DataStorageIntf

+ saveFile(JSONObject _toBeStored): void

+ readFile(): JSONObject

MockDataStorage

- desiredLoadedData JSON(

APICallerInft

+ getGamesByGenre(String _genre, int _page): JSONObject
+ getGamesByTag(String _tag, int _page): JSONObject
+ getGenres(): JSONObject
+ getTags(): JSONObject

RAWGCaller

- static final baseURL String = "https://api.rawg.io/api"
- static final apiKey String = "6346c4bd4d004ac58323138cd49d65cb"

- call(String _callURL): JSONObject

GameFactory

- gameGenres String[]
- gameTags String[]
- gameTranslator GameTranslator
- static gameFactory GameFactory
- pageKeeperGenres HashMap<String, Integer>
- pageKeeperTags HashMap<String, Interger>

- isArryaRated(Game[] _toRate): boolean
+ getGameQueue(): GameQueue<Game>
+ getRecommendation(): Game
- compareGameBasedOnTag(Game _Game1, Game _game2, String _favoriteTag): Game
- getAnArrayThatIsNotRated(Game[] _toCheck, String _searchValue): Game[]

DataStorage

- fileOut FileWriter

- static fileIn BufferedReader

- static final fileName String = "C:\\TestingProjectSaveFiles\\gameGenieSaveData.txt"
- static final canNotFindFileJSONString String = "{\"viewedGames\":[],\"userTags\":[],\"userGenre\":[]}"

SaveDataTranslator

- static dataStorage DataStorageIntf

+ static saveUserData(UserHistoryIntf _dataToSave): void
+ static loadUserData(): UserHistoryInts
- static unwrap(JSONObject _saveData): UserHistoryIntf
- static jsonArrayToStrIntMap(JSONArray _inputArray): HashMap<String, Integer>
- static jsonArrayToStrStrMap(JSONArray _inputArray): HashMap<String, String>
- static toJSON(HashMap<String, Integer>, HashMap<String, Integer>, HashMap<String, String>): JSONObject
- static fromStringIntMap(HashMap<String, Integer>): JSONArray
- static fromStringStringMap(HashMap<String,String>): JSONArray

Test

+ static test()

- static test1()

- static test2()

GameTranslatorIntf

+ gerGamesByGenre(String _genre, int _page): Game[]
+ getGamesByTag(String _tag, int _page): Game[]
+ getGenres(): String[]
+ getTags(): String[]

GameTranslator

- myCaller APICallerInft

- parseGameJSON(JSONObject _toParse): Game[]

Main

+ static main(String[] args): void
+ start(Stage orimaryStage): void
+ stop(): void

«interface»
AbstractQueue<Game>

GameQueue

Game

- final genre String[]
- final releaseDate String
- final title String
- final tags String[]
- final platforms String[]
- final metacriticScore String
- gameCoverURL URL
- coverFilePath Path
- final gameID String

+ hasGenre(String _genre): boolean
+ hasTag(String _tag): boolean

GameGenieController

- primaryStage Stage
- static controller GameGenieController
- static gameQueue GameQueue<Game>
- static final startScreenFile String = "/startScreen.fxml"
- static final gamePickerFile String = "/gamePickerScreen.fxml"
- static final gameRecommendationFile String = "/gameRecommendationScreen.fxml"

\# setPrimaryStage(Stage _stage): void
- updateStage(String _fxmlFile): void
+ changeSceneIntoGamePicker(): void
+ changeSceneIntoGameRecommendation(): void
+ static getInstance(): GameGenieController
+ static handleStartCheckBoxes(CheckBox[] _startCheckBoxes): void
+ static getGamePickerGame(): Game
+ static userLikedGame(): void
+ static userDislikedGame(): void
+ static userDoesNotKnow(): void
+ static getRecommendation(): Game
+ static applicationClosing(): void

GameController

+ field: type

\# static getRecommendation(): Game
\# static getGameQueue(): GameQueue<Game>
+ static wasGameViewed(String _gameID): boolean
+ static mostLikedGenre(): String
+ static mostLikedTag(): String

StartScreenLogic

- actionBox CheckBox
- shooterBox CheckBox
- indieBox CheckBox
- casualBox CheckBox
- adventureBox CheckBox
- rpgBox CheckBox
- strategyBox CheckBox
- simulationBox CheckBox
- puzzleBox CheckBox
- arcadeBox CheckBox
- platformerBox CheckBox
- racingBox CheckBox
- sportsBox CheckBox
- massivelyMultiplayerBox CheckBox
- fightingBox CheckBox
- boardGamesBox CheckBox
- educationalBox CheckBox
- cardBox CheckBox
+ doneButton Button

+ initialize
- userClickedDoneButton()

GamePickerLogic

- currentGame Game
\# gameTitle Lable
\# getRecommendationButton Button
\# dislikeButton Button
\# likeButton Button
\# doNotKnowButton Button
\# gameCoverArtImageView ImageView

+ initialize(): void
+ DislikeClicked(ActionEvent actionEvent
+ LikeClicked(ActionEvent actionEvent)
+ DoNotKnowClicked(ActionEvent actionEvent)
+ setCurrentGame(Game _game)
+ getNextGame()
+ getRecommendationClicked()

GameRecommendationLogic

- currentGame Game
- gamePlatforms Label
- gameTitleLabel Label
- gameReleaseDate Label
- gameGenre Label
- gameMetacritic Lable
- recommendationGameTitle
- recommendationGameCoverArt ImageView
- doneWithRecommendationButton Button

+ initialize()
+ doneClicked()

UserHistoryInf

+ getTopGenre(): String
+ getTopTag(): String
+ addGenre(String _genre, int _preferenceMod): void
+ addTag(String _tag, int _preferenceMod): void
+ addGame(String _gameTitle, String _gameID): void
+ hasViewedGame(String _gameID): boolean
+ isEmpty(): bollean
+ isEqual(UserHistoryIntf _inputObj): boolean

UserController

- static final int defaultLikeValue = 1
- static final int defaultDislikeValue = -1

\# static handleCheckBoxes(CheckBox[] _checkBoxArray): void
\# static liked(Game _game): void
\# static disliked(Game _game): void
\# static userDataLoaded(): boolean
\# static programClose(): void

UserHistory

- userGenres DoubledLinkList
- userTags DoubledLinkList
- viewedGames HashMap<String, String>

- fromMapToLinkList(HashMap<String, Integer> _inputMap): DoubledLinkList
- fromLinkListToMap(DoubledLinkList _inputList): HashMap<String, Inreger>

User

- static user User
- userHistory UserHistoryIntf

+ static addStartScreenSelections(CheckBox[] _userSelection, int _incrementValue): void
+ static parseGame(Game _inputGame, int _incrementValue): void
- addViewedGame(String _gameID, String _gameTitle): void
+ wasViewed)String _gameID): boolean
- addGenre(String _genreName, int _inputValue): void
- addTag(String _tagName, int _inputValue): void

DoubledLinkList

\# head Node = null
\# tail Node = null

+ addElement(String _nodeTitle, int _preferenceValue): void
- searchAndPlace(Node _node): boolean
- orderAscending(): void
- orderDescending(): void
- swap(Node _lesserValue, Node _greaterValue): void
- printList(): void
- hasNode(Node _toCompare): boolean
+ isEqual(DoubledLinkList _toCompare): boolean
+ greatestValue(): String
+ lowestValue(): String
+ empty(): boolean