Split screen multiplayer

Group 66

This document describes the design using responsibility driven design and UML. Starting with the drafted requirements.

Functional requirements

Must haves

- The basic game mechanics must work the same as in single player
- Two levels must be generated, each with their own cannon
- Both levels must be the same, in terms of ball placement and wall detection
- When either player wins or loses, the game is over

Should haves

- The two roofs should move independently
- Each player should have their individual score

Could haves

- Each player could have their individual shooting timeout

Won't haves

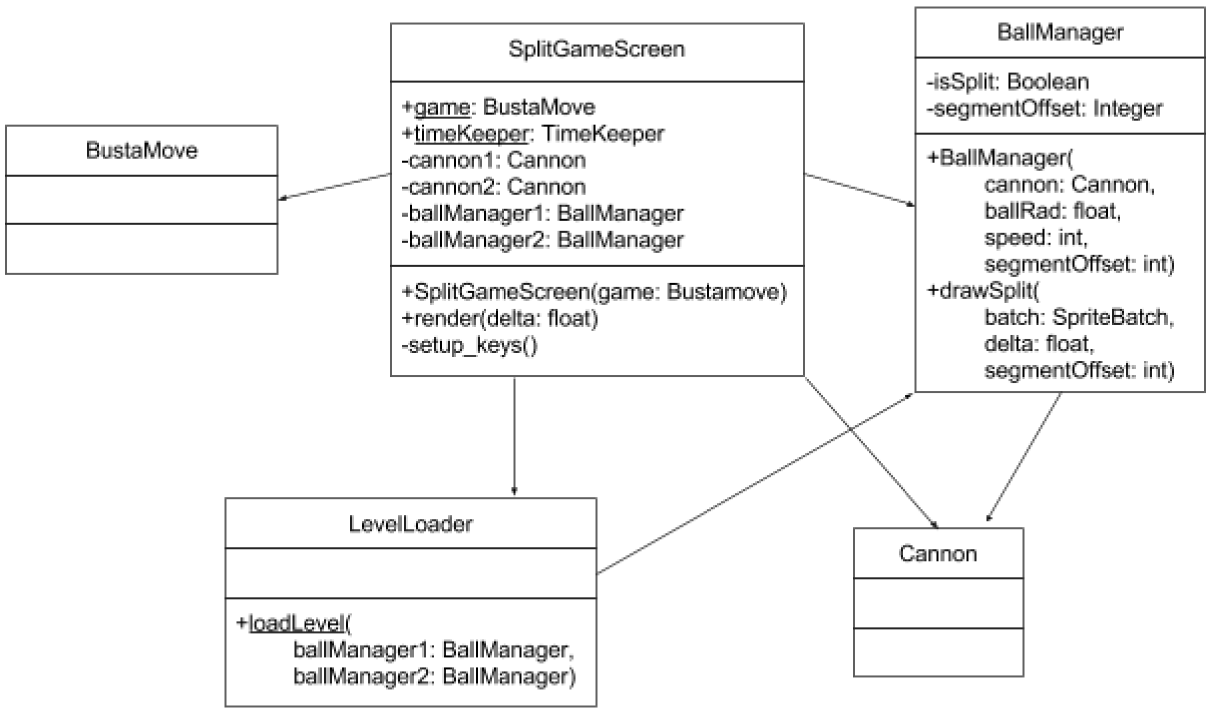
- The code won't (for now) be extended to support 3 or 4 players

Non-functional requirements

CRC

Class	**Responsibility**	**Collaborations**
SplitGameScreen	Defines two play areas, calls multiple other classes	All other classes

UML



Technical design

The SplitGameScreen will work mostly the same as the regular GameScreen, except that it has two copies of several objects. There will be two ball managers, two cannons and two roofs. Each of these will work independently. When either player wins or loses, the game is over.

The shooting timeout and score will be shared between the two players (because of deadline).

Level loader has to load the same level twice.

The ball manager has to handle an offset for every new screen.