rustpy

Azat Ibrakov

Mar 28, 2023

CONTENTS:

1 Submodules **1**

1.1 `option` module 1

1.2 `result` module 2

1.3 `primitive` module 3

2 Indices and tables **5**

Python Module Index **7**

Index **9**

CHAPTER

ONE

SUBMODULES

1.1 option module

class rustpy.option.**None_**

Represents absense of value. Implements *Option* protocol.

class rustpy.option.**Some**(*value: _T*)

Contains a value. Implements *Option* protocol.

class rustpy.option.**Option**(**args*, ***kwds*)

Protocol of an optional value.

and_(*_other:* Option*[_T]*) → *Option*[_T]

Returns self if it *is_none()*, otherwise returns other.

and_then(*_function: Callable[[_T],* Option*[_T2]]*) → *Option*[_T2]

Returns self if it *is_none()*, otherwise returns the result of given function on the contained value.

expect(*_message: str*) → _T

Returns the contained value or raises a `ValueError` with given message if none.

is_none() → _bool

Checks if the option does not contain a value.

is_some() → _bool

Checks if the option contains a value.

map(*_function: Callable[[_T], _T2]*) → *Option*[_T2]

Returns self if it *is_none()*, otherwise applies given function to a contained value.

map_or(*_default: _T2*, *_function: Callable[[_T], _T2]*) → _T2

Returns given default if self *is_none()*, otherwise returns the result of given function on the contained value.

map_or_else(*_default: Callable[[], _T2]*, *_function: Callable[[_T], _T2]*) → _T2

Returns the result of given default function if self *is_none()*, otherwise returns the result of given function on the contained value.

ok_or(*_err: _E*) → _Result[_T, _E]

Returns the contained value wrapped in *rustpy.result.Ok* if self *is_some()*, otherwise returns given value wrapped in *rustpy.result.Err*.

ok_or_else(*_function: _t.Callable[[], _E]*) → _Result[_T, _E]

Returns the contained value wrapped in `rustpy.result.Ok` if self `is_some()`, otherwise returns the result of given function wrapped in `rustpy.result.Err`.

or_(*_other:* Option*[_T]*) → *Option*[_T]

Returns self if it `is_some()`, otherwise returns other.

or_else(*_function: Callable[[],* Option*[_T]]*) → *Option*[_T]

Returns self if it `is_some()`, otherwise returns the result of the given function.

unwrap() → _T

Returns the contained value.

unwrap_or(*_default: _T*) → _T

Returns the default if self `is_none()`, otherwise returns the contained value.

unwrap_or_else(*_function: Callable[[], _T]*) → _T

Returns the result of default function if self `is_none()`, otherwise returns the contained value.

1.2 `result` module

class rustpy.result.**Err**(*value: _E*)

Represents the error value. Implements `Result` protocol.

class rustpy.result.**Ok**(*value: _T*)

Represents the success value. Implements `Result` protocol.

class rustpy.result.**Result**(**args*, ***kwds*)

and_(*_other:* Result*[_T2, _E]*) → *Result*[_T2, _E]

Returns self if it `is_err()`, otherwise returns other.

and_then(*_other: Callable[[_T],* Result*[_T2, _E]]*) → *Result*[_T2, _E]

Returns self if it `is_err()`, otherwise returns the result of given function on the success value.

err() → _Option[_E]

Returns error value wrapped in `rustpy.option.Some` if self `is_err()`, otherwise returns `rustpy.option.None_`.

expect(*_message: str*) → _T

Returns the success value or raises a `ValueError` with given message if error.

expect_err(*_message: str*) → _E

Returns the error value or raises a `ValueError` with given message if success.

is_err() → _bool

Checks if the result is an error.

is_ok() → _bool

Checks if the result is a success.

map(*_function: Callable[[_T], _T2]*) → *Result*[_T2, _E]

Returns self if it `is_err()`, otherwise applies given function to a success value.

map_err(*_function: Callable[[_E], _E2]*) → *Result*[_T, _E2]

Returns self if it `is_ok()`, otherwise applies given function to an error value.

map_or(*_default: _T2*, *_function: Callable[[_T], _T2]*) → _T2

Returns given default if self `is_err()`, otherwise returns the result of given function on the success value.

map_or_else(*_default: Callable[[_E], _T2]*, *_function: Callable[[_T], _T2]*) → _T2

Returns the result of given default function on the error value if self `is_err()`, otherwise returns the result of given function on the success value.

ok() → _Option[_T]

Returns success value wrapped in `rustpy.option.Some` if self `is_ok()`, otherwise returns `rustpy.option.None_`.

or_(*_other:* Result*[_T, _E2]*) → *Result*[_T, _E2]

Returns self if it `is_ok()`, otherwise returns other.

or_else(*_other: Callable[[_E],* Result*[_T, _E2]]*) → *Result*[_T, _E2]

Returns self if it `is_ok()`, otherwise returns the result of given function on the success value.

unwrap() → _T

Returns the success value.

unwrap_err() → _E

Returns the error value.

unwrap_or(*_default: _T*) → _T

Returns the default if self `is_err()`, otherwise returns the success value.

unwrap_or_else(*_function: Callable[[_E], _T]*) → _T

Returns the result of default function on error value if self `is_err()`, otherwise returns the success value.

1.3 primitive module

class rustpy.primitive.**bool_**(*_value: bool*)

Represents a value, which could only be either true or false. If you cast a bool into an integer, true will be 1 and false will be 0.

class rustpy.primitive.**f32**(*_value: float*)

A 32-bit floating point type (specifically, the "binary32" type defined in IEEE 754-2008).

class rustpy.primitive.**f64**(*_value: float*)

A 64-bit floating point type (specifically, the "binary64" type defined in IEEE 754-2008).

class rustpy.primitive.**i128**(*_value: int*)

The 128-bit signed integer type.

class rustpy.primitive.**i16**(*_value: int*)

The 16-bit signed integer type.

class rustpy.primitive.**i32**(*_value: int*)

The 32-bit signed integer type.

class rustpy.primitive.**i64**(*_value: int*)

The 64-bit signed integer type.

class rustpy.primitive.**i8**(*_value: int*)

The 8-bit signed integer type.

class `rustpy.primitive.`**`isize`**`(`*_value: int*`)`

The pointer-sized signed integer type.

class `rustpy.primitive.`**`u128`**`(`*_value: int*`)`

The 128-bit unsigned integer type.

class `rustpy.primitive.`**`u16`**`(`*_value: int*`)`

The 16-bit unsigned integer type.

class `rustpy.primitive.`**`u32`**`(`*_value: int*`)`

The 32-bit unsigned integer type.

class `rustpy.primitive.`**`u64`**`(`*_value: int*`)`

The 64-bit unsigned integer type.

class `rustpy.primitive.`**`u8`**`(`*_value: int*`)`

The 8-bit unsigned integer type.

class `rustpy.primitive.`**`usize`**`(`*_value: int*`)`

The pointer-sized unsigned integer type.

Note: If member is not listed in documentation it should be considered as implementation detail that can change and should not be relied upon.

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

r

rustpy.option, 1
rustpy.primitive, 3
rustpy.result, 2

INDEX

A

and_() (*rustpy.option.Option method*), 1
and_() (*rustpy.result.Result method*), 2
and_then() (*rustpy.option.Option method*), 1
and_then() (*rustpy.result.Result method*), 2

B

bool_ (*class in rustpy.primitive*), 3

E

Err (*class in rustpy.result*), 2
err() (*rustpy.result.Result method*), 2
expect() (*rustpy.option.Option method*), 1
expect() (*rustpy.result.Result method*), 2
expect_err() (*rustpy.result.Result method*), 2

F

f32 (*class in rustpy.primitive*), 3
f64 (*class in rustpy.primitive*), 3

I

i128 (*class in rustpy.primitive*), 3
i16 (*class in rustpy.primitive*), 3
i32 (*class in rustpy.primitive*), 3
i64 (*class in rustpy.primitive*), 3
i8 (*class in rustpy.primitive*), 3
is_err() (*rustpy.result.Result method*), 2
is_none() (*rustpy.option.Option method*), 1
is_ok() (*rustpy.result.Result method*), 2
is_some() (*rustpy.option.Option method*), 1
isize (*class in rustpy.primitive*), 3

M

map() (*rustpy.option.Option method*), 1
map() (*rustpy.result.Result method*), 2
map_err() (*rustpy.result.Result method*), 2
map_or() (*rustpy.option.Option method*), 1
map_or() (*rustpy.result.Result method*), 2
map_or_else() (*rustpy.option.Option method*), 1
map_or_else() (*rustpy.result.Result method*), 3
module
    rustpy.option, 1
    rustpy.primitive, 3
    rustpy.result, 2

N

None_ (*class in rustpy.option*), 1

O

Ok (*class in rustpy.result*), 2
ok() (*rustpy.result.Result method*), 3
ok_or() (*rustpy.option.Option method*), 1
ok_or_else() (*rustpy.option.Option method*), 1
Option (*class in rustpy.option*), 1
or_() (*rustpy.option.Option method*), 2
or_() (*rustpy.result.Result method*), 3
or_else() (*rustpy.option.Option method*), 2
or_else() (*rustpy.result.Result method*), 3

R

Result (*class in rustpy.result*), 2
rustpy.option
    module, 1
rustpy.primitive
    module, 3
rustpy.result
    module, 2

S

Some (*class in rustpy.option*), 1

U

u128 (*class in rustpy.primitive*), 4
u16 (*class in rustpy.primitive*), 4
u32 (*class in rustpy.primitive*), 4
u64 (*class in rustpy.primitive*), 4
u8 (*class in rustpy.primitive*), 4
unwrap() (*rustpy.option.Option method*), 2
unwrap() (*rustpy.result.Result method*), 3
unwrap_err() (*rustpy.result.Result method*), 3
unwrap_or() (*rustpy.option.Option method*), 2
unwrap_or() (*rustpy.result.Result method*), 3

`unwrap_or_else()` (*rustpy.option.Option method*), 2
`unwrap_or_else()` (*rustpy.result.Result method*), 3
`usize` (*class in rustpy.primitive*), 4