EECE435L_Game_Framework

Generated by Doxygen 1.8.13

Contents

Chapter 1

EECE 435L Game Framework

Author

Karim Zarzour
Maarouf Yassine

Date

20-11-2021

Chapter 2

README

This README would normally document whatever steps are necessary to get your application up and running.

What is this repository for?

- Quick summary
- Version
- `Learn Markdown`

How do I get set up?

- Summary of set up
- Configuration
- Dependencies
- Database configuration
- How to run tests
- Deployment instructions

Contribution guidelines

- Writing tests
- Code review
- Other guidelines

Who do I talk to?

- Repo owner or admin
- Other community or team contact

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

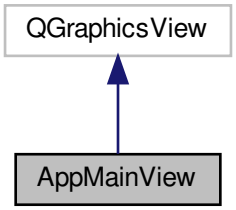
Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

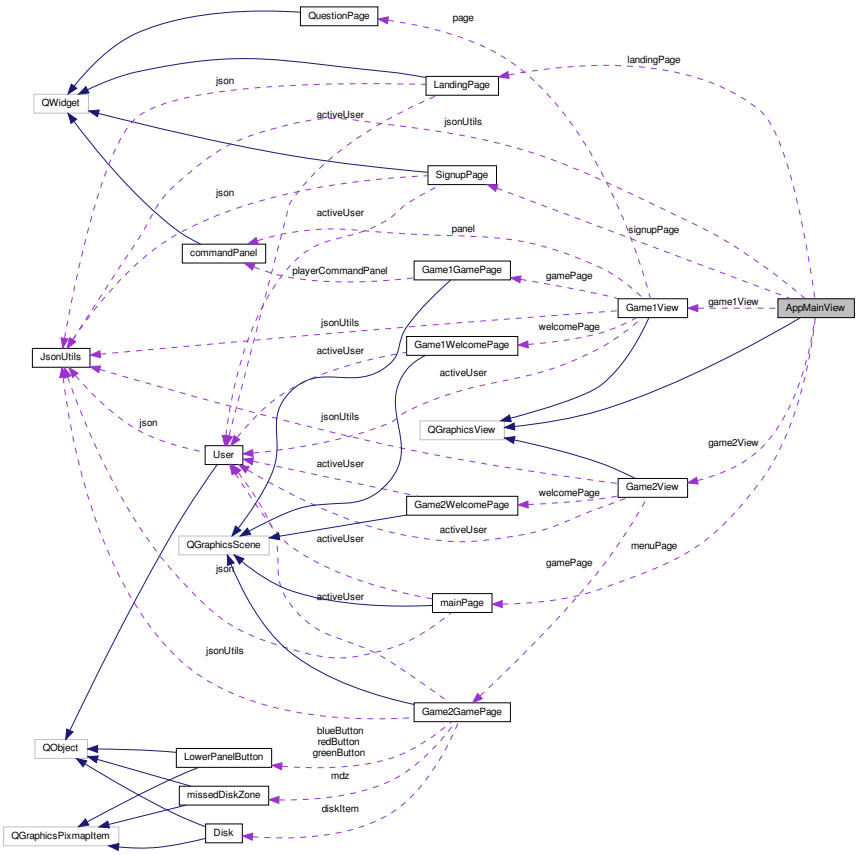
Class Documentation

5.1 AppMainView Class Reference

Inheritance diagram for AppMainView:



Collaboration diagram for AppMainView:



Public Slots

- void signup ()

  AppMainView::signup, changes the page to the signup page.
- void openMainPage ()

  AppMainView::openMainPage, changes the page to the main page.
- void playAsGuest ()

  AppMainView::playAsGuest, changes the page to the main page and allows player to play without an account.
- void login ()

  AppMainView::login, changes the page to the login page and clears widgets.
- void authenticateUser ()

  AppMainView::authenticateUser, on login checks if the username and password pair are correct.
- void logOut ()

  AppMainView::logOut, logs out the user from their account.
- void playGame1 ()

  AppMainView::playGame1, changes page to game 1 welcome page.
- void playGame2 ()

  AppMainView::playGame2, changes page to game 2 welcome page.

Public Member Functions

- void connectButtons ()

  AppMainView::connectButtons, connects the buttons to the functions that need to be called when they are clicked.

Public Attributes

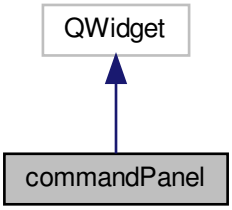
- JsonUtils * **jsonUtils**
- SignupPage * **signupPage**
- LandingPage * **landingPage**
- mainPage * **menuPage**
- Game1View * **game1View**
- Game2View * **game2View**

The documentation for this class was generated from the following files:

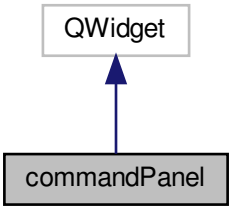
- appmainview.h
- appmainview.cpp

5.2 commandPanel Class Reference

Inheritance diagram for commandPanel:



Collaboration diagram for commandPanel:



Public Member Functions

- **commandPanel** (QWidget *parent=nullptr)

Public Attributes

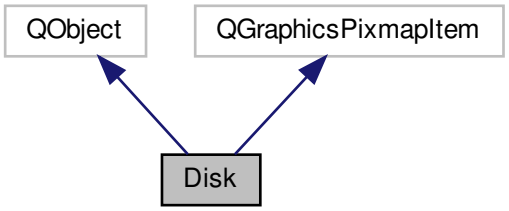
- QLabel * **mainLabel**
- QPushButton * **confirmPB**
- QLineEdit * **targetLineEdit**
- QVBoxLayout * **vlayout**

The documentation for this class was generated from the following files:

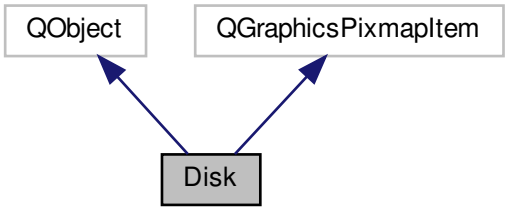
- Game1-BattleShip/commandpanel.h
- Game1-BattleShip/commandpanel.cpp

5.3 Disk Class Reference

Inheritance diagram for Disk:



Collaboration diagram for Disk:



Public Member Functions

- **Disk** (QObject *parent=nullptr, int gameSpeed=0)

Public Attributes

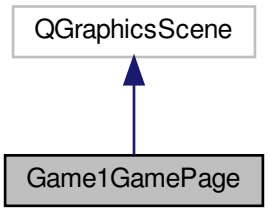
- int **type**
- int **gameSpeed** =0
- QTimer ∗ **timer**

The documentation for this class was generated from the following files:

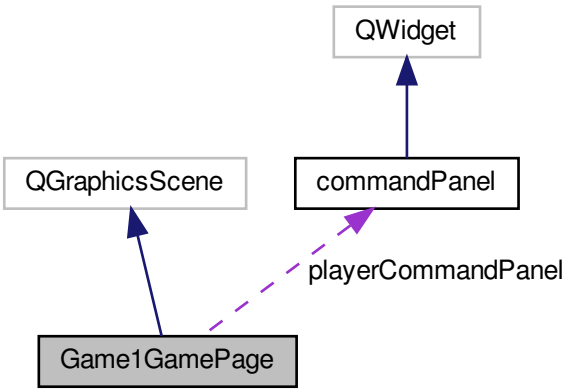
- Game2-ShootingDiscs/disk.h
- Game2-ShootingDiscs/disk.cpp

5.4 Game1GamePage Class Reference

Inheritance diagram for Game1GamePage:



Collaboration diagram for Game1GamePage:



Public Member Functions

- void setupScene ()

  Game1GamePage::setupScene, sets up the window of game 1 page and its background.
- void setupGrids ()

  Game1GamePage::setupGrids, sets up the grids of the user and the enemy.
- void fillScene ()

  Game1GamePage::fillScene, fills the game scene with all widgets and items.
- void setupBoats ()

  Game1GamePage::setupBoats, sets up boats on the grids of player and enemy.
- void setupWidgets ()

  Game1GamePage::setupWidgets, sets up widgets (Geometry and Appearance)
- void setupLabels ()

  Game1GamePage::setupLabels, sets up labels above the main players boats.
- void setupButtons ()

  Game1GamePage::setupButtons, set the geometry of buttons that cover the enemy's ships.
- QVector< int > getButtonPosition (QPushButton *button)

  Game1GamePage::getButtonPosition, gets position of the button that the player targets in the enemy's grid(ie. box position in the grid)
- bool discoverBlock (int x, int y)

  Game1GamePage::discoverBlock, discovers if under the button lies a part of an enemy ship.

Public Attributes

- QGraphicsPixmapItem * **player1Grid**
- QGraphicsPixmapItem * **player2Grid**
- QGraphicsPixmapItem * **player1Boat1**
- QGraphicsPixmapItem * **player1Boat2**
- QGraphicsPixmapItem * **player1Boat3**
- QGraphicsPixmapItem * **player2Boat1**
- QGraphicsPixmapItem * **player2Boat2**
- QGraphicsPixmapItem * **player2Boat3**
- QGraphicsPixmapItem * **player2Boat4**
- QGraphicsPixmapItem * **player2Boat5**
- QGraphicsPixmapItem * **player2Boat6**
- commandPanel * **playerCommandPanel**
- QLabel * **GCPLabel**
- QLabel * **BCPLabel**
- QLabel * **correctAnswerNo**
- QLabel * **incorrectAnswerNo**
- QLabel * **gameStatus**
- QPushButton * **button00**
- QPushButton * **button01**
- QPushButton * **button02**
- QPushButton * **button03**
- QPushButton * **button10**
- QPushButton * **button11**
- QPushButton * **button12**
- QPushButton * **button13**
- QPushButton * **button20**
- QPushButton * **button21**
- QPushButton * **button22**

- QPushButton * **button23**
- QPushButton * **button30**
- QPushButton * **button31**
- QPushButton * **button32**
- QPushButton * **button33**
- QLabel * **boat1Part1Label**
- QLabel * **boat1Part2Label**
- QLabel * **boat1Part3Label**
- QLabel * **boat2Label**
- QLabel * **boat3Label**
- int **badAnswers**
- QVector< QVector< QPushButton * > > **gridButtons**
- QVector< QVector< bool > > **userBoatPositions**
- QVector< QVector< bool > > **enemyBoatPositions**
- QString **lastBoxChosen**
- QPushButton * **home**
- bool **endGame** = false

5.4.1 Member Function Documentation

5.4.1.1 discoverBlock()

```
bool Game1GamePage::discoverBlock (
            int x,
            int y )
```

Game1GamePage::discoverBlock, discovers if under the button lies a part of an enemy ship.

Parameters

x	representing the x position.
y	representing the y position.

Returns

true if ship found, false otherwise.

5.4.1.2 getButtonPosition()

```
QVector< int > Game1GamePage::getButtonPosition (
            QPushButton * button )
```

Game1GamePage::getButtonPosition, gets position of the button that the player targets in the enemy's grid(ie. box position in the grid)

Parameters

button	

Returns

Qvector contianing the x,y coordinates of the button targeted. In case button not in grid return vector containing -1.

5.4.1.3 setupButtons()

```
void Game1GamePage::setupButtons ( )
```

Game1GamePage::setupButtons, set the geometry of buttons that cover the enemy's ships.

The buttons are not clickable, the are just present to cover the enemy ships.

5.4.1.4 setupLabels()

```
void Game1GamePage::setupLabels ( )
```

Game1GamePage::setupLabels, sets up labels above the main players boats.

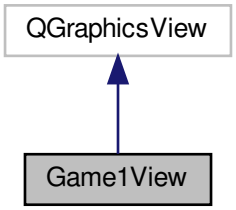
These labels get recoloured to red once the enemy hits one of the main player's boats.

The documentation for this class was generated from the following files:

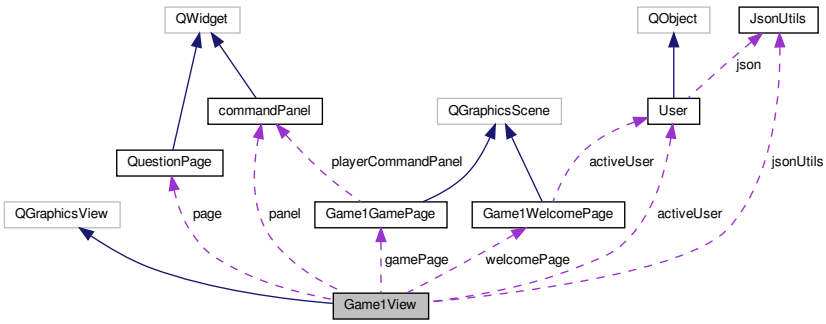
- Game1-BattleShip/game1gamepage.h
- Game1-BattleShip/game1gamepage.cpp

5.5 Game1View Class Reference

Inheritance diagram for Game1View:



Collaboration diagram for Game1View:



Public Slots

- void startGame ()

  Game1View::startGame, starts the game and changes scene to gamePage.
- void correctAnswer ()

  Game1View::correctAnswer, handles case of user entering a correct answer.
- void wrongAnswer ()

  Game1View::wrongAnswer, handles case of user entering a wrong answer.
- void hitHome ()

  Game1View::hitHome, handles case when user receives a hit from enemy.
- void attack ()

  Game1View::attack, takes input from command panel and attacks respective box in grid.
- void goToHome ()

  Game1View::goToHome, returns the user to home page to choose between the 2 games.

Public Member Functions

- void keyPressEvent (QKeyEvent ∗event)

  Game1View::keyPressEvent, starts the game when F1 Button is pressed.
- void connectButtons ()

  Game1View::connectButtons, connects multiple buttons to their corresponding slots.
- void clearQuestionPage ()

  Game1View::clearQuestionPage, clears question page when the game ends.
- void checkCurrGameScores ()

  Game1View::checkCurrGameScores, checks the in-game scores of the active game.
- void endCurrentGame (bool winOrLose)

  Game1View::endCurrentGame, ends game when user either wins or loses.
- void revealBox (int x, int y)

  Game1View::revealBox, reveals box if the user answered its corresponding question correctly, or if no ship is found under the box.
- void strikeBox (int x, int y)

  Game1View::strikeBox, marks box as red if user misses the correct answer to the question.

Public Attributes

- User * **activeUser** = NULL
- Game1WelcomePage * **welcomePage**
- Game1GamePage * **gamePage**
- QuestionPage * **page**
- QVector< bool > **currentGameScores**
- JsonUtils * **jsonUtils**
- commandPanel * **panel**
- QGraphicsView * **appMainView**

5.5.1 Member Function Documentation

5.5.1.1 endCurrentGame()

```
void Game1View::endCurrentGame (
            bool winOrLose )
```

Game1View::endCurrentGame, ends game when user either wins or loses.

Parameters

winOrLose,true	for winning and false for losing

5.5.1.2 keyPressEvent()

```
void Game1View::keyPressEvent (
            QKeyEvent * event )
```

Game1View::keyPressEvent, starts the game when F1 Button is pressed.

Parameters

event	

5.5.1.3 revealBox()

```
void Game1View::revealBox (
            int x,
            int y )
```

Game1View::revealBox, reveals box if the user answered its corresponding question correctly, or if no ship is found under the box.

Parameters

x	representing the x position of the box
y	representing the y position of the box

5.5.1.4 strikeBox()

```
void Game1View::strikeBox (
            int x,
            int y )
```

Game1View::strikeBox, marks box as red if user misses the correct answer to the question.

Parameters

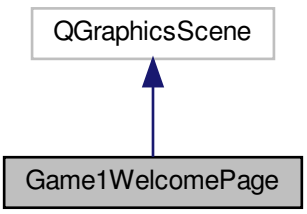
x	representing the x coordinate of the box
y	representing the y coordinate of the box

The documentation for this class was generated from the following files:

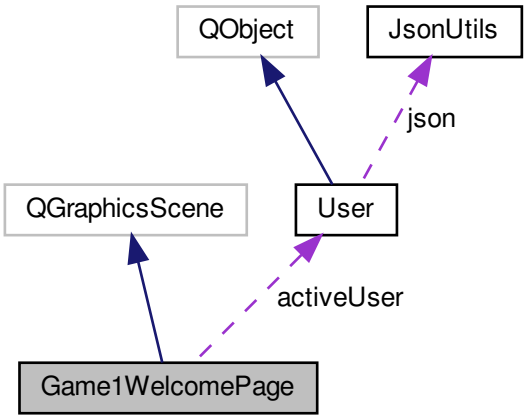
- Game1-BattleShip/game1view.h
- Game1-BattleShip/game1view.cpp

5.6 Game1WelcomePage Class Reference

Inheritance diagram for Game1WelcomePage:



Collaboration diagram for Game1WelcomePage:



Public Member Functions

- void setupScene ()

  Game1WelcomePage::setupScene, sets up the scene for the welcome page of game 1.

Public Attributes

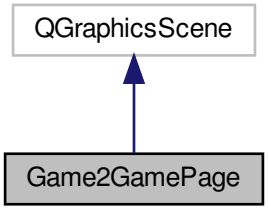
- User * **activeUser** = NULL
- QLabel * **gameName**
- QLabel * **gameInsructions**
- QPushButton * **playGame**

The documentation for this class was generated from the following files:

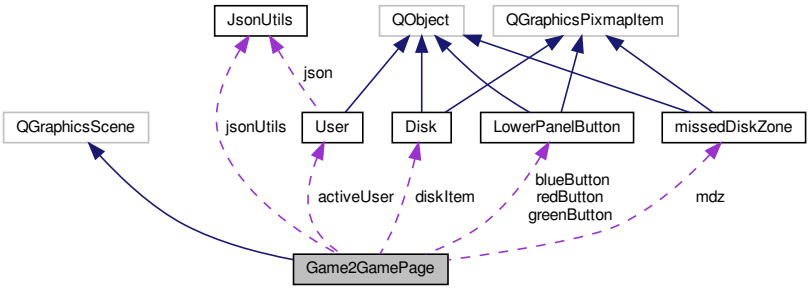
- Game1-BattleShip/game1welcomepage.h
- Game1-BattleShip/game1welcomepage.cpp

5.7 Game2GamePage Class Reference

Inheritance diagram for Game2GamePage:



Collaboration diagram for Game2GamePage:



Public Slots

- void addDisk ()

  Game2GamePage::addDisk, creates and adds a disk on the game boards.
- void checkMissedDisks ()

  Game2GamePage::checkMissedDisks, function that checks if a disk made it beyond the point of getting hit.

Public Member Functions

- void setupScene ()

  Game2GamePage::setupScene, sets up the background for game 2.
- void setupWidgets ()

  Game2GamePage::setupWidgets, sets up the widgets by setting the location and style of the widgets on the page.
- void setupGrid ()

  Game2GamePage::setupGrid, sets the game grid for game 2.
- void **setupButtons** ()

- void **setupLabels** ()
- void fillScene ()

  Game2GamePage::fillScene, add the widgets to the scene.
- void start ()

  Game2GamePage::start, starts the timers for game 2 every 2 seconds a new disk is aded and every 0.1 sexconds we check if a disk has been missed.
- void incrementScore (int n)

  Game2GamePage::incrementScore, increments the score depending on the color of the disk, if the score is greater or equal to 150 the game is ended.
- void incrementMisses ()

  Game2GamePage::incrementMisses, if a disk is missed increments the counter and if 3 disks are missed the game ends.
- void finishGame ()

  Game2GamePage::finishGame, funtion called when the game is supposed to end, if the score is greater or equal to 150, the player misses, if not then it means they missed 3 disks so they lose. Remaining disks are deleted. Score is added if player is logged in.
- void interuptGame ()

  Game2GamePage::interuptGame, stops the game when the player clicks the home button midgame.

Public Attributes

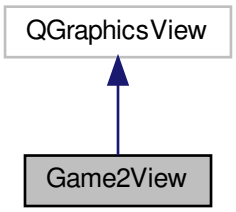
- JsonUtils * **jsonUtils**
- QGraphicsPixmapItem * **leftArrow**
- QGraphicsPixmapItem * **downArrow**
- QGraphicsPixmapItem * **rightArrow**
- QGraphicsPixmapItem * **gameGrid**
- QPushButton * **home**
- QLabel * **currentScore**
- QLabel * **highScore**
- QLabel * **currentScoreValue**
- QLabel * **highScoreValue**
- QLabel * **missedDisks**
- QLabel * **missedDisksValue**
- QLabel * **gameResult**
- LowerPanelButton * **redButton**
- LowerPanelButton * **greenButton**
- LowerPanelButton * **blueButton**
- missedDiskZone * **mdz**
- bool **endGame** = false
- Disk * **diskItem**
- User * **activeUser** = NULL
- QTimer * **timer**
- int **currentUserScore**
- int **highestScore**
- int **currentMissedDisks**
- int **gameSpeed**

The documentation for this class was generated from the following files:

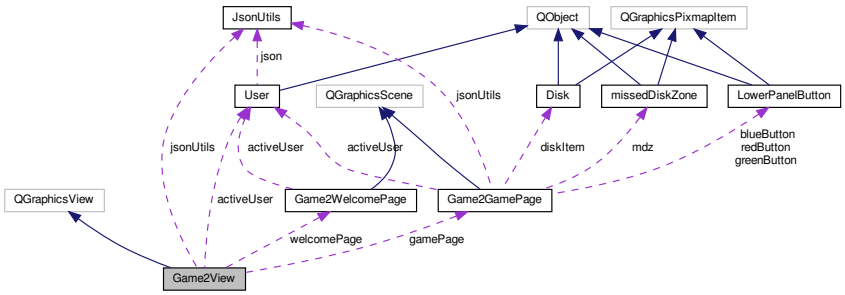
- Game2-ShootingDiscs/game2gamepage.h
- Game2-ShootingDiscs/game2gamepage.cpp

5.8 Game2View Class Reference

Inheritance diagram for Game2View:



Collaboration diagram for Game2View:



Public Slots

- void startGame ()

  Game2View::startGame, is called to start the game.
- void goToHome ()

  Game2View::goToHome, is called when home button is pressed, stops the game and takes the player back to the welcome page.

Public Member Functions

- void keyPressEvent (QKeyEvent ∗event)

  Game2View::keyPressEvent, is called when the arrow keys are pressed to delete the disks and add to the score.
- void connectButtons ()

  Game2View::connectButtons, connects the buttons to the functions that need to be called when they are clicked.

Public Attributes

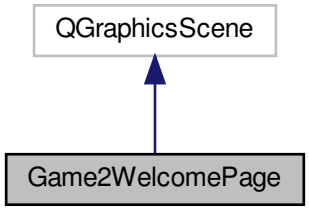
- User * **activeUser** = NULL
- QGraphicsView * **appMainView**
- JsonUtils * **jsonUtils**
- Game2WelcomePage * **welcomePage**
- Game2GamePage * **gamePage**

The documentation for this class was generated from the following files:

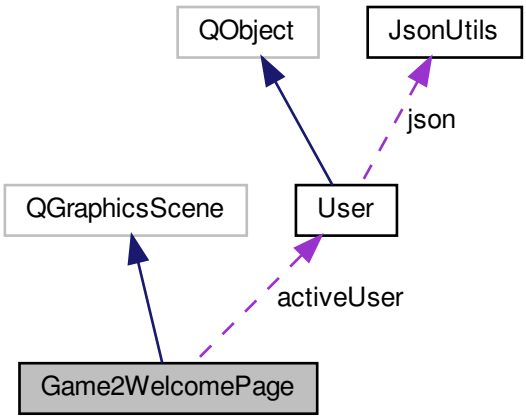
- Game2-ShootingDiscs/game2view.h
- Game2-ShootingDiscs/game2view.cpp

5.9 Game2WelcomePage Class Reference

Inheritance diagram for Game2WelcomePage:



Collaboration diagram for Game2WelcomePage:



Public Member Functions

- void setupScene ()

  Game2GamePage::setupScene, sets up the welcome page instructions and widgets.

Public Attributes

- User * **activeUser** = NULL
- QLabel * **gameName**
- QLabel * **gameInsructions**
- QPushButton * **playGame**

The documentation for this class was generated from the following files:

- Game2-ShootingDiscs/game2welcomepage.h
- Game2-ShootingDiscs/game2welcomepage.cpp

5.10 JsonUtils Class Reference

Public Member Functions

- void addUserToJson (QJsonObject user)

  Takes a newly created user and appends it to the users.json document.
- void updateScores (QString username, QVector< int > scores, int highscore, int gameIdentifier)

  JsonUtils::updateScores, Update the User Scores in the Json object of the User.
- QJsonDocument getJsonDocument ()

  Gets the JsonDocument of the file path.
- QJsonObject validateUser (QString &username, QString &password)

  Checks if a user who attempted to login has already signed up before.
- QJsonValue encodeProfilePicture (QPixmap &p)

  JsonUtils::encodeProfilePicture, Takes a picture, encodes it, and returns the corresponding hashed QJsonValue.
- QPixmap decodeProfilePicture (QJsonValue pic)

  JsonUtils::decodeProfilePicture, Takes a QJsonValue corresponding to a picture, decodes it, and returns the corresponding QPixmap.

Public Attributes

- QString **pathToJsonFile**

5.10.1 Member Function Documentation

5.10.1.1 addUserToJson()

```
void JsonUtils::addUserToJson (
            QJsonObject user )
```

Takes a newly created user and appends it to the users.json document.

Parameters

User	Object of type QJson Object

5.10.1.2 decodeProfilePicture()

```
QPixmap JsonUtils::decodeProfilePicture (
            QJsonValue pic )
```

JsonUtils::decodeProfilePicture, Takes a QJsonValue corresponding to a picture, decodes it, and returns the corresponding QPixmap.

Parameters

pic	

Returns

QJsonValue for the decoded image

5.10.1.3 encodeProfilePicture()

```
QJsonValue JsonUtils::encodeProfilePicture (
            QPixmap & p )
```

JsonUtils::encodeProfilePicture, Takes a picture, encodes it, and returns the corresponding hashed QJsonValue.

Parameters

p,picture	

Returns

QJsonValue for the encoded image

5.10.1.4 getJsonDocument()

```
QJsonDocument JsonUtils::getJsonDocument ( )
```

Gets the JsonDocument of the file path.

Returns

QJsonDocument of the file path

5.10.1.5 updateScores()

```
void JsonUtils::updateScores (
            QString username,
            QVector< int > scores,
            int highscore,
            int gameIdentifier )
```

JsonUtils::updateScores, Update the User Scores in the Json object of the User.

Parameters

username,username	of User
scores,scores	Arraylist of user
highscore,User's	Highscore
gameIdentifier,0	for game1 and 1 for game2

5.10.1.6 validateUser()

```
QJsonObject JsonUtils::validateUser (
            QString & username,
            QString & password )
```

Checks if a user who attempted to login has already signed up before.

Parameters

username,username	of user
password,password	of user

Returns

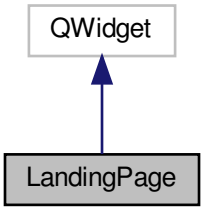
If the login was successful, returns the user as a QJsonObject.Else returns an empty QJSonObject

The documentation for this class was generated from the following files:

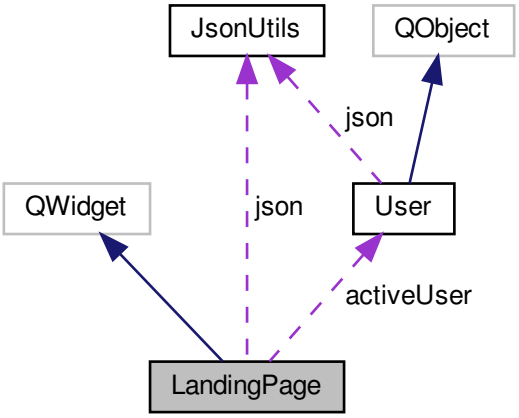
- Accounts_Framework/jsonutils.h
- Accounts_Framework/jsonutils.cpp

5.11 LandingPage Class Reference

Inheritance diagram for LandingPage:



Collaboration diagram for LandingPage:



Public Member Functions

- LandingPage (QWidget *parent=nullptr)

  LandingPage::LandingPage, sets the geometry of all widgets and labels. Adds them to the scene.

Public Attributes

- User * **activeUser** = NULL
- JsonUtils **json**
- QLabel * **welcomeLabel**
- QLabel * **warningLabel**

- QLabel * **userNameLabel**
- QLabel * **passwordLabel**
- QLineEdit * **userNameLineEdit**
- QLineEdit * **passwordLineEdit**
- QPushButton * **signInPushButton**
- QPushButton * **signUpPushButton**
- QPushButton * **guestPushButton**
- QGridLayout * **gridLayout**
- QVBoxLayout * **verticalLayout**

5.11.1 Constructor & Destructor Documentation

5.11.1.1 LandingPage()

```
LandingPage::LandingPage (
            QWidget * parent = nullptr )  [explicit]
```

LandingPage::LandingPage, sets the geometry of all widgets and labels. Adds them to the scene.

Parameters

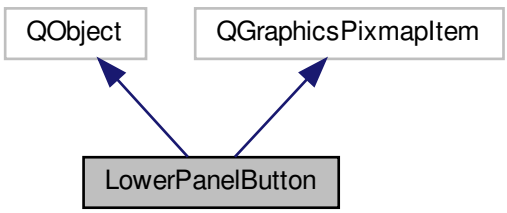
parent	

The documentation for this class was generated from the following files:

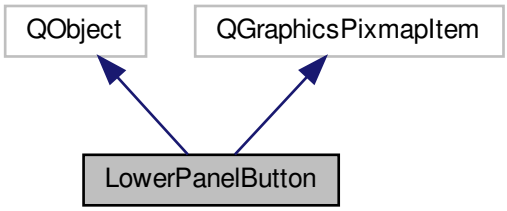
- Accounts_Framework/landingpage.h
- Accounts_Framework/landingpage.cpp

5.12 LowerPanelButton Class Reference

Inheritance diagram for LowerPanelButton:



Collaboration diagram for LowerPanelButton:



Public Member Functions

- **LowerPanelButton** (QObject *parent=nullptr, int type=-1)

Public Attributes

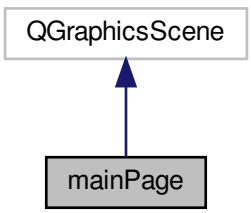
- int **type**

The documentation for this class was generated from the following files:

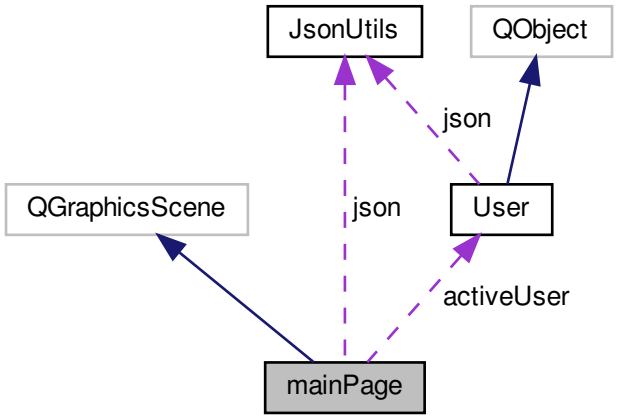
- Game2-ShootingDiscs/lowerpanelbutton.h
- Game2-ShootingDiscs/lowerpanelbutton.cpp

5.13 mainPage Class Reference

Inheritance diagram for mainPage:



Collaboration diagram for mainPage:



Public Member Functions

- **mainPage** (QObject *parent=nullptr)
- void setupGameLogos ()

  mainPage::setupGameLogos, Sets the icons of the games in their corresponding place on the scene
- void setupWidgetLocations ()

  mainPage::setupWidgetLocations, Sets the geometry of the widgets
- void addProfilePicture ()

  mainPage::addProfilePicture, Decodes a user's profile picture from a QJsonValue into a QPixmap. Sets the Pixmap p to the corresponding profile pic location on the scene
- void adjustLabelAppearance ()

  mainPage::adjustLabelAppearance, Function used to fix Labels.
- void fillScene ()

  mainPage::fillScene, Function Used to fill the Scene
- void updateScores ()

  mainPage::updateScores, Displays a user's scores to the scene for each corresponding game
- void clearPage ()

  mainPage::clearPage Called when we need to go to the maingview Cleans all widgets in order to prepare for another user to login/signup
- void setFlag ()

  mainPage::setFlag, sets the flag for corresponding active user

Public Attributes

- User * **activeUser** = NULL
- JsonUtils **json**
- QLabel * **welcomeL**
- QLabel * **dateL**
- QGraphicsPixmapItem * **game1Logo**

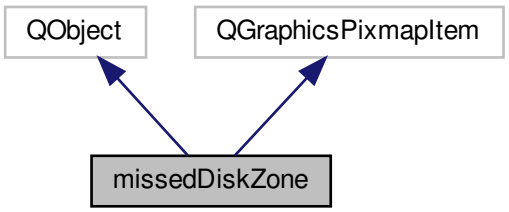
- QGraphicsPixmapItem * **game2Logo**
- QGraphicsPixmapItem * **userProfilePicture**
- QGraphicsPixmapItem * **flag**
- QPushButton * **game1B**
- QPushButton * **game2B**
- QPushButton * **homeB**
- QLabel * **game1Scores**
- QLabel * **game2Scores**

The documentation for this class was generated from the following files:

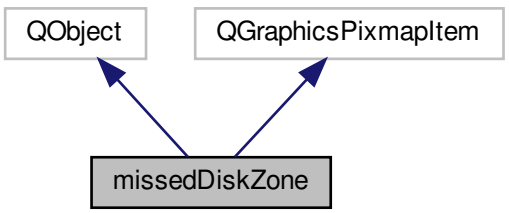
- Accounts_Framework/mainpage.h
- Accounts_Framework/mainpage.cpp

5.14 missedDiskZone Class Reference

Inheritance diagram for missedDiskZone:



Collaboration diagram for missedDiskZone:



Public Member Functions

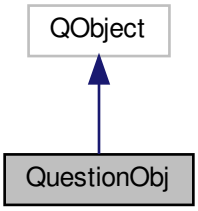
- **missedDiskZone** (QObject ∗parent=nullptr)

The documentation for this class was generated from the following files:

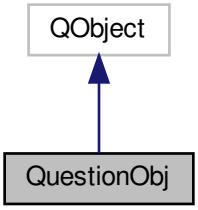
- Game2-ShootingDiscs/misseddiskzone.h
- Game2-ShootingDiscs/misseddiskzone.cpp

5.15 QuestionObj Class Reference

Inheritance diagram for QuestionObj:



Collaboration diagram for QuestionObj:



Public Member Functions

- QuestionObj (QObject ∗parent=nullptr)

  QuestionObj::QuestionObj, initializes a question object randomly.
- QuestionObj (QJsonObject jsonQuestion)

  QuestionObj::QuestionObj, returns a question object from a json input.
- QJsonObject getRandomQuestionFromJsonDocument ()

  QuestionObj::getRandomQuestionFromJsonDocument, chooses a question by random from the Json document containing questions.

Public Attributes

- QString **question**
- QString **trueAnswer**
- QString **falseAnswer**

5.15.1 Constructor & Destructor Documentation

5.15.1.1 QuestionObj() [1/2]

```
QuestionObj::QuestionObj (
            QObject * parent = nullptr )  [explicit]
```

QuestionObj::QuestionObj, initializes a question object randomly.

Parameters

parent	

5.15.1.2 QuestionObj() [2/2]

```
QuestionObj::QuestionObj (
            QJsonObject jsonQuestion )  [explicit]
```

QuestionObj::QuestionObj, returns a question object from a json input.

Parameters

jsonQuestion,question	read fron json file

5.15.2 Member Function Documentation

5.15.2.1 getRandomQuestionFromJsonDocument()

```
QJsonObject QuestionObj::getRandomQuestionFromJsonDocument ( )
```

QuestionObj::getRandomQuestionFromJsonDocument, chooses a question by random from the Json document containing questions.

Returns

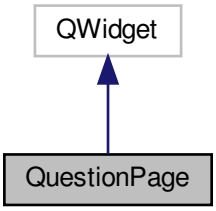
question of type questionObj

The documentation for this class was generated from the following files:

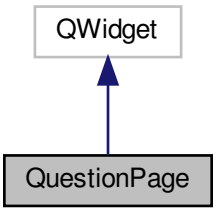
- Game1-BattleShip/questionobj.h
- Game1-BattleShip/questionobj.cpp

5.16 QuestionPage Class Reference

Inheritance diagram for QuestionPage:



Collaboration diagram for QuestionPage:



Public Member Functions

- **QuestionPage** (QWidget *parent=nullptr)
- void generateQuestion ()

  QuestionPage::generateQuestion, generates a new question object and updates the question page accordingly.

Public Attributes

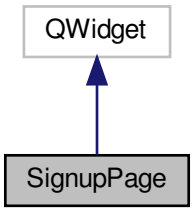
- QLabel * **questionL**
- QPushButton * **correctAnswerPB**
- QPushButton * **wrongAnswerPB**
- QVBoxLayout * **vlayout**

The documentation for this class was generated from the following files:

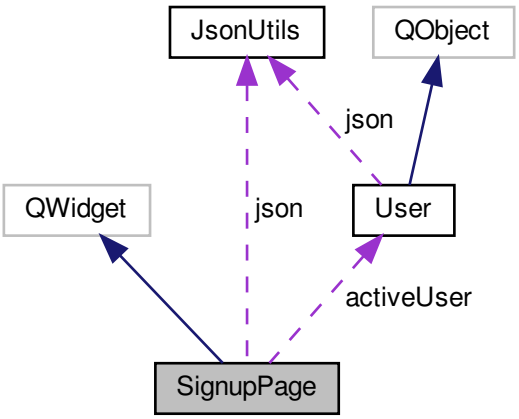
- Game1-BattleShip/questionpage.h
- Game1-BattleShip/questionpage.cpp

5.17 SignupPage Class Reference

Inheritance diagram for SignupPage:



Collaboration diagram for SignupPage:



Public Slots

- void addUser ()

  SignupPage::addUser Called whenever the signup button is pressed calls createUser() in order to check all necessary conditions before adding a new user to the users.json file if createUser() returned a user, setUser() appends it to users.json.
- void browseImage ()

  SignupPage::browseImage.

Public Member Functions

- **SignupPage** (QWidget *parent=nullptr)
- bool checkPassword (QString password)

  SignupPage::checkPassword, Checks if a password is valid, of size at least 4 and have special chars.
- User * createUser ()

  SignupPage::createUser, Reads the input from the widgets and attemps to create a new user.
- void clearPage ()

  SignupPage::clearPage, this methods resets all the widgets that took user input.
- void setupWidgets ()

  SignupPage::setupWidgets Sets the geometry of the widgets.

Public Attributes

- User * **activeUser** = NULL
- JsonUtils **json**
- QLabel * **headerL**
- QLabel * **warningL**
- QLabel * **SignInPromptL**
- QLineEdit * **firstNameLE**
- QLineEdit * **lastNameLE**
- QLineEdit * **usernameLE**
- QLineEdit * **passwordLE**
- QLineEdit * **confirmPasswordLE**
- QLineEdit * **phoneNumberLE**
- QSpinBox * **daySB**
- QSpinBox * **monthSB**
- QSpinBox * **yearSB**
- QRadioButton * **maleRB**
- QRadioButton * **femaleRB**
- QPushButton * **signUpB**
- QPushButton * **signInB**
- QPushButton * **uploadImageB**
- QPushButton * **playAsGuestB**
- QGroupBox * **groupBox**
- QFormLayout * **formL**
- QVBoxLayout * **GenderVerticalL**
- QHBoxLayout * **signInL**
- QHBoxLayout * **playAsGuestL**
- QHBoxLayout * **dateL**
- QVBoxLayout * **viewL**
- QHBoxLayout * **birthdayL**
- QString **fileName**

5.17.1 Member Function Documentation

5.17.1.1 browseImage

```
void SignupPage::browseImage ( )  [slot]
```

SignupPage::browseImage.

Takes profile picture file path from user updates the corresponding class member

5.17.1.2 checkPassword()

```
bool SignupPage::checkPassword (
            QString password )
```

SignupPage::checkPassword, Checks if a password is valid, of size at least 4 and have special chars.

Parameters

password	

Returns

True if valid, false otherwise.

5.17.1.3 createUser()

```
User * SignupPage::createUser ( )
```

SignupPage::createUser, Reads the input from the widgets and attemps to create a new user.

Returns

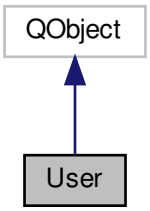
if successful, returns the new user (not yet added to users.json) else, returns NULL

The documentation for this class was generated from the following files:

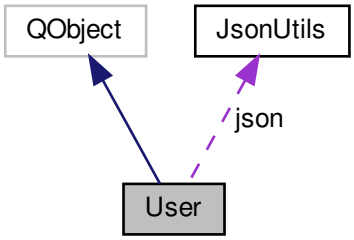
- Accounts_Framework/signuppage.h
- Accounts_Framework/signuppage.cpp

5.18 User Class Reference

Inheritance diagram for User:



Collaboration diagram for User:



Public Member Functions

- **User** (QObject *parent=nullptr)
- User (QJsonObject user)
- bool isUnique ()

  User::isUnique, Checks whether a User is unique or not.
- bool isValid ()

  User::isValid, Checks whether User's input is valid.
- QJsonObject userToJson ()

  User::userToJson, Transforms a User to a QJsonObject.
- QJsonArray scoresAsJsonArray (QVector< int > &scores)

  User::scoresAsJsonArray, Transforms a vector of scores to QJsonArray.
- QString findCorrespondingFlag ()

  User::findCorrespondingFlag, finds users corresponding flag from phone number.

Public Attributes

- JsonUtils **json**
- QString **username**
- QString **password**
- QString **firstName**
- QString **lastName**
- QString **dob**
- QString **gender**
- QString **phonenumber**
- QJsonValue **profilePicture**
- int **game1HighScore** =0
- int **game2HighScore** =0
- QVector< int > **game1Scores** ={}
- QVector< int > **game2Scores** ={}

5.18.1 Constructor & Destructor Documentation

5.18.1.1 User()

```
User::User (
            QJsonObject user )  [explicit]
```

Gets the User from a QJsonObject

Returns

a user from the users.json

5.18.2 Member Function Documentation

5.18.2.1 findCorrespondingFlag()

```
QString User::findCorrespondingFlag ( )
```

User::findCorrespondingFlag, finds users corresponding flag from phone number.

Returns

encoded flag image

5.18.2.2 isUnique()

```
bool User::isUnique ( )
```

User::isUnique, Checks whether a User is unique or not.

Returns

True if unique, False otherwise.

5.18.2.3 isValid()

```
bool User::isValid ( )
```

User::isValid, Checks whether User's input is valid.

Returns

true if valid, false otherwise.

5.18.2.4 scoresAsJsonArray()

```
QJsonArray User::scoresAsJsonArray (
            QVector< int > & scores )
```

User::scoresAsJsonArray, Transforms a vector of scores to QJsonArray.

Parameters

scores	

Returns

QJsonArray of scores

5.18.2.5 userToJson()

```
QJsonObject User::userToJson ( )
```

User::userToJson, Transforms a User to a QJsonObject.

Returns

Corresponding QJsonObject

The documentation for this class was generated from the following files:

- Accounts_Framework/user.h
- Accounts_Framework/user.cpp