turtle-1

Generated by Doxygen 1.9.1

1 Source content **1**

1.0.1 Contents: 1

2 Hierarchical Index **3**

2.1 Class Hierarchy 3

3 Class Index **5**

3.1 Class List 5

4 Class Documentation **7**

4.1 MainWindow Class Reference 7

4.2 Storage Class Reference 7

4.2.1 Member Function Documentation 7

4.2.1.1 addToHistory() 7

4.2.1.2 getHistory() 8

4.2.1.3 getModel() 8

4.3 Turtle Class Reference 8

4.3.1 Detailed Description 9

4.3.2 Constructor & Destructor Documentation 10

4.3.2.1 Turtle() 10

4.3.3 Member Function Documentation 10

4.3.3.1 enqueueCommand() 10

4.3.3.2 forward() 10

4.3.3.3 gameWon() 11

4.3.3.4 getDrawing() 11

4.3.3.5 getGamePos() 11

4.3.3.6 getPosition() 11

4.3.3.7 getRotation() 12

4.3.3.8 go() 12

4.3.3.9 setBrushSize() 12

4.3.3.10 setDrawing() 12

4.3.3.11 setHouse() 13

4.3.3.12 turn() 13

4.3.3.13 updateBrushColor() 13

Index **15**

Chapter 1

Source content

This folder should contain only hpp/cpp files of your implementation. You can also place hpp files in a separate directory `include`.

You can create a summary of files here. It might be useful to describe file relations, and brief summary of their content.

1.0.1 Contents:

main.cpp

mainwindow.cpp / .h, `includes "../ui/ui_mainwindow.h"`

storage.cpp / .h

turtle.cpp / .h, `includes "../ui/ui_mainwindow.h"`

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

- QGraphicsPixmapItem
  - Turtle 8
- QMainWindow
  - MainWindow 7
- QObject
  - Turtle 8
- Storage 7

Chapter 3

Class Index

3.1 Class List

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

MainWindow
Represents the main window of the application, managing the UI and interactions with the turtle 7
Storage 7
Turtle
Represents a turtle object that can move and draw paths in a graphics scene 8

Chapter 4

Class Documentation

4.1 MainWindow Class Reference

Represents the main window of the application, managing the UI and interactions with the turtle.

```
#include <mainwindow.h>
```

Inheritance diagram for MainWindow:

4.2 Storage Class Reference

Public Member Functions

- void addToHistory (const QString &line)
  addToHistory adds user typed commands to a list
- QStringList getHistory () const
  getHistory
- QStringListModel ∗ getModel () const
  getModel
- void helpDisplay ()
  helpDisplay shows all the commands that the user can type as a list in the historyView window
- void clearHistory ()
  clearHistory clears the user input list/history

4.2.1 Member Function Documentation

4.2.1.1 addToHistory()

```
void Storage::addToHistory (
             const QString & line )
```

addToHistory adds user typed commands to a list

Parameters

line	is the user given string that gets added to history

4.2.1.2 getHistory()

```
QStringList Storage::getHistory ( ) const
```

getHistory

Returns

returns a QStringList where all the commands are stored as a list

4.2.1.3 getModel()

```
QStringListModel * Storage::getModel ( ) const
```

getModel

Returns

returns the currently used model

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

- src/storage.h
- src/storage.cpp

4.3 Turtle Class Reference

Represents a turtle object that can move and draw paths in a graphics scene.

```
#include <turtle.h>
```

Inheritance diagram for Turtle:

Collaboration diagram for Turtle:

Public Member Functions

- Turtle (const QString &imagePath, QGraphicsScene ∗scene, Ui::MainWindow ∗ui)

  Constructs a Turtle object with the given image path and scene.
- void forward (int distance)

  Moves the turtle forward by the specified distance.
- void turn (int angle)

  Rotates the turtle counter-clockwise by the specified angle.
- void go (int x, int y)

  Moves the turtle to the given position.
- void setDrawing (bool drawing)

  Sets whether the turtle is drawing as it moves.
- bool getDrawing () const

  Checks whether the turtle is currently drawing.
- std::pair< int, int > getPosition () const

  Gets the current position of the turtle.
- int getRotation () const

  Gets the current rotation of the turtle.
- void setBrushSize (int value)

  Sets the brush size for drawing.
- void updateBrushColor (QColor color)

  Updates the brush color for drawing.
- void resetTurtle ()

  Resets the turtle and clears all drawn paths.
- void enqueueCommand (const std::function< void()> &command)

  Adds a command to the command queue for sequential execution.
- void processNextCommand ()

  Processes the next command in the queue, if available.
- void **star** ()
- void **triangle** ()
- void **square** ()
- void **rectangle** ()
- void **circle** ()
- void **cyclohexane** ()
- void **house** ()
- void **spinning** (int sides)
- void **random** ()
- std::pair< int, int > getGamePos ()

  Gets the game-specific random position.
- void updateUI ()

  Updates the user interface with the turtle's current position and rotation.
- void gameify ()

  Turns the turtle graphics into a game mode.
- void setHouse (QGraphicsPixmapItem ∗house)

  Sets the house object for the game mode.
- bool gameWon () const

  Checks whether the game has been won by reaching the target area.

4.3.1 Detailed Description

Represents a turtle object that can move and draw paths in a graphics scene.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Turtle()

```
Turtle::Turtle (
             const QString & imagePath,
             QGraphicsScene * scene,
             Ui::MainWindow * ui )
```

Constructs a Turtle object with the given image path and scene.

Parameters

imagePath	Path to the turtle image.
scene	Pointer to the QGraphicsScene where the turtle is displayed.
ui	Pointer to the user interface

4.3.3 Member Function Documentation

4.3.3.1 enqueueCommand()

```
void Turtle::enqueueCommand (
             const std::function< void()> & command )
```

Adds a command to the command queue for sequential execution.

Parameters

command	The command to enqueue as a std::function.

4.3.3.2 forward()

```
void Turtle::forward (
             int distance )
```

Moves the turtle forward by the specified distance.

Parameters

distance	Distance to move.

4.3.3.3 gameWon()

```
bool Turtle::gameWon ( ) const
```

Checks whether the game has been won by reaching the target area.

Returns

True if the game is won, false otherwise.

4.3.3.4 getDrawing()

```
bool Turtle::getDrawing ( ) const
```

Checks whether the turtle is currently drawing.

Returns

True if drawing is enabled, false otherwise.

4.3.3.5 getGamePos()

```
std::pair< int, int > Turtle::getGamePos ( )
```

Gets the game-specific random position.

Returns

The random position as a pair of (x, y) coordinates.

4.3.3.6 getPosition()

```
std::pair< int, int > Turtle::getPosition ( ) const
```

Gets the current position of the turtle.

Returns

The current position as a pair of (x, y) coordinates.

4.3.3.7 getRotation()

```
int Turtle::getRotation ( ) const
```

Gets the current rotation of the turtle.

Returns

The rotation in degrees.

4.3.3.8 go()

```
void Turtle::go (
             int x,
             int y )
```

Moves the turtle to the given position.

Parameters

x	coordinate given as int
y	coordinate given as int

4.3.3.9 setBrushSize()

```
void Turtle::setBrushSize (
             int value )
```

Sets the brush size for drawing.

Parameters

value	The size of the brush.

4.3.3.10 setDrawing()

```
void Turtle::setDrawing (
             bool drawing )
```

Sets whether the turtle is drawing as it moves.

Parameters

drawing	True to enable drawing, false to disable.

4.3.3.11 setHouse()

```
void Turtle::setHouse (
            QGraphicsPixmapItem * house )
```

Sets the house object for the game mode.

Parameters

house	Pointer to the QGraphicsPixmapItem representing the house.

4.3.3.12 turn()

```
void Turtle::turn (
            int angle )
```

Rotates the turtle counter-clockwise by the specified angle.

Parameters

angle	Angle to turn in degrees.

4.3.3.13 updateBrushColor()

```
void Turtle::updateBrushColor (
            QColor color )
```

Updates the brush color for drawing.

Parameters

color	The new brush color.

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

- src/turtle.h
- src/turtle.cpp

Index

addToHistory
    Storage, 7

enqueueCommand
    Turtle, 10

forward
    Turtle, 10

gameWon
    Turtle, 11
getDrawing
    Turtle, 11
getGamePos
    Turtle, 11
getHistory
    Storage, 8
getModel
    Storage, 8
getPosition
    Turtle, 11
getRotation
    Turtle, 11
go
    Turtle, 12

MainWindow, 7

setBrushSize
    Turtle, 12
setDrawing
    Turtle, 12
setHouse
    Turtle, 13
Storage, 7
    addToHistory, 7
    getHistory, 8
    getModel, 8

turn
    Turtle, 13
Turtle, 8
    enqueueCommand, 10
    forward, 10
    gameWon, 11
    getDrawing, 11
    getGamePos, 11
    getPosition, 11
    getRotation, 11
    go, 12
    setBrushSize, 12
    setDrawing, 12
    setHouse, 13
    turn, 13
    Turtle, 10
    updateBrushColor, 13

updateBrushColor
    Turtle, 13