**Software Architecture Specification**

March 20, 2017

Mango WEB (Caleb, Julian, Collin)

1. System and Architectural Context

1.1. Rationale

For a single-player smartphone game, only a single device is required per copy of the application. . Therefore, the most architecture describes the hierarchy of classes and flow of control. A modular view of the architecture is most appropriate for this game since there aren’t any other devices (i.e. servers, databases, etc.) with which the application must communicate. .

1.2. Scope

The extent of the system is a single-player mobile game for Android and/or iOS. The game must be compatible with the two mobile operating systems. This system does not depend on or interact with any other devices, systems, applications, databases, or servers other than the device that the game is running on.

The stakeholders of the game are the users, developers, and testers. Users demand a functional, enjoyable, and intuitive game. Developers must understand the architectural requirements, functional requirements, and design goals. In addition, they must complete the program within the allotted schedule, cost, and quality goals. Testers must understand the same specifications as the developers in order to create suitable binary tests that align with the functional requirements of the game.

1.4. Behavior

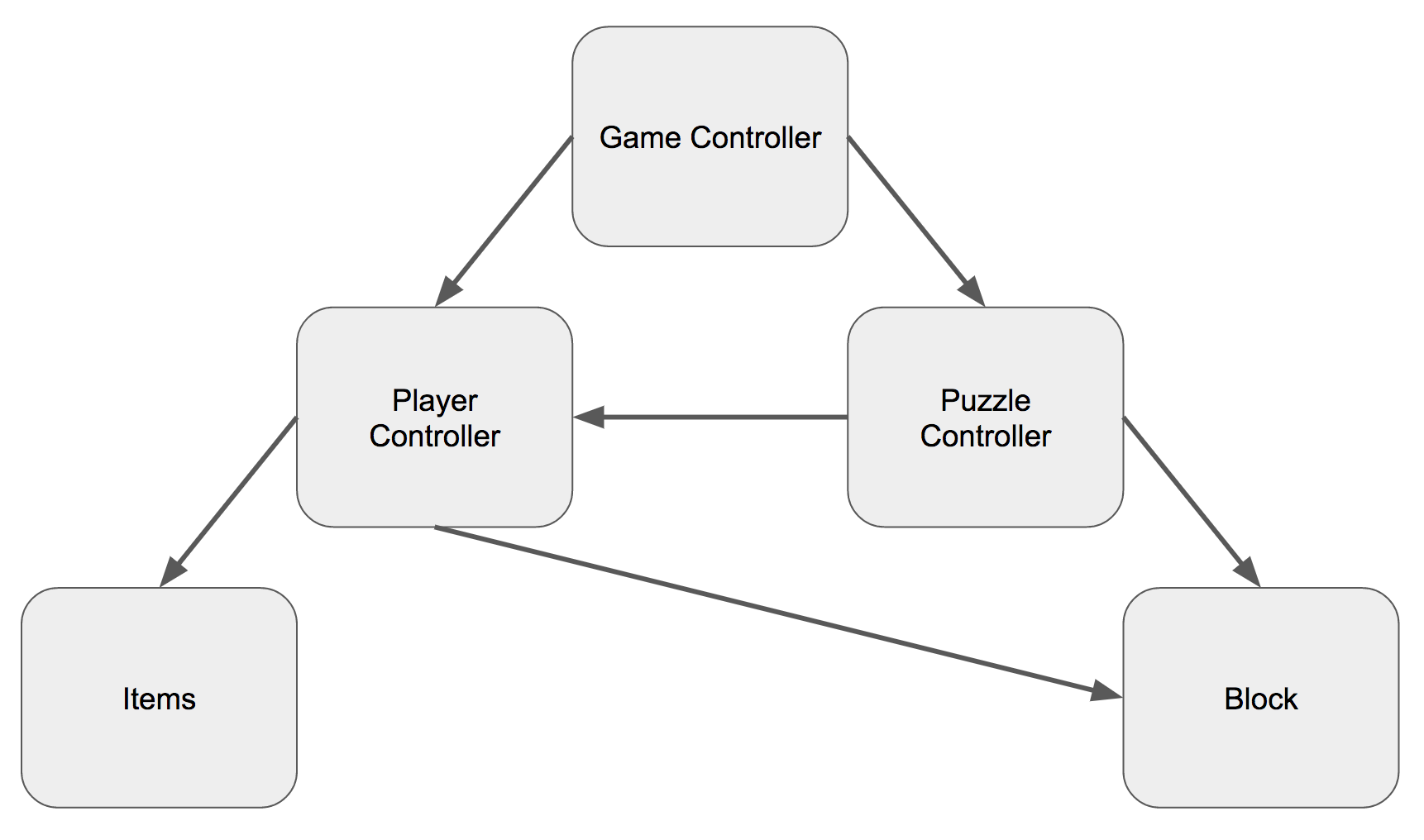
Since there are no external devices to communicate with, all the functionality will be handled within the smartphone ecosystem. Different menus and actions will be available to the user at any point in time and represented on the display. The user will be able to interact with these actions or menus by tapping or swiping the appropriate section of the display. Interacting with menus will result in the display of a list of options. For the main menu, the list will comprise of graphics options, saving the current game state and other options as we deem them necessary. The inventory menu will consist of a list of currently held items. Interacting with the character in the game (visible on the display) will result in movement of the character. Interaction with objects on the screen will direct the character in the game to interact with the object. If the object is part of a puzzle, the display will change to a puzzle view, where the user is shown available objects to place in the puzzle. Once the puzzle is solved or the user chooses to quit, this display will revert back to a 2-D character view. If the object is an item, this item will be added to the inventory (‘inventory menu’ button visible to the user at all times, along with the ‘menu’ button).

Touches on the screen will be converted into a position on the screen with a velocity. The position will be used as a point, and the velocity will be used to interpret a swipe when necessary.

2. Architecture Views

2.1. Views

Our project consists of a single unit that functions as an interactive interface. A module view of the architecture is most useful. In this view, each module is a class that will determine a certain subsection of the behavior of the game and its components.



2.2. Element Catalog

GameController - The main script that dictates the overall behavior and state of the game.

PlayerController - Controls player avatar position and behavior. . Interprets input from GameController script and user interface. .

PuzzleController - When a puzzle is in play, dictates state and behavior of objects available for player use in the given puzzle. Reports ‘solved’ when finished.

Block - General class describing the behavior of the puzzle elements used to solve each puzzle. May be implemented in several subclasses to create blocks with different attributes and behaviors.

Item - Objects available to the user to pick up/use in a puzzle and in wider game world. These are used by the PuzzleController and/or PlayerController scripts.

3. Across Views Description (needed only if multiple views are included in the previous section.)

3.1. Views

List the views from Section 2. For each view, describe how its elements relate to the elements in the other views.

This section is omitted due to the use of a single architecture view.