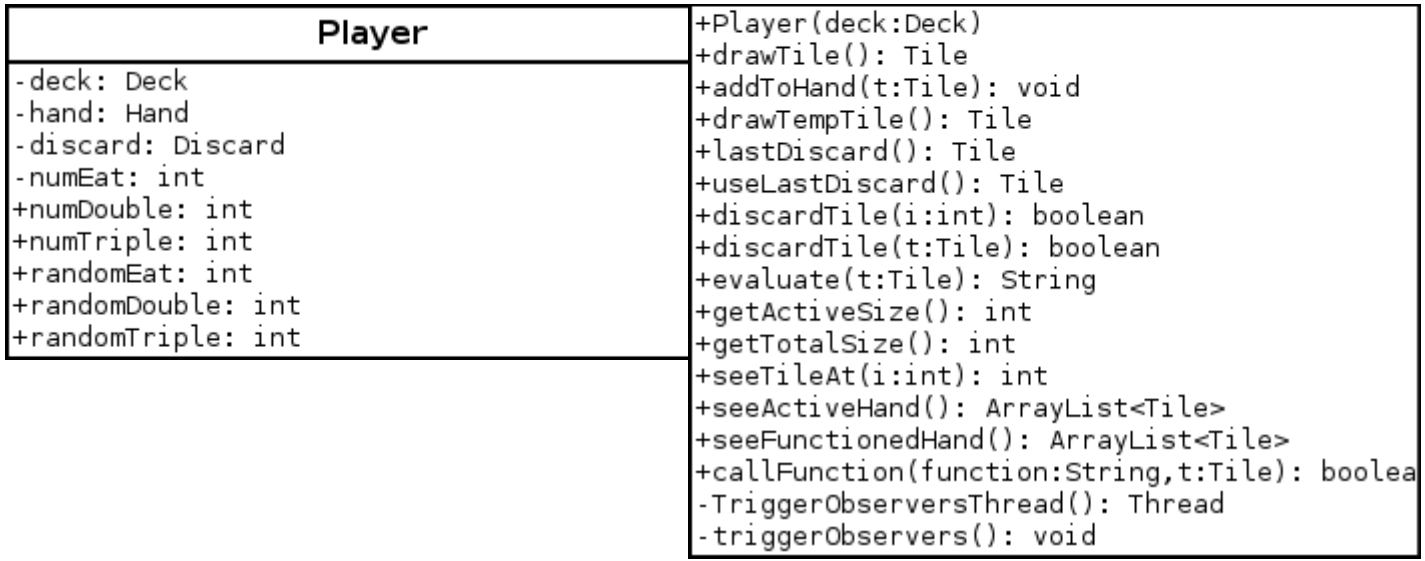
Design - UML diagram

Hand



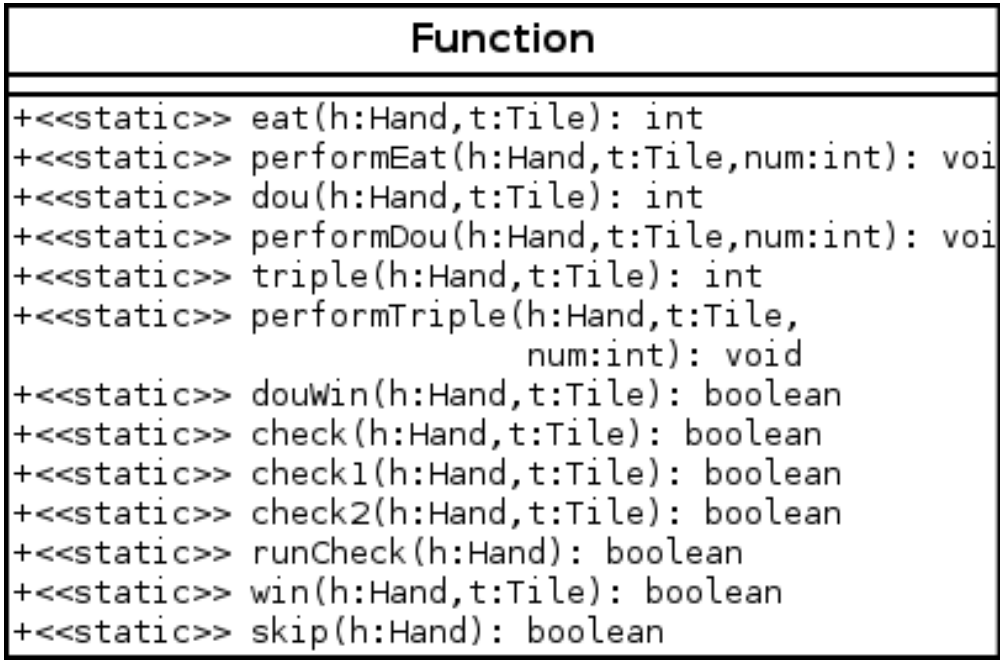
Design - UML diagram

Player



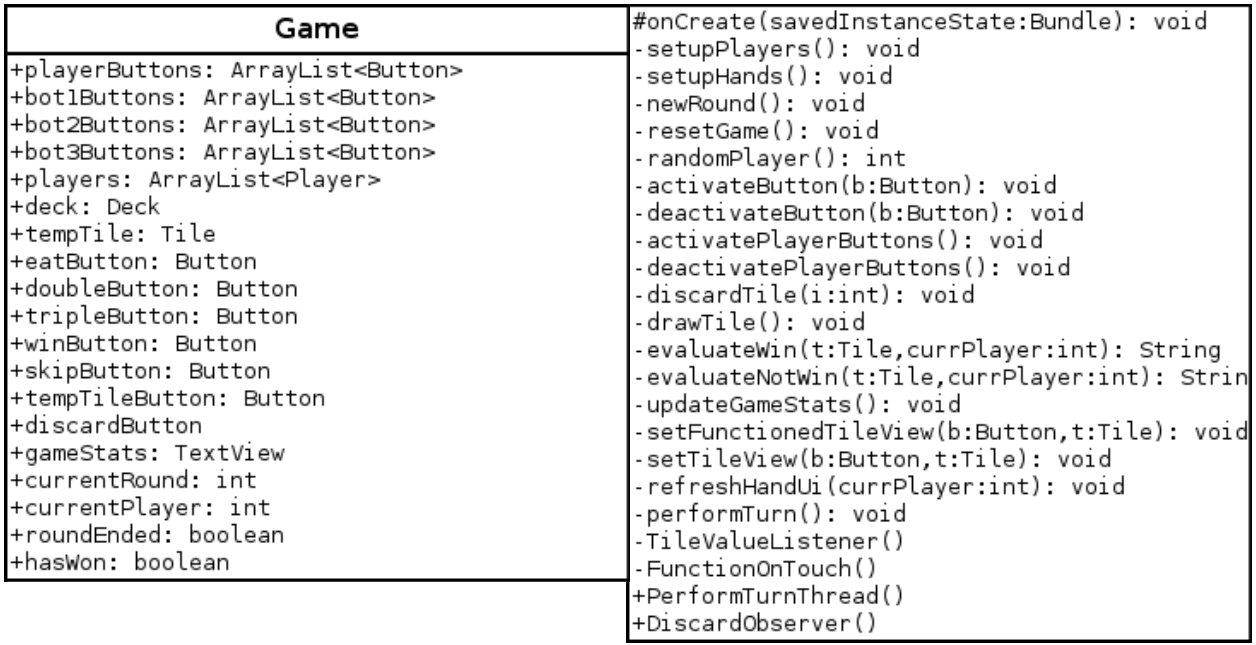
Design - UML diagram

Function



Design - UML diagram

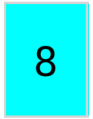
Game



Design change - Tile

Original Tile design: 

Attempted to import pictures and use them to represent a tile.

Final Tile design: 

Colors and numbers to represent suit and value of the tile respectively.

Implementation

Tools: Android Development Environment

Language: Java and XML

Testing: JUnit testing & playing actual game on phones

Debugging: LogCat was extremely helpful

Issues

- Image scaling
- Screen densities and sizes
- Exiting the application
- Getting GitHub to sync between computers
  - import from repo, check build paths
- Functions not working

http://developer.android.com/design/index.html

Issues

- Later issues with phone interfacing with Eclipse
  - Reinstalled drivers and APIs
- Positioning elements with a `Linear` or `RelationalLayout`
  - Looked into `Fragment`, decided against
- Stopping `RoundThread`

http://www.wordans.co.uk/wvc-1323996506/wordansfiles/images/2011/12/15/112704/112704_340.jpg

Things we couldn’t get done

- Not being able to display all discarded tiles at once
- Deployment to Google Play

http://www.egotailor.com/product/10612/images/10612design-1.jpg

Future Work

Potential Features:

- Money points system embedded
- Different levels of bots
  - Easy, moderate, expert
- Online game

Demo

http://www.bennugd.org/sites/default/files/androids.png

Questions

http://www.oneclickroot.com/wp-content/uploads/2012/12/question-android.jpg