Group Name(Maybe)

Demonstration

Code Walkthrough

GameCentre

Login/logout system

Game save/autosaves

Scoreboard save/load

Design Patterns (Singleton)

GameCentre

Keep game and user consistent between Activities

‘instance’ - keep track of the current instance

‘getInstance()’ - allows Activity to access current instance (or creates new instance)

Facilitates the MVC model. (Acts partly as a model, controller)

- Model - Tracks user and game
- Controller - Saving/loading of games/scoreboards.

Design Patterns (MVC)

Model:

- Board (Observable): Stores a list of Tile
- Tile: Contains the state and id of tiles

View:

- GestureDetectGridView (pull model): Accesses controller classes
- GameActivity (push model): Observer of Board

Controller:

- MovementController: Accessed by GestureDetectGridView
- BoardManager: Updates Board if an action is valid

Design Patterns (Iterator)

·TileIterator:

Defined in board class and used when filling up the attribute tiles containing tiles of the current board

·Allows classes to operate on Tiles without accessing the ‘tiles’ array

·Preserves the “immutability” of the instance outside of its class.

·Allows for iterating through the collections in different manners (could have made an iterator for going through it row wise and column wise)

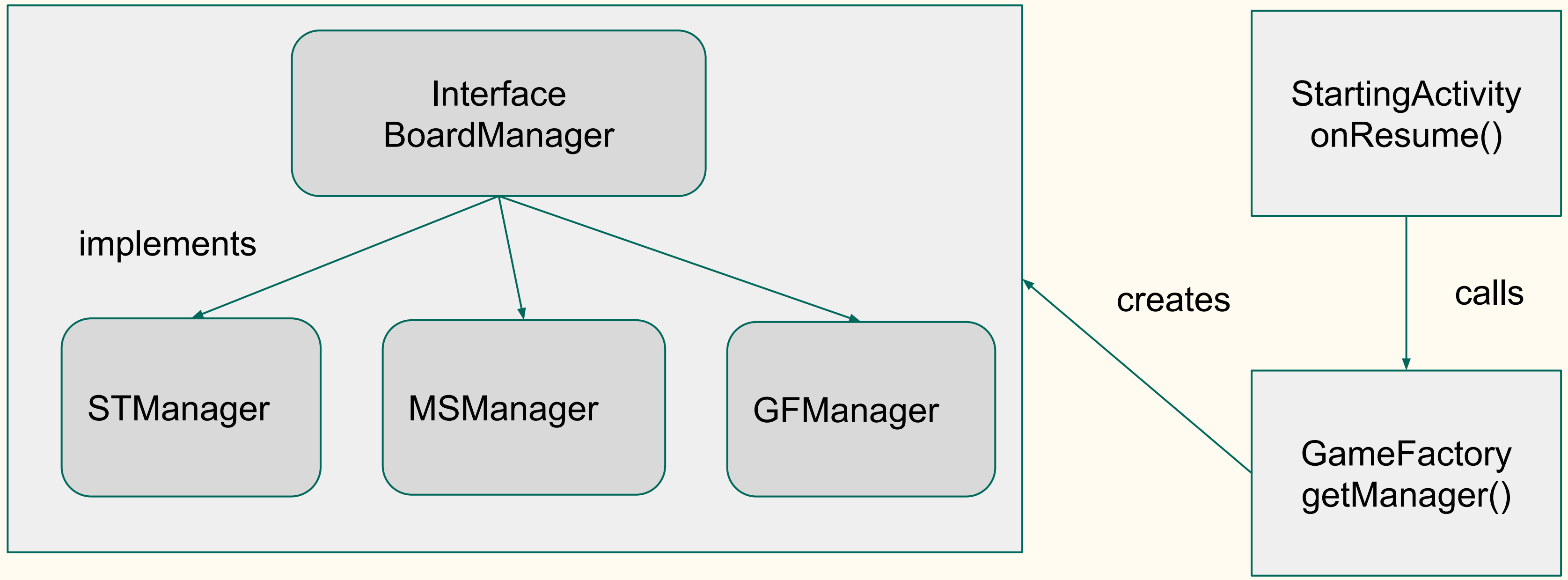
Design Patterns (Factory Design)

GameFactory

·Creates each individual BoardManager for the games

·Gets called in StartingActivity when initializing a new game.

- MS, ST, GF that all implement the BoardManager interface
- Allows for a different implementation of essential methods such as touchMove, isValidTap, isPuzzleSolved.



·Without the factory the method would need to call the getters for each manager of each game separately.

·The structure makes extension in the future more feasible.

Unit Test Walkthrough

(MSTest)