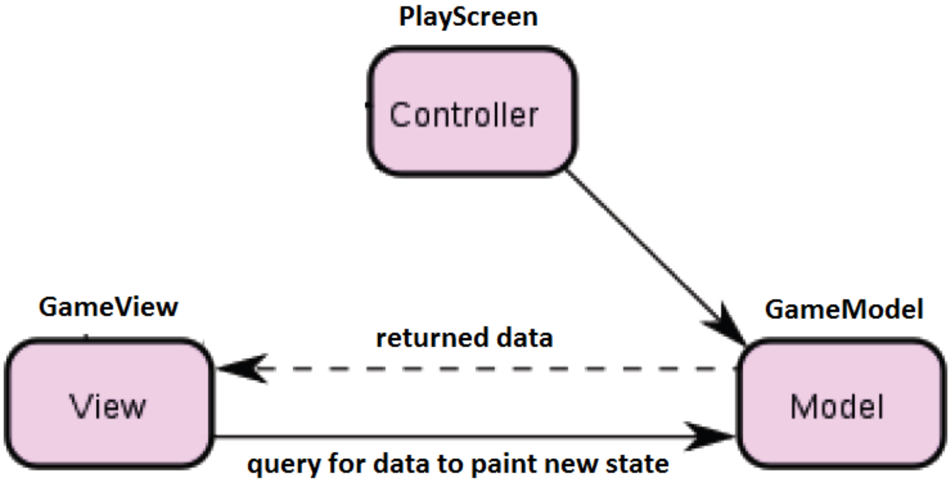
System design document for “GradiuZ”

MVC
Vi tänkte försöka använda oss av designmönstret Model-View-Controller för själva spelfönstret. Vi har läst att det ska passa bra till spelutvecklingsprojekt, dock är vi osäkra och får gärna feedback på denna punkten.

komponenter i spelet
Model, View, Event-Based Controller, (Spaceship, obstacles, graphics, weapons, background)



Klasser i komponenter

- **StartGameScreen** - The starting screen that shows a few options likes “New game”, “Options” etc..
- **GameOverScreen** - The screen that shows up when the player/spaceship gets destroyed or dies.
- **PauseGameScreen** - A screen that pauses the game in case of a phone call or some other more important event.
- **HighScoreScreen** - The highscores are shown.
- **OptionScreen** - Different options, e.g. mute on/off

- **PlayScreen (Event-Based Controller. The “Controller” part of MVC**) - The background screen that the game takes place at. This class contains the logic of the game and extends Androids Activity class and implements some Listener interface to be able to handle events. These events notifies the GameModel class to change the state of the game.
- **SpaceShip** - The main player in the game. The object that the player moves around on the screen (PlayScreen).
- **Obstacles** - A superclass for objects in the game that are solid and can destroy the spaceship.

- **Stone** - A subclass to **Obstacles** that describes some kind of static object that the spaceship can crash into and get destroyed.
- **Monster** - Another subclass to **Obstacles** OR to **Stone** that can either move around on the playing screen and/or shoot the spaceship.
- **Weapons** - A superclass for all the different types of weapons that might be used by the spaceship to shoot obstacles.
- **Projectiles** - Projectiles shot by the Spaceship, or shot by monsters.
- **GameModel (The “Model” part of MVC)** - Handles the states of objects like coordinates and interactions of “objects” in the game.
- **GameView (The “View” part of MVC)** - Includes the “Game loop” of the game that repaints the continuous new states of the game as the Spaceship moves forward. The GameView retracts the data from the GameModel class to “know” how to present the objects of the game.