# Pocket Calc

Generated by Doxygen 1.8.18

# Chapter 1

# Namespace Index

## 1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

# Chapter 2

# Hierarchical Index

## 2.1 Class Hierarchy

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

# Chapter 3

# Class Index

## 3.1 Class List

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

# Chapter 4

# File Index

## 4.1 File List

Here is a list of all documented files with brief descriptions:

# Chapter 5

# Namespace Documentation

## 5.1 CalculatorPack Namespace Reference

Package of files for calculator program.

### 5.1.1 Detailed Description

Package of files for calculator program.

**See also**

main.py
GUI.py
calculator.py
math_library.py

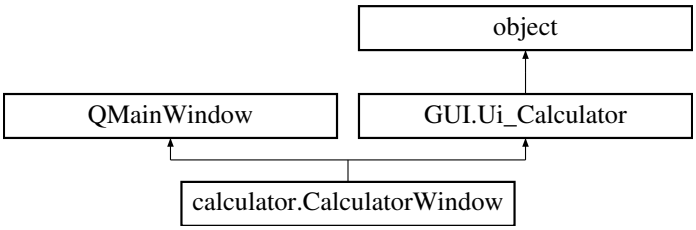
# Chapter 6

# Class Documentation

## 6.1 calculator.CalculatorWindow Class Reference

The controller of the calculator.

Inheritance diagram for calculator.CalculatorWindow:




### Public Member Functions

- def **__init__** (self)
- def digit_pressed (self)

  *Correctly transfers a digit to the main window.*
- def point_pressed (self)

  *Checks if a point can be entered.*
- def plus_minus_pressed (self)

  *Cases when the 'plus-minus' button is not appropriate.*
- def fact_pressed (self)

  *Consideration of departures when calculating factorial and calculating factorial.*
- def binary_operation_pressed (self)

  *The completion of one operation, the conclusion of the result and the preparation of indicators for another.*
- def equal_pressed (self)

  *Completion of the last operation and conclusion of the final result.*
- def clear_pressed (self)

  *Setting all indicators to False (and clearing a window)*
- def holding_button_clearing (self)

  *Setting all holding buttons indicators to False.*
- def holding_button_setting (self, button)

*Setting True to the button of the binary operation that was pressed.*

- def special_button_check (self, button)

  *Remembering the first operand and assigning status if a special operation is pressed.*
- def special_calculation (self, first, second, trying)

  *Exception Check and Calculation.*
- def new_window_jump (self)

  *Assigning status when moving to a new window.*
- def operands_connection (self)

  *Long operation input.*
- def overflow_check (self, overflow, answer)

  *Checking for a large number and possible error output.*
- def backspace_pressed (self)

  *Delete last digit.*
- def save_pressed (self)

  *Storing numbers in memory.*
- def log_insert (self, num, sign)

  *Check whether to change sign, and write the number and operation in the log.*
- def info_pressed (self)

  *Help output.*

## Public Attributes

- text

  *Window behavior indicator.*
- OverFlowError

  *Window behavior indicator.*
- TryBranch

  *Window behavior indicator.*
- NULLInsert

  *Window behavior indicator.*
- windowClearing

  *Window behavior indicator.*
- specialHolding

  *Holding special buttons indicator.*
- specialEntry

  *Holding special buttons indicator.*
- specialButtonClickedCounter

  *Holding special buttons indicator.*
- addHolding

  *Holding buttons indicator.*
- substractHolding

  *Holding buttons indicator.*
- multiplyHolding

  *Holding buttons indicator.*
- divideHolding

  *Holding buttons indicator.*
- powerHolding

  *Holding buttons indicator.*
- rootHolding

  *Holding buttons indicator.*

- operand
  *Variable for calculation.*
- first_special_operand
  *Variable for calculation.*
- second_special_operand
  *Variable for calculation.*
- answer
  *Variable for calculation.*
- num
  *Variable for calculation.*
- num_counter
  *Variable for calculation.*
- saving
  *Variable for saving.*
- max_length_input
  *Operand with >13 digits cant be.*
- pointStatus
  *Point indicator.*
- plus_minusStatus
  *Plus-minus indicator.*

### 6.1.1 Detailed Description

The controller of the calculator.

### 6.1.2 Member Function Documentation

#### 6.1.2.1 backspace_pressed()

```
def calculator.CalculatorWindow.backspace_pressed (
              self )
```

Delete last digit.

Backspace Click Processing

#### 6.1.2.2 binary_operation_pressed()

```
def calculator.CalculatorWindow.binary_operation_pressed (
              self )
```

The completion of one operation, the conclusion of the result and the preparation of indicators for another.

Binary Operation Click Processing

#### 6.1.2.3 clear_pressed()

```
def calculator.CalculatorWindow.clear_pressed (
              self )
```

Setting all indicators to False (and clearing a window)

Clear Click Processing

#### 6.1.2.4 digit_pressed()

```
def calculator.CalculatorWindow.digit_pressed (
              self )
```

Correctly transfers a digit to the main window.

**Parameters**

| | |
|---|---|
| *Digit* | Keystroke processing |

#### 6.1.2.5 equal_pressed()

```
def calculator.CalculatorWindow.equal_pressed (
              self )
```

Completion of the last operation and conclusion of the final result.

Equals Click Processing

#### 6.1.2.6 fact_pressed()

```
def calculator.CalculatorWindow.fact_pressed (
              self )
```

Consideration of departures when calculating factorial and calculating factorial.

Factorial Click Processing

#### 6.1.2.7 holding_button_clearing()

```
def calculator.CalculatorWindow.holding_button_clearing (
              self )
```

Setting all holding buttons indicators to False.

Holding Buttons clearing

#### 6.1.2.8 holding_button_setting()

```
def calculator.CalculatorWindow.holding_button_setting (
              self,
              button )
```

Setting True to the button of the binary operation that was pressed.

Holding pressed binary button in a memory

**Parameters**

| | |
|---|---|
| *button* | Code of pressed button |

#### 6.1.2.9 log_insert()

```
def calculator.CalculatorWindow.log_insert (
              self,
              num,
              sign )
```

Check whether to change sign, and write the number and operation in the log.

Log record Processing

**Parameters**

| | |
|---|---|
| *num* | Number to be logged |
| *sign* | Sign for verification |

#### 6.1.2.10 new_window_jump()

```
def calculator.CalculatorWindow.new_window_jump (
              self )
```

Assigning status when moving to a new window.

**Returns**

True

#### 6.1.2.11 operands_connection()

```
def calculator.CalculatorWindow.operands_connection (
              self )
```

Long operation input.

Connection of operands

#### 6.1.2.12 overflow_check()

```
def calculator.CalculatorWindow.overflow_check (
              self,
              overflow,
              answer )
```

Checking for a large number and possible error output.

**Returns**

'True' if overflow

**Parameters**

| | |
|---|---|
| *overflow* | Status of overflow |
| *answer* | Answer for checking |

#### 6.1.2.13 plus_minus_pressed()

```
def calculator.CalculatorWindow.plus_minus_pressed (
              self )
```

Cases when the 'plus-minus' button is not appropriate.

Plus-Minus Click Processing

#### 6.1.2.14 point_pressed()

```
def calculator.CalculatorWindow.point_pressed (
              self )
```

Checks if a point can be entered.

Point Click Processing

#### 6.1.2.15 save_pressed()

```
def calculator.CalculatorWindow.save_pressed (
              self )
```

Storing numbers in memory.

Save Click Processing

#### 6.1.2.16 special_button_check()

```
def calculator.CalculatorWindow.special_button_check (
              self,
              button )
```

Remembering the first operand and assigning status if a special operation is pressed.

Enabling calculation behavior in a special operation

**Returns**

'True' if special button pressed

**Parameters**

| | |
|---|---|
| *button* | Code of pressed button |

#### 6.1.2.17 special_calculation()

```
def calculator.CalculatorWindow.special_calculation (
              self,
              first,
              second,
              trying )
```

Exception Check and Calculation.

Special operations calculation

**Returns**

special operation value

**Parameters**

| | |
|---|---|
| *first* | First operand of special operations |
| *second* | Second operand of special operations |
| *trying* | Status of exception check |

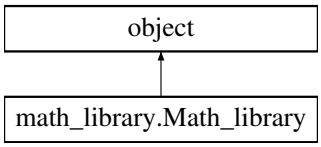
The documentation for this class was generated from the following file:

- calculator.py

## 6.2 math_library.Math_library Class Reference

Own math. operations.

Inheritance diagram for math_library.Math_library:




### Static Public Member Functions

- def add (x, y)
  *Adding.*
- def sub (x, y)
  *Subtraction.*
- def mul (x, y)
  *Multiplying.*
- def div (x, y)
  *Division.*
- def fact (x)
  *Factorial.*
- def n_root (x, y)
  *Common root.*
- def pow (x, y)
  *Power.*

### 6.2.1 Detailed Description

Own math. operations.

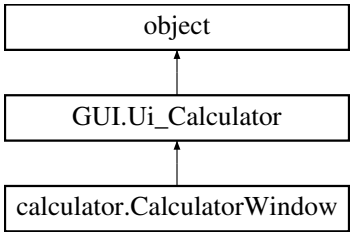
The documentation for this class was generated from the following file:

- math_library.py

## 6.3 GUI.Ui_Calculator Class Reference

Graphics settings generated in the designer.

Inheritance diagram for GUI.Ui_Calculator:



### Public Member Functions

- def **setupUi** (self, Calculator)
- def **retranslateUi** (self, Calculator)

### Public Attributes

- **label_main**
- **pushButton_7**
- **pushButton_8**
- **pushButton_9**
- **pushButton_multiply**
- **pushButton_4**
- **pushButton_5**
- **pushButton_6**
- **pushButton_substract**
- **pushButton_divide**
- **pushButton_add**
- **pushButton_equal**
- **pushButton_3**
- **pushButton_1**
- **pushButton_2**
- **pushButton_plus_minus**
- **pushButton_0**
- **pushButton_point**
- **pushButton_clear**
- **pushButton_save**
- **pushButton_factorial**
- **pushButton_power**
- **pushButton_root**
- **label_upper**
- **pushButton_backspace**
- **info**

#### 6.3.1 Detailed Description

Graphics settings generated in the designer.

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

- GUI.py

# Chapter 7

# File Documentation

## 7.1 calculator.py File Reference

Implementation of a controller.

### Classes

- class calculator.CalculatorWindow

  *The controller of the calculator.*

### Namespaces

- CalculatorPack

  *Package of files for calculator program.*

### 7.1.1 Detailed Description

Implementation of a controller.

**See also**

math_library.Math_library

GUI.Ui_Calculator

**Precondition**

Functional and tested math library, GUI

## 7.2 GUI.py File Reference

Graphical Interface.

### Classes

- class GUI.Ui_Calculator

  *Graphics settings generated in the designer.*

### Namespaces

- CalculatorPack

  *Package of files for calculator program.*

#### 7.2.1 Detailed Description

Graphical Interface.

## 7.3 main.py File Reference

Implementation of project main.

### Namespaces

- CalculatorPack

  *Package of files for calculator program.*

### Variables

- main.app = QApplication(sys.argv)

  *Preparation for GUI launch.*
- main.calculator = CalculatorWindow()

  *Controller connection.*

#### 7.3.1 Detailed Description

Implementation of project main.

**See also**

calculator.CalculatorWindow

GUI.Ui_Calculator

**Precondition**

Functional GUI and Controller

## 7.4 math_library.py File Reference

Implementation of own math. library.

### Classes

- class math_library.Math_library
  *Own math. operations.*

### Namespaces

- CalculatorPack
  *Package of files for calculator program.*

### 7.4.1 Detailed Description

Implementation of own math. library.

**Postcondition**

Library begins to be tested

## 7.5 standart_deviation.py File Reference

Calculation of Standart deviation.

### Variables

- int **standart_deviation.N** = 0
- int **standart_deviation.sum_sqrt_nums** = 0
- int **standart_deviation.sum_nums** = 0
- **standart_deviation.num** = float(num)
- **standart_deviation.mean** = math.div(sum_nums, N)
- **standart_deviation.result** = math.n_root(math.div(math.sub(sum_sqrt_nums, math.mul(N, math.↩ pow(mean, 2))), math.sub(N, 1)), 2)

### 7.5.1 Detailed Description

Calculation of Standart deviation.

**See also**

math_library.Math_library

**Precondition**

The data file consists of a number on each line and it is readable

**Postcondition**

Printing of standart deviation

## 7.6 test_basics.py File Reference

Tests of the basic math functions function from math_library.

### Functions

- def **test_basics.test_add** ()
- def **test_basics.test_add_float** ()
- def **test_basics.test_add_float_2** ()
- def **test_basics.test_sub** ()
- def **test_basics.test_sub_float** ()
- def **test_basics.test_sub_zero** ()
- def **test_basics.test_sub_minus** ()
- def **test_basics.test_mul** ()
- def **test_basics.test_mul_float** ()
- def **test_basics.test_mul_zero** ()
- def **test_basics.test_mul_minus** ()
- def **test_basics.test_div** ()
- def **test_basics.test_div_float** ()
- def **test_basics.test_div_float_2** ()
- def **test_basics.test_div_zero** ()

### 7.6.1 Detailed Description

Tests of the basic math functions function from math_library.

**See also**

math_library.Math_library

## 7.7 test_fact.py File Reference

Tests of the fact() function from math_library.

### Functions

- def **test_fact.test_fact** ()
- def **test_fact.test_fact_neg** ()
- def **test_fact.test_fact_zero** ()
- def **test_fact.test_fact_zero_float** ()
- def **test_fact.test_fact_float** ()

### 7.7.1 Detailed Description

Tests of the fact() function from math_library.

**See also**

math_library.Math_library

## 7.8 test_pow.py File Reference

Tests of the pow() function from math_library.

### Functions

- def **test_pow.test_pow** ()
- def **test_pow.test_pow_neg** ()
- def **test_pow.test_pow_zero** ()

### 7.8.1 Detailed Description

Tests of the pow() function from math_library.

**See also**

math_library.Math_library

## 7.9 test_root.py File Reference

Tests of the n_root() function from math_library.

### Functions

- def **test_root.test_root** ()
- def **test_root.test_root_2** ()
- def **test_root.test_root_neg** ()
- def **test_root.test_root_neg_2** ()
- def **test_root.test_root_zero** ()

### 7.9.1 Detailed Description

Tests of the n_root() function from math_library.

**See also**

math_library.Math_library

# Index