Rubikan Design

by

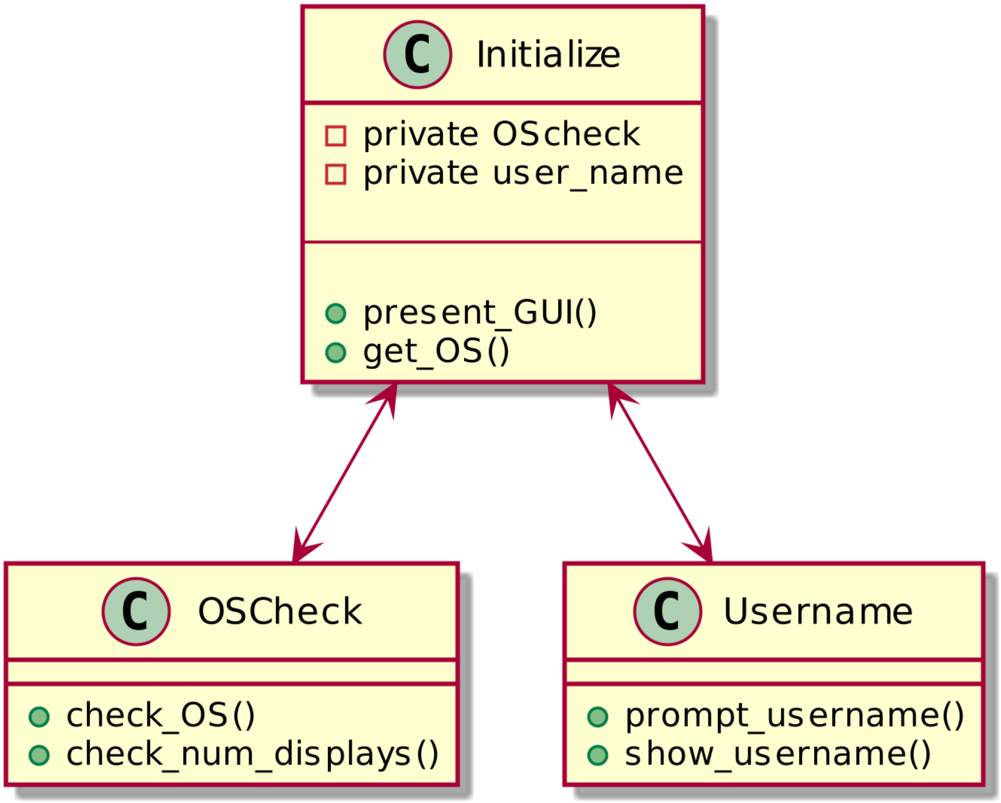
Christerpher Hunter

Nova Southeaster University

July 23, 2021

Design

Startup



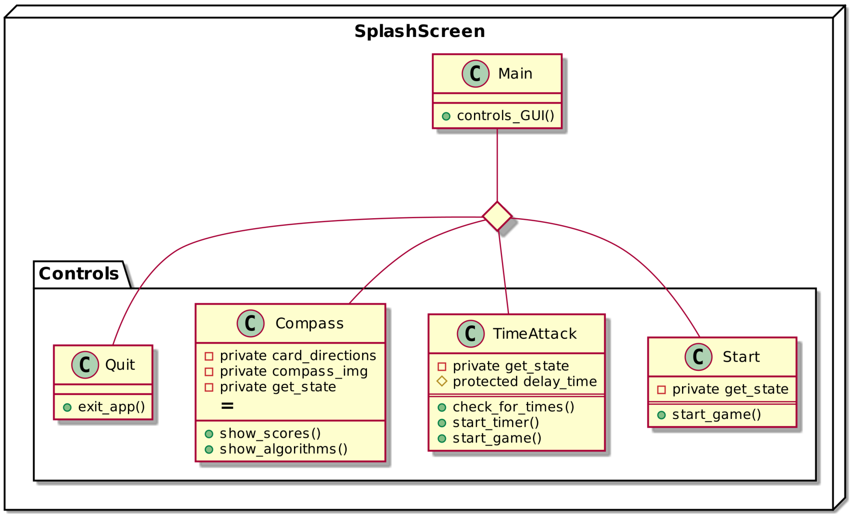
The purpose of the class Initialize is to launch the GUI window, get the username from the user (similar to an Arcade), and check the operating system. The GUI window will present the Standby State presented in the next diagram. The username function will prompt the user for a name to use when commiting scores. The operating system check will get the name of the operating system and which distribution, if necessary.

The intent of having a module to display the GUI before the module that contains the buttons and actions is due to the manner of implementation of the GUI operation. The idea here is to set up the frame and backdrop for the application first and then load the features afterward. This will allow future enhancement of the application to be simpler. The Standby State will reside in waiting and then immediately load into the Initialization class without delay. This is due to a fluid and manipulative design.

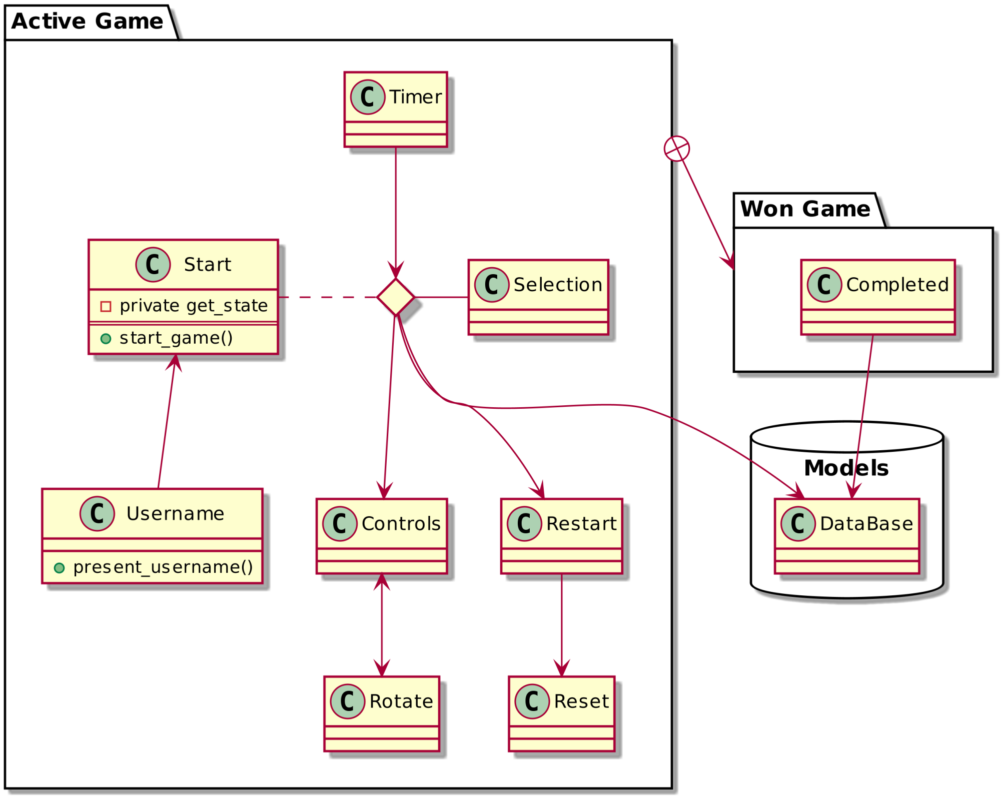
The class Username is used in several places throughout the application in order to keep track of scores. Each completed attempt will be coupled with a time and a username. The username in, every instance, will be provided by class Username. The class Username will prompt the user at the beginning of each launch of the application. Another user, at any time during the Stanby State, can add a new username. Upon the completion of a game, if there are two or more usernames, a confirmation prompt with the time and the username list will be shown. The present user will need to choose, or optionally, enter a username; this is required to enter a winning time into the database.

The class OSCheck is intended to get the name and version of the operating system from which the application launches. Should the operating system come back as a Linux operating system the class OSCheck will then attain the specfic distribution. The purpose of checking the operating is two fold. Firstly, the class MultiMonitor will communicate to the class OSCheck to obtain the operating system; this is needed for the class MultiMonitor to function as expected. Secondly, to appropriately display the application and its popups the manner in which frames are displayed must be ascertained and is directly related to the operating system the application launches from.

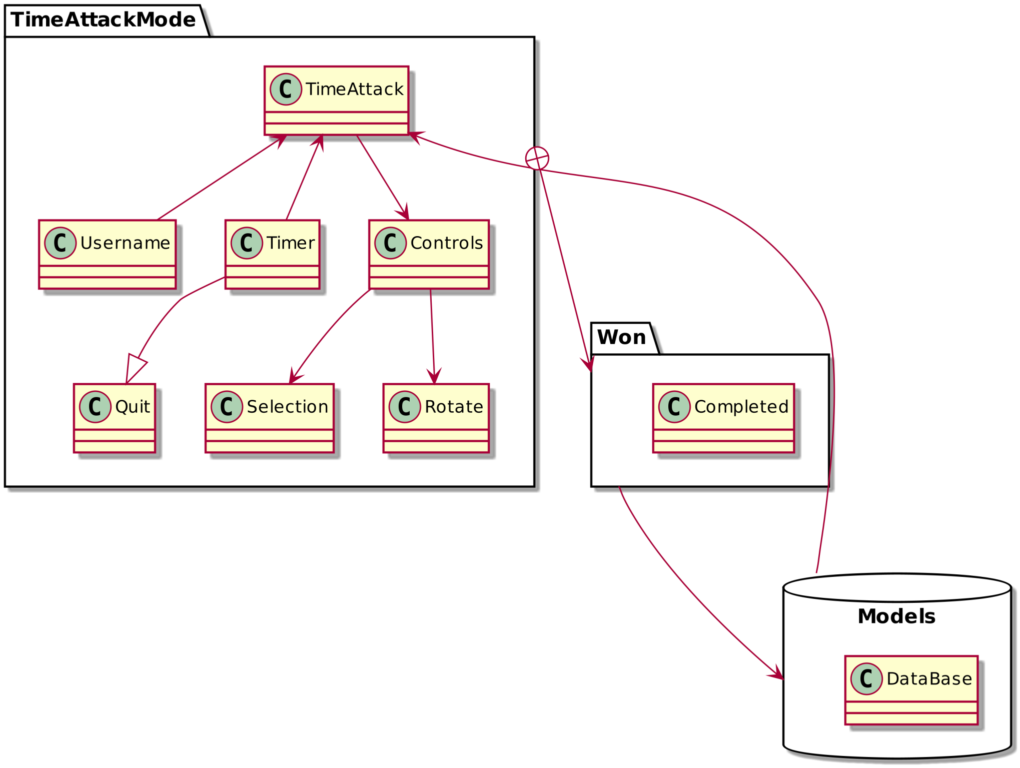
Standby State



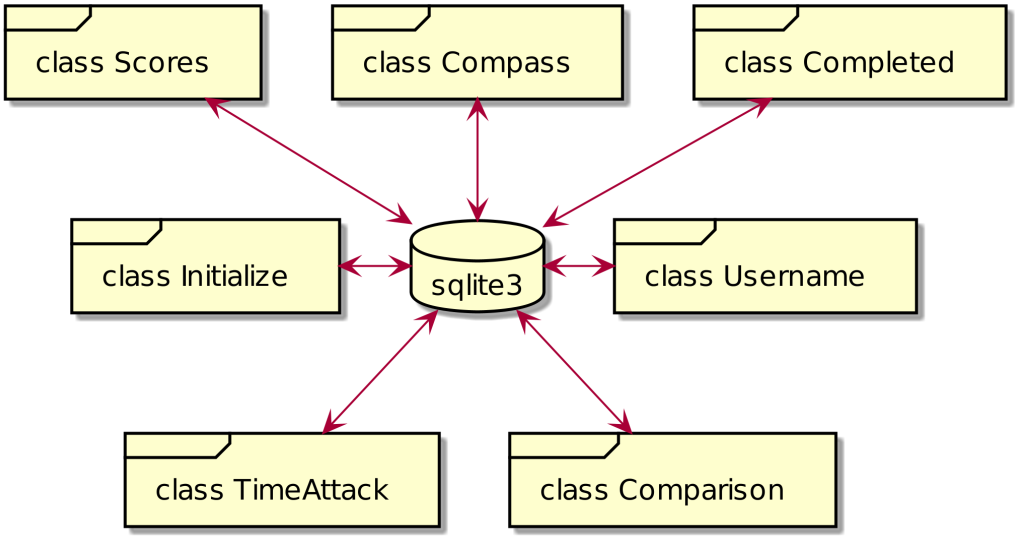
Active State



Time-Attack State



Data-centered Architecture


class Scores
class Compass
class Completed
class Initialize
sqlite3
class Username
class TimeAttack
class Comparison