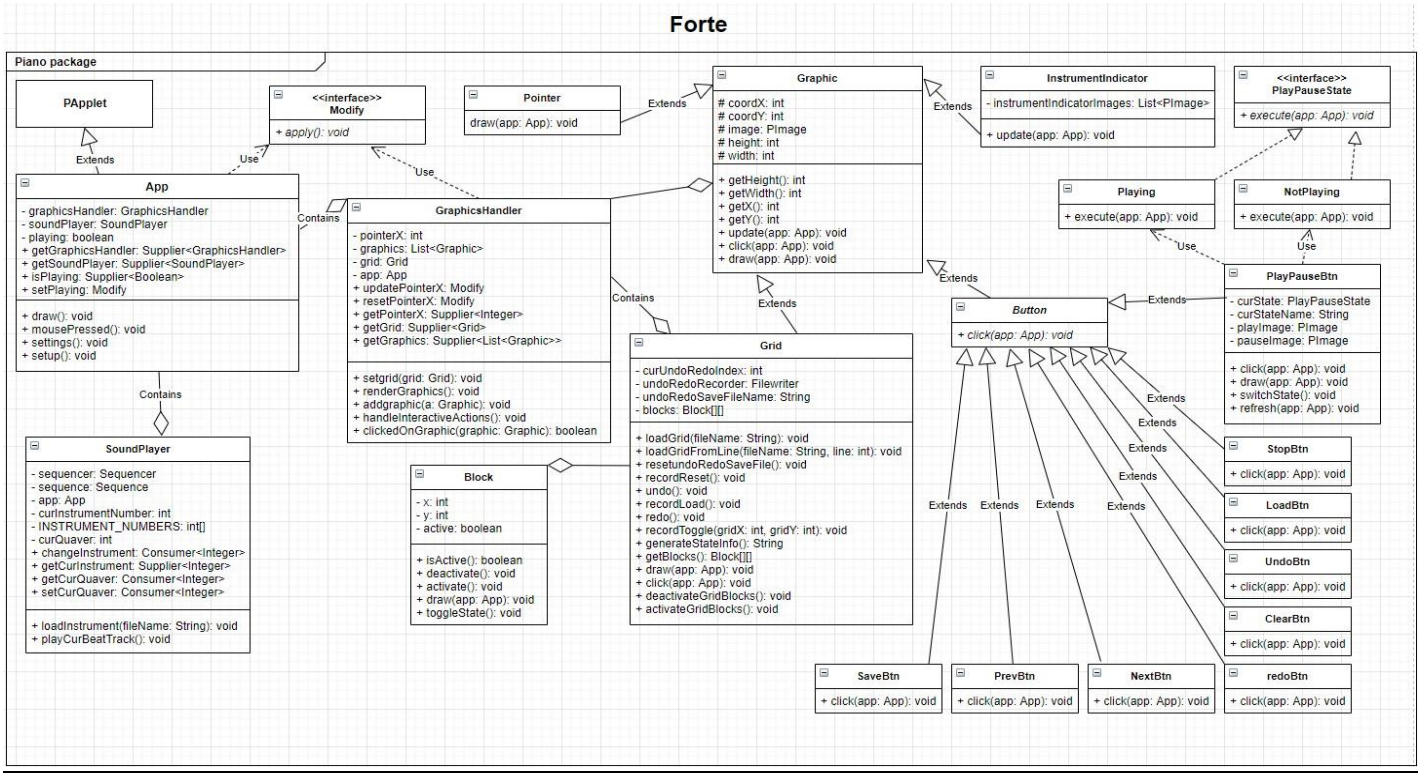
SID: 490432691

Assignment 3 report

UML diagram of application



Object oriented design (OOD) principles used

The three major design principles of OOD are encapsulation, polymorphism and inheritance. This Application utilizes classes, design patterns, Lambda expressions, functional and 'non-functional' interfaces, abstract classes and inheritance hierarchies in an aim to best pertain to these principles.

The following are the key features in relation to OOD in the application:

1. ### Minimal logic in App class

   Within the App class, essentially all the logic is handled by two objects; one of type GraphicsHandler, the other of type SoundPlayer. The GraphicsHandler object handles all graphics, while the SoundPlayer handles audio. Due to minimal logic housed, the App class contains under 70 lines of code.

2. ### Graphic hierarchy

   The graphic hierarchy used results in **minimal boiler-plate code** and **easier-to-understand code.**

   There are multiple graphics which must be handled in this application. The main features shared among these graphics such as coordinates are reflected in the fields and methods of the Graphic class. There are also varied features shared among the graphics including movement, interactivity (being able to be clicked and respond by executing some actions), varied response (i.e. PlayPauseBtn) and varied image (e.g. Instrument indicator).

   Static graphics such as the keyboard strip can be objects defined directly by the Graphic class as they have no non-common functionality.

   Non-static graphic classes override the update (app: App) method so they can access data to change their position.

   Interactive Graphic classes extend the Button abstract class and override the click (app: App) abstract method to enable response.

Graphics with varied images include the playPauseBtn and the Instrument indicator. The playPauseBtn changes its image based on its state while the instrument indicator contains images which it switches based on the current instrument number which it accesses from the soundPlayer.

3. State design pattern

   The state design pattern is used for the PlayPauseBtn, it results in simple and concise code.

4. Lambda expressions

   Lambda expressions are used primarily to make the code more concise. The functional interface Modify is defined so that Lambdas can be made to modify field values in a pre-defined way without any parameter input.

Undo and Redo (Extension feature) implementation

Undo redo functionality

The group of operations within the application which for which the undo and redo operations apply include CLEAR/RESET, LOAD and TOGGLE, where toggle refers to clicking on a block in the grid to deactivate or activate it.

Implementation details

Whenever one of the aforementioned operations occurs, some data is written to a save file. There is a cursor which keeps track of which entry in this file corresponds to the current state of the application. When undo is called the cursor goes back one entry and sets the application state based on the data in the entry. Similar to undo, redo makes the cursor go up one entry (if this is possible) and sets the state based on the entry the cursor is on.