**Boron Project Plan**

Java Sidescrolling Platformer API

Squishy Frog Studios

**Abstract**

The objective of the Boron API is to extend the Slick API to allow speedy development of sidescrolling platformer games. This document covers the required features of the Boron API and plans of action for accomplishing those features.

# The Slick2D API

Slick presents us with a helpful low-level API tool for game development. However, high-level features like physics, animation, text, and so forth are left to the game developer—presumably to allow the API to serve as many genres and game styles as possible.

However, as we make games, it is pointless to reinvent the wheel every time we want to make a new game, especially new games within a previously explored genre. Our goal then, is to extend the Slick API with genre-specific functionality that can be used in the development of games within that genre without needing to recode common concepts again and again.

# Objective

Boron is our 2D platformer API, a Java library that can be included in any Slick project to add platformer functionality to it.

Boron must be intuitive and easy to use; it must be organized so that people who had no hand in developing it will be able to pick it up and understand it quickly. Ideally, much of Slick will be covered by more friendly terms in Boron. For example, there are many different sorts of Game classes in the Slick API for different purposes (BasicGame, StateBasedGame, etc). We should choose one that is the ideal sort for the games we’ll be making, and extend it into a PlatformGame or BoronGame class for ease of use. Similarly, Slick’s Animation class is useful for creating animated sprites, but has no other functionality—a utility class. We will introduce classes that incorporate several utility classes into one, such as an Actor class that has Animation, Audio, Shape, and other elements within it. In this way, the Slick API becomes more accessible due to more focused language.

Boron must be designed to support a wide range of potential gameplay mechanics and not assume that any one concept is desirable for all games. For example, Boron must not assume that gravity always pulls down. Boron must never adjust for one game—if that feature is not desirable in future games, it will not be incorporated into the API, but will be developed within that game’s code.

Boron must be well-documented, so that developers can quickly become acquainted with the tools it presents to them. Code must be well-commented so that Javadoc documentation can be easily generated.

Finally, as with all APIs we will develop for games, Boron remains a work in progress. Feature enhancements, optimization, and addition will continue until such time as we deem that an entirely new engine is necessary to progress.

# Boron Feature Requirements

This section of the document catalogues required features not provided or provided in unfocused terms by Slick2D.