PackagingTest

Release 0.2.1

Philipp Schuette

Oct 04, 2020

TABLE OF CONTENTS:

CHAPTER

ONE

INTRODUCTION TO MY AWESOME DOCUMENTATION

This is a custom introduction for the documentation of my awesome PackagingTest! At this point, it is simply a placeholder for something meaningful. The only actual information you can find here, are the following references: [**?**].

1.1 Example Section for the Spinx Documentation

Here is a section with a very complicated formula:

$$1 + 1 = 2 \tag{1.1}$$

CHAPTER

TWO

MODULE 1 DOCUMENTATION

class `module1.`**`Employee`**(*id*, *first*, *last*)

Test class with some type checked attributes and some logging.

`__init__`(*id*, *first*, *last*)

Parameters

- **`id`** (*`int`*) – employee id
- **`first`** (*`str`*) – employee's first name
- **`last`** (*`str`*) – employee's last name

Return type instance of the *Employee* class

`module1.`**`add`**(*x*, *y*)

Adds two floats.

Parameters

- **`x`** (*`float`*) – first summand
- **`y`** (*`float`*) – second summand

Return type float

`module1.`**`divide`**(*x*, *y*)

Divides two floats where the second must be non-zero, otherwise a ZeroDivisionError is raise.

Parameters

- **`x`** (*`float`*) – numerator
- **`y`** (*`float != 0`*) – denominator

Return type float

`module1.`**`func1`**()

Type None

Return type str

`module1.`**`func2`**()

Type None

Return type List[float]

`module1.`**`func3`**()

Type None

Return type int

`module1.`**`get_parser`**`()`

Set logging level from command line.

`module1.`**`multiply`**`(`*x*, *y*`)`

Multiplies two floats.

Parameters

- **`x`** (*`float`*) – first factor
- **`y`** (*`float`*) – second factor

Return type float

`module1.`**`subtract`**`(`*x*, *y*`)`

Subtracts two floats.

Parameters

- **`x`** (*`float`*) – positive
- **`y`** (*`float`*) – negative

Return type float

CHAPTER

THREE

MODULE 2 DOCUMENTATION

`module2.`**`func1`**`()`

> **Type** None
>
> **Return type** None

`module2.`**`func2`**`()`

> **Type** None
>
> **Return type** str

`module2.`**`tail`**`(`*s*`)`

Takes an input string and returns its tail, i.e. everything except the first element.

> **Type** str
>
> **Return type** str

CHAPTER

FOUR

MODULE 3 DOCUMENTATION

`module3.`**`bar`**`()`

Also a discription, this time with some basic **rst** syntax.

`module3.`**`foo`**`()`

This is a long docstring that actually doesn't convey any useful information.

Type None

Return type None

`module3.`**`foo_bar`**`()`

Type None

Return type str

CHAPTER

FIVE

SUB_MODULE / MODULE 4 DOCUMENTATION

`sub_module.module4.`**`func1`**`()` → None
Function printing the module name *sub_module.module4*.

`sub_module.module4.`**`func2`**`()` → None
Function printing the complete path to file *module4*.

`sub_module.module4.`**`func3`**`()` → str
Function returning the submodule name *sub_module*.

`sub_module.module4.`**`go_fast`**`(`*a*`)`
Numba accelerated function doing computations on the main diagonal of an input NumPy array.

`sub_module.module4.`**`go_slow`**`(`**args*, ***kwargs*`)`

`sub_module.module4.`**`np_sum`**`(`*A*`)`
Sum of square roots of entries of a NumPy array using np.dot.

`sub_module.module4.`**`profiling`**`(`*param=False*`)`

`sub_module.module4.`**`sum_parallel`**`(`*A*`)`
The same as np_sum but Numba accelerated and in parallel.

`sub_module.module4.`**`sum_parallel_fast`**`(`*A*`)`
The same as sum_parallel but with *fastmath=True* enabled.

CHAPTER

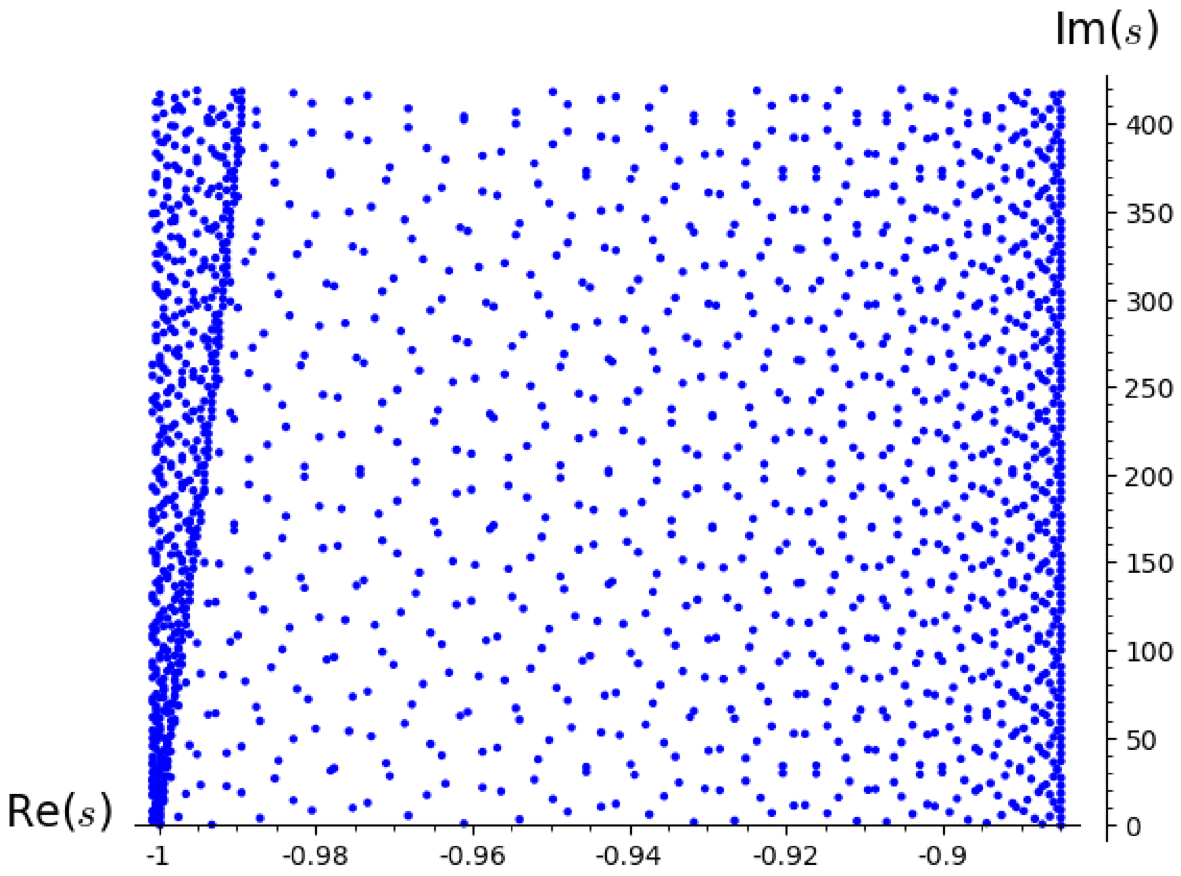
SIX

INDICES AND TABLES

CHAPTER

SEVEN

AN EXAMPLE GRAPHIC



BIBLIOGRAPHY

Otto Föllinger, *Regelungstechnik: Einführung in die Methoden und ihre Anwendungen*, 6. ed., Hüthig Buch Verlag GmbH, Heidelberg, 1990.

PYTHON MODULE INDEX

m

module1, **??**
module2, **??**
module3, **??**
MyIndexTest, **??**

s

sub_module.module4, **??**