

SimpleCalc

Generated by Doxygen 1.8.12

Contents

1	SimpleCalc documentation	1
1.1	Introduction	1
1.2	Authors	1
2	Namespace Index	3
2.1	Packages	3
3	Hierarchical Index	5
3.1	Class Hierarchy	5
4	Class Index	7
4.1	Class List	7
5	File Index	9
5.1	File List	9
6	Namespace Documentation	11
6.1	main Namespace Reference	11
6.1.1	Detailed Description	12
6.1.2	Function Documentation	12
6.1.2.1	__init__()	12
6.1.2.2	ac()	12
6.1.2.3	build()	12
6.1.2.4	calculate()	13
6.1.2.5	delete()	13
6.1.2.6	dochelp()	13

6.1.2.7	dotpress()	14
6.1.2.8	numpress()	14
6.1.2.9	oppress()	14
6.1.2.10	signpress()	15
6.1.2.11	trigpress()	15
6.2	mat_module Namespace Reference	15
6.2.1	Detailed Description	16
6.2.2	Function Documentation	16
6.2.2.1	add()	16
6.2.2.2	calcBasicOperations()	17
6.2.2.3	calcFactorSqrt()	17
6.2.2.4	calcSum()	18
6.2.2.5	calculate()	18
6.2.2.6	cos()	19
6.2.2.7	determineSign()	19
6.2.2.8	div()	19
6.2.2.9	evaluate()	20
6.2.2.10	factorial()	20
6.2.2.11	findEndOfNum()	21
6.2.2.12	isnum()	21
6.2.2.13	lbidx()	22
6.2.2.14	log()	22
6.2.2.15	mul()	22
6.2.2.16	power()	23
6.2.2.17	rbidx()	23
6.2.2.18	root()	24
6.2.2.19	sin()	24
6.2.2.20	tan()	25
6.2.2.21	trigonFunc()	25
7	Class Documentation	27
7.1	main.CalcGridLayout Class Reference	27
7.1.1	Detailed Description	27
7.2	main.CalculatorApp Class Reference	27
7.2.1	Detailed Description	27
8	File Documentation	29
8.1	mat_module.py File Reference	29
8.1.1	Detailed Description	30
	Index	31

Chapter 1

SimpleCalc documentation

1.1 Introduction

This is the introduction about our project.

1.2 Authors

Andrej Nano Peter Marko Stanislav Mechl

Chapter 2

Namespace Index

2.1 Packages

Here are the packages with brief descriptions (if available):

main	Documentation for main	11
mat_module	Documentation of source code of mat_module which will be used in main.py	15

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

App	
main.CalculatorApp	27
GridLayout	
main.CalcGridLayout	27

Chapter 4

Class Index

4.1 Class List

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

main.CalcGridLayout	
Custom kivy GUI layout class	27
main.CalculatorApp	
Main application instance class	27

Chapter 5

File Index

5.1 File List

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

mat_module.py	29
-------------------------------	----

Chapter 6

Namespace Documentation

6.1 main Namespace Reference

Documentation for main.

Classes

- class [CalcGridLayout](#)
custom kivy GUI layout class
- class [CalculatorApp](#)
main application instance class

Functions

- def [__init__](#) (self, kwargs)
Creator method to setup Calculator instance overrides the class init method and sets input rules with boolean flags,.
- def [dotpress](#) (self)
Method to check if dot printing is allowed.
- def [numpress](#) (self, num)
Method to adjust flags when a number was pressed.
- def [signpress](#) (self, sign)
Method to adjust flags when a sign was pressed.
- def [oppress](#) (self, op)
Method to adjust flags when an operation was pressed.
- def [trigpress](#) (self, op)
Method to adjust flags when a trigonometric function was pressed.
- def [calculate](#) (self, calculation)
Result calculation method, uses '[mat_module](#)' module.
- def [delete](#) (self)
Character deleting method.
- def [ac](#) (self)
All clean method.
- def [build](#) (self)
Kivy method to build the GUI.
- def [dochelp](#) (self, args)
Method to generate help popup.

Variables

- `calc_inst = CalculatorApp()`
creates Calculator instance
- `op_allowed`
- `dot_allowed`
- `trig_allowed`
- `plus_allowed`
- `minus_allowed`
- `grid`
- `title`
- `icon`

6.1.1 Detailed Description

Documentation for main.

6.1.2 Function Documentation

6.1.2.1 `__init__()`

```
def main.__init__ (
    self,
    kwargs )
```

Creator method to setup Calculator instance overrides the class init method and sets input rules with boolean flags.

Parameters

<code>self,kwargs</code>	
--------------------------	--

6.1.2.2 `ac()`

```
def main.ac (
    self )
```

All clean method.

deletes the whole string in display.text and resets flags

Parameters

<code>self</code>	
-------------------	--

6.1.2.3 `build()`

```
def main.build (
```



```
self )
```

Kivy method to build the GUI.

Parameters

<i>self</i>	defines apps layout, title and icon
-------------	-------------------------------------

Returns

gui layout

6.1.2.4 calculate()

```
def main.calculate (
    self,
    calculation )
```

Result calculation method, uses '[mat_module](#)' module.

Parameters

<i>self, calculation</i>	passes string from display.text to mat_module to calculate the equation
--------------------------	---

Precondition

[mat_module](#) evaluation returns no exception

6.1.2.5 delete()

```
def main.delete (
    self )
```

Character deleting method.

Parameters

<i>self</i>	deletes the last character in display.text
-------------	--

Precondition

display.text is already empty -> resets flags
to be deleted character is an operation -> adjusts flags

6.1.2.6 dochelp()

```
def main.dochelp (
    self,
    args )
```

Method to generate help popup.

Parameters

<i>self, args</i>	(not used) adds a new widget, a help popup, to the kivy layout
-------------------	--

6.1.2.7 dotpress()

```
def main.dotpress (
    self )
```

Method to check if dot printing is allowed.

Parameters

<i>self</i>	
-------------	--

Precondition

input of dot is allowed (flag is true)

Returns

prints dot if the condition was true, otherwise nothing

6.1.2.8 numpress()

```
def main.numpress (
    self,
    num )
```

Method to adjust flags when a number was pressed.

Parameters

<i>self, num</i>	
------------------	--

Returns

updates the display.text with the new number

6.1.2.9 oppress()

```
def main.oppress (
    self,
    op )
```

Method to adjust flags when an operation was pressed.

Parameters

<i>self,op</i>	updates the display.text with the new operation symbol
----------------	--

6.1.2.10 signpress()

```
def main.signpress (
    self,
    sign )
```

Method to adjust flags when a sign was pressed.

Parameters

<i>self,sign</i>	updates the display.text with the new sign symbol
------------------	---

6.1.2.11 trigpress()

```
def main.trigpress (
    self,
    op )
```

Method to adjust flags when a trigonometric function was pressed.

Parameters

<i>self,op</i>	updates the display.text with the new trig. op symbol
----------------	---

6.2 mat_module Namespace Reference

documentation of source code of [mat_module](#) which will be used in main.py

Functions

- def [add](#) (A, B)
...
- def [mul](#) (A, B)
Function muls A and B.
- def [div](#) (A, B)
Function divides A by B.
- def [root](#) (A, B)
Function calculates A-th root of number B.
- def [factorial](#) (A)
Function calculates factorial of A.

- def [sin](#) (A)
Function finds sine of A.
- def [cos](#) (A)
Function finds cosine of A.
- def [tan](#) (A)
Function finds tangent of A.
- def [log](#) (A)
Function finds natural logarithm of A.
- def [power](#) (A, B)
Function calculates value of A to the power of B.
- def [isnum](#) (char)
Function determines whether character passed is digit.
- def [lidx](#) (string)
Function lidx = left bracket index determines last index at which is "(" before ")".
- def [ridx](#) (string)
Function ridx = right bracket index determines first index at which is ")" after "(".
- def [evaluate](#) (string)
Function evaluates string as it is collection of mathematical operations.
- def [findEndOfNum](#) (string)
Function finds index where number stored in string ends.
- def [trigonFunc](#) (string, sign, func)
Function calculates trigonometric functions as sin cos tan or log.
- def [calcFactorSqrt](#) (string)
Function calculates values of roots, factorials and pi and substitutes them into original string.
- def [determineSign](#) (string)
Function function determines sign of number from "before number" substring which starts with "-" or "+".
- def [calcSum](#) (string)
Function evaluates value of string containing only "-" or "+" operations.
- def [calcBasicOperations](#) (string, sign, operation)
- ...
- def [calculate](#) (string)
Function evaluates simple expression consisting only from operations $^$ $$ $/$ root !*

6.2.1 Detailed Description

documentation of source code of [mat_module](#) which will be used in main.py

Date

22 April 2017 this module defines functions for evaluation of string as mathematical operation

6.2.2 Function Documentation

6.2.2.1 add()

```
def mat_module.add (
    A,
    B )
```

...

Function adds A and B

Parameters

<i>A,B</i>	
------------	--

Precondition

A and B are float numbers

Returns

sum of A and B

6.2.2.2 calcBasicOperations()

```
def mat_module.calcBasicOperations (
    string,
    sign,
    operation )
```

...

Function substitutes value of "*" or "/" operations to original string

Parameters

<i>string</i>	sign is "*" or "/" operation is pointer to func
---------------	--

Precondition

string is instance of data type str

sign is instance of data type str and can contain just "*" or "/"

operation is pointer to function with two arguments which should be used to evaluate operation

Returns

calculated value converted to str substituted to original string

6.2.2.3 calcFactorSqrt()

```
def mat_module.calcFactorSqrt (
    string )
```

Function calculates values of roots, factorials and pi and substitutes them into original string.

Parameters

<i>string</i>	
---------------	--

Precondition

string is instance of data type str

Returns

string with substituted values of roots and factorials

6.2.2.4 calcSum()

```
def mat_module.calcSum (  
    string )
```

Function evaluates value of string containing only "-" or "+" operations.

Parameters

<i>string</i>	
---------------	--

Precondition

string is instance of data type str and contains no other operations than "+" or "-"

Returns

calculated value converted to str

6.2.2.5 calculate()

```
def mat_module.calculate (  
    string )
```

Function evaluates simple expression consisting only from operations ^ * / root !

Parameters

<i>string</i>	
---------------	--

Precondition

string is instance of data type str

Returns

calculated value converted to str substituted to original string

6.2.2.6 cos()

```
def mat_module.cos (
    A )
```

Function finds cosine of A.

Parameters

<i>A</i>	
----------	--

Precondition

A is float number

Returns

cosine of A

6.2.2.7 determineSign()

```
def mat_module.determineSign (
    string )
```

Function function determines sign of number from "before number" substring which starts with "-" or "+".

Parameters

<i>string</i>	
---------------	--

Precondition

string is instance of data type str and starts with "+" or "-"

Returns

original string with with calculated and substituted sign at the begining

6.2.2.8 div()

```
def mat_module.div (
    A,
    B )
```

Function divides A by B.

Parameters

<i>A,B</i>	
------------	--

Precondition

A and B are float numbers

Returns

A divided by B

6.2.2.9 evaluate()

```
def mat_module.evaluate (
    string )
```

Function evaluates string as it is collection of mathematical operations.

Parameters

<i>string</i>	
---------------	--

Precondition

string is instance of data type str

Returns

result of operations from input string transformed in string format

Postcondition

error raised if not correct mathematical operation passed

6.2.2.10 factorial()

```
def mat_module.factorial (
    A )
```

Function calculates factorial of A.

Parameters

<i>A</i>	
----------	--

Precondition

A is float but natural number

Returns

factorial of A

6.2.2.11 findEndOfNum()

```
def mat_module.findEndOfNum (
    string )
```

Function finds index wher number stored in string ends.

Parameters

<i>string</i>	
---------------	--

Precondition

string is instance of data type str

Returns

int end index of number in string

6.2.2.12 isnum()

```
def mat_module.isnum (
    char )
```

Function determines whether character passed is digit.

Parameters

<i>char</i>	
-------------	--

Precondition

char is a string of length 1

Returns

1. 1 if char is digit
2. if char is not digit 0

6.2.2.13 lbidx()

```
def mat_module.lbidx (  
    string )
```

Function lbidx = left bracket index determines last index at which is "(" before ")."

Parameters

<i>string</i>	
---------------	--

Precondition

string is instance of data type str

Returns

1. -1 if "(" or ")" not found
2. else last index at which is "(" before ")"

6.2.2.14 log()

```
def mat_module.log (  
    A )
```

Function finds natural logarithm of A.

Parameters

<i>A</i>	
----------	--

Precondition

A is float number

Returns

natural logarithm of A

6.2.2.15 mul()

```
def mat_module.mul (  
    A,  
    B )
```

Function muls A and B.

Parameters

<i>A,B</i>	
------------	--

Precondition

A and B are float numbers

Returns

multiplication of A and B

6.2.2.16 power()

```
def mat_module.power (
    A,
    B )
```

Function calculates value of A to the power of B.

Parameters

<i>A</i>	base
<i>B</i>	exponent

Precondition

A and B are float numbers
B is natural whole number

Returns

A to the power B

6.2.2.17 rbidx()

```
def mat_module.rbidx (
    string )
```

Function rbidx = right bracket index determines first index at which is ")" after "(".

Parameters

<i>string</i>	
---------------	--

Precondition

string is instance of data type str

Returns

1. -1 if "(" or ")" not found
2. else first index at which is ")" after "("

6.2.2.18 root()

```
def mat_module.root (
    A,
    B )
```

Function calculates A-th root of number B.

Parameters

<i>A,B</i>	
------------	--

Precondition

A and B are float numbers

Returns

A-th root of B

6.2.2.19 sin()

```
def mat_module.sin (
    A )
```

Function finds sine of A.

Parameters

<i>A</i>	
----------	--

Precondition

A is float number

Returns

sine of A

6.2.2.20 tan()

```
def mat_module.tan (  
    A )
```

Function finds tangent of A.

Parameters

<i>A</i>	
----------	--

Precondition

A is float number

Returns

tangent of A

6.2.2.21 trigonFunc()

```
def mat_module.trigonFunc (  
    string,  
    sign,  
    func )
```

Function calculates trigonometric functions as sin cos tan or log.

Parameters

<i>string,sign</i>	is string containing sin cos tan or log - depends on operation
--------------------	--

Precondition

string is instance of data type str
sign is instance of data type str
func is pointer to function sin cos tan or log

Returns

string with substituted values of function

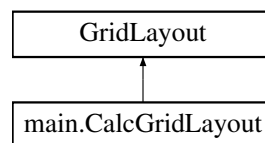
Chapter 7

Class Documentation

7.1 main.CalcGridLayout Class Reference

custom kivy GUI layout class

Inheritance diagram for main.CalcGridLayout:



7.1.1 Detailed Description

custom kivy GUI layout class

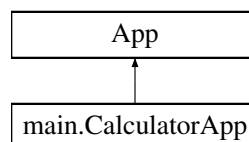
The documentation for this class was generated from the following file:

- main.py

7.2 main.CalculatorApp Class Reference

main application instance class

Inheritance diagram for main.CalculatorApp:



7.2.1 Detailed Description

main application instance class

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

- main.py

Chapter 8

File Documentation

8.1 mat_module.py File Reference

Namespaces

- [mat_module](#)

documentation of source code of [mat_module](#) which will be used in main.py

Functions

- def [mat_module.add](#) (A, B)
...
- def [mat_module.mul](#) (A, B)
Function muls A and B.
- def [mat_module.div](#) (A, B)
Function divides A by B.
- def [mat_module.root](#) (A, B)
Function calculates A-th root of number B.
- def [mat_module.factorial](#) (A)
Function calculates factorial of A.
- def [mat_module.sin](#) (A)
Function finds sine of A.
- def [mat_module.cos](#) (A)
Function finds cosine of A.
- def [mat_module.tan](#) (A)
Function finds tangent of A.
- def [mat_module.log](#) (A)
Function finds natural logarithm of A.
- def [mat_module.power](#) (A, B)
Function calculates value of A to the power of B.
- def [mat_module.isnum](#) (char)
Function determines whether character passed is digit.
- def [mat_module.lbidx](#) (string)
Function lbidx = left bracket index determines last index at which is "(" before ")".
- def [mat_module.rbidx](#) (string)

- Function rbdx = right bracket index determines first index at which is ")" after "(".*
- def [mat_module.evaluate](#) (string)

Function evaluates string as it is collection of mathematical operations.
- def [mat_module.findEndOfNum](#) (string)

Function finds index where number stored in string ends.
- def [mat_module.trigonFunc](#) (string, sign, func)

Function calculates trigonometric functions as sin cos tan or log.
- def [mat_module.calcFactorSqrt](#) (string)

Function calculates values of roots, factorials and pi and substitutes them into original string.
- def [mat_module.determineSign](#) (string)

Function function determines sign of number from "before number" substring which starts with "-" or "+".
- def [mat_module.calcSum](#) (string)

Function evaluates value of string containing only "-" or "+" operations.
- def [mat_module.calcBasicOperations](#) (string, sign, operation)

...
- def [mat_module.calculate](#) (string)

*Function evaluates simple expression consisting only from operations ^ * / root !*

8.1.1 Detailed Description

Author

Peter Marko

Index

- `__init__`
 - `main`, [12](#)
- `ac`
 - `main`, [12](#)
- `add`
 - `mat_module`, [16](#)
- `build`
 - `main`, [12](#)
- `calcBasicOperations`
 - `mat_module`, [17](#)
- `calcFactorSqrt`
 - `mat_module`, [17](#)
- `calcSum`
 - `mat_module`, [18](#)
- `calculate`
 - `main`, [13](#)
 - `mat_module`, [18](#)
- `cos`
 - `mat_module`, [18](#)
- `delete`
 - `main`, [13](#)
- `determineSign`
 - `mat_module`, [19](#)
- `div`
 - `mat_module`, [19](#)
- `dochelp`
 - `main`, [13](#)
- `dotpress`
 - `main`, [14](#)
- `evaluate`
 - `mat_module`, [20](#)
- `factorial`
 - `mat_module`, [20](#)
- `findEndOfNum`
 - `mat_module`, [21](#)
- `isnum`
 - `mat_module`, [21](#)
- `lbidx`
 - `mat_module`, [21](#)
- `log`
 - `mat_module`, [22](#)
- `main`, [11](#)

- `__init__`, [12](#)
- `ac`, [12](#)
- `build`, [12](#)
- `calculate`, [13](#)
- `delete`, [13](#)
- `dochelp`, [13](#)
- `dotpress`, [14](#)
- `numpress`, [14](#)
- `oppress`, [14](#)
- `signpress`, [15](#)
- `trigpress`, [15](#)
- `main.CalcGridLayout`, [27](#)
- `main.CalculatorApp`, [27](#)
- `mat_module`, [15](#)
 - `add`, [16](#)
 - `calcBasicOperations`, [17](#)
 - `calcFactorSqrt`, [17](#)
 - `calcSum`, [18](#)
 - `calculate`, [18](#)
 - `cos`, [18](#)
 - `determineSign`, [19](#)
 - `div`, [19](#)
 - `evaluate`, [20](#)
 - `factorial`, [20](#)
 - `findEndOfNum`, [21](#)
 - `isnum`, [21](#)
 - `lbidx`, [21](#)
 - `log`, [22](#)
 - `mul`, [22](#)
 - `power`, [23](#)
 - `rbidx`, [23](#)
 - `root`, [24](#)
 - `sin`, [24](#)
 - `tan`, [24](#)
 - `trigonFunc`, [25](#)
- `mat_module.py`, [29](#)
- `mul`
 - `mat_module`, [22](#)
- `numpress`
 - `main`, [14](#)
- `oppress`
 - `main`, [14](#)
- `power`
 - `mat_module`, [23](#)
- `rbidx`
 - `mat_module`, [23](#)

root
 mat_module, [24](#)

signpress
 main, [15](#)

sin
 mat_module, [24](#)

tan
 mat_module, [24](#)

trigonFunc
 mat_module, [25](#)

trigpress
 main, [15](#)