Zonis

Skelmis

Feb 26, 2024

CONTENTS:

1 Client 3

2 Server 5

3 Errors 9

4 Other References 11

5 Packet data structures 13

6 Router 15

7 Indices and tables 17

Index 19

Agnostic IPC for your Python programs.

CHAPTER

ONE

CLIENT

class zonis.**Client**

Bases: *RouteHandler*

Parameters

- **reconnect_attempt_count** (int) – The number of times that the *Client* should attempt to reconnect.
- **url** (str) – Defaults to `ws://localhost`.
- **port** (Optional[int]) – The port that the *Client* should use.

async block_until_closed()

A blocking call which releases when the WS closes.

async start() → None

Start the IPC client.

async close() → None

Stop the IPC client.

async request(*route: str*, ***kwargs*)

Make a request to the server

load_routes() → None

Loads all decorated routes.

register_class_instance_for_routes(*instance*, **routes*) → None

Register a class instance for the given route.

When you turn a method on a class into an IPC route, you need to call this method with the instance of the class to use as well as the names of the routes for registration to work correctly.

Parameters

- **instance** – The class instance the methods live on
- **routes** – A list of strings representing the names of the IPC routes for this class.

route(*route_name: Optional[str] = None*)

Turn an async function into a valid IPC route.

Parameters

route_name (*Optional[str]*) – An optional name for this IPC route, defaults to the name of the function.

Raises

DuplicateRoute – A route with this name already exists

Notes

If this is a method on a class, you will also need to use the `register_class_instance_for_routes` method for this to work as an IPC route.

CHAPTER

TWO

SERVER

class zonis.**Server**

Bases: *RouteHandler*

Parameters

- **using_fastapi_websockets** (bool) – Defaults to False.
- **override_key** (Optional[str]) –
- **secret_key** (str) – Defaults to an emptry string.

async disconnect(*identifier: str*) → None

Disconnect a client connection.

Parameters

identifier (*str*) – The client identifier to disconnect

Notes

This doesn't yet tell the client to stop gracefully, this just removes it from our store.

async request(*route: str*, *, *client_identifier: str = 'DEFAULT'*, ***kwargs*) → Any

Make a request to the provided IPC client.

Parameters

- **route** (*str*) – The IPC route to call.
- **client_identifier** (*Optional[str]*) – The client to make a request to.

  This only applies in many to one setups or if you changed the default identifier.
- **kwargs** – All the arguments you wish to invoke the IPC route with.

Returns

The data the IPC route returned.

Return type

Any

Raises

RequestFailed – The IPC request failed.

async request_all(*route: str*, ***kwargs*) → Dict[str, Any]

Issue a request to connected IPC clients.

Parameters

- **route** (*str*) – The IPC route to call.
- **kwargs** – All the arguments you wish to invoke the IPC route with.

Returns

A dictionary where the keys are the client identifiers and the values are the returned data.

The data could also be an instance of :py:class:RequestFailed:

Return type
Dict[str, Any]

async parse_identify(*packet: PacketT*, *websocket*) → str

Parse a packet to establish a new valid client connection.

Parameters

- **packet** (Packet) – The packet to read
- **websocket** – The websocket this connection is using

Returns
The established clients identifier

Return type
str

Raises

- ***BaseZonisException*** – Unexpected WS issue
- ***DuplicateConnection*** – Duplicate connection without override keys

load_routes() → None

Loads all decorated routes.

register_class_instance_for_routes(*instance*, **routes*) → None

Register a class instance for the given route.

When you turn a method on a class into an IPC route, you need to call this method with the instance of the class to use as well as the names of the routes for registration to work correctly.

Parameters

- **instance** – The class instance the methods live on
- **routes** – A list of strings representing the names of the IPC routes for this class.

route(*route_name: Optional[str] = None*)

Turn an async function into a valid IPC route.

Parameters
route_name (*Optional[str]*) – An optional name for this IPC route, defaults to the name of the function.

Raises
DuplicateRoute – A route with this name already exists

Notes

If this is a method on a class, you will also need to use the `register_class_instance_for_routes` method for this to work as an IPC route.

CHAPTER

THREE

ERRORS

class zonis.**BaseZonisException**

Bases: Exception

A base exception handler for Zonis.

class zonis.**DuplicateConnection**

Bases: *BaseZonisException*

You have attempted to connect with a duplicated identifier. Please try again with a unique one or provide the correct override key.

class zonis.**DuplicateRoute**

Bases: *BaseZonisException*

You are attempting to register multiple routes with the same name. Consider setting the route_name argument to something unique.

class zonis.**UnhandledWebsocketType**

Bases: *BaseZonisException*

Found a websocket type we can't handle.

class zonis.**UnknownRoute**

Bases: *BaseZonisException*

The route you requested does not exist.

class zonis.**UnknownClient**

Bases: *BaseZonisException*

The client you requested is not currently connected.

class zonis.**UnknownPacket**

Bases: *BaseZonisException*

Router received a packet without a known handler.

class zonis.**MissingReceiveHandler**

Bases: *BaseZonisException*

Cannot handle incoming requests due to a lack of receive handlers.

class zonis.**RequestFailed**

Bases: *BaseZonisException*

This request resulted in an error on the end client.

CHAPTER

FOUR

OTHER REFERENCES

class zonis.**WebsocketProtocol**

Bases: Protocol

The protocol underlying websockets should be exposed via.

async send(*content: str*) → None

Send a message down the wire.

Parameters
content (*str*) – The content to send down the wire.

async receive() → str | bytes | dict

Listen for a message on the wire.

Returns
The content received

Return type
str|bytes|dict

class zonis.**FastAPIWebsockets**

Bases: object

A websocket implementation wrapping FastAPI websockets

async send(*content: str*) → None

async receive() → str | bytes | dict

class zonis.**Websockets**

Bases: object

A websocket implementation wrapping the websockets library

async send(*content: str*) → None

async receive() → str | bytes | dict

CHAPTER

FIVE

PACKET DATA STRUCTURES

At the router level:

```
{
    "packet_id": "str",
    "type": "request|response",
    "data": {...}
}
```

The packets which can be sent at the client level:

class zonis.**Packet**

Bases: TypedDict

data: Any

type: Literal['IDENTIFY', 'REQUEST', 'SUCCESS_RESPONSE', 'FAILURE_RESPONSE', 'CLIENT_REQUEST', 'CLIENT_REQUEST_RESPONSE']

identifier: str

class zonis.**RequestPacket**

Bases: TypedDict

route: str

arguments: Dict[str, Any]

class zonis.**IdentifyDataPacket**

Bases: TypedDict

override_key: Optional[str]

secret_key: str

class zonis.**IdentifyPacket**

Bases: TypedDict

identifier: str

type: Literal['IDENTIFY']

data: *IdentifyDataPacket*

class zonis.**ClientToServerPacket**

Bases: TypedDict

identifier: str

type: Literal['CLIENT_REQUEST']

data: *RequestPacket*

CHAPTER

SIX

ROUTER

zonis.**route**(*route_name: Optional[str] = None*)

Turn an async function into a valid IPC route.

Parameters
route_name (*Optional[str]*) – An optional name for this IPC route, defaults to the name of the function.

Raises
DuplicateRoute – A route with this name already exists

class zonis.**Router**

Bases: object

A router class for enabling two-way communication down a single pipe.

This is built on the architecture ideals that if you want to send something down the pipe, then this class can return you the response just fine. However, if you wish to receive content for this class then you need to register an async method to handle that incoming data as this class doesn't have the context required to handle it.

async block_until_closed()

A blocking call which releases when the WS closes.

Notes

This will block even if the underlying WS has yet to connect.

async connect_client(*url: str*, *idp:* IdentifyDataPacket) → None

A blocking call which connects the underlying WS.

async connect_server() → None

async close() → None

Close the underlying WS

async send(*data: dict*) → Future

Send data down the pipe and receive a response.

Parameters
data (*dict*) – The packet to send down the pipe

Returns
This future will contain the returned data once it becomes available

Return type
asyncio.Future

async send_response(*, *packet_id: str*, *data: Optional[dict] = None*)

Sends a response to a received request without waiting anything.

Parameters

- **packet_id** (*str*) – The packet ID to respond to
- **data** (*dict*) – The data being responded with

register_receiver(*callback*) → *Router*

Register the provided callback to handle packets with the request type.

Parameters

callback – The callback to call with the packet data.

class zonis.**RouteHandler**

Bases: object

route(*route_name: Optional[str] = None*)

Turn an async function into a valid IPC route.

Parameters

route_name (*Optional[str]*) – An optional name for this IPC route, defaults to the name of the function.

Raises

DuplicateRoute – A route with this name already exists

Notes

If this is a method on a class, you will also need to use the `register_class_instance_for_routes` method for this to work as an IPC route.

load_routes() → None

Loads all decorated routes.

register_class_instance_for_routes(*instance*, **routes*) → None

Register a class instance for the given route.

When you turn a method on a class into an IPC route, you need to call this method with the instance of the class to use as well as the names of the routes for registration to work correctly.

Parameters

- **instance** – The class instance the methods live on
- **routes** – A list of strings representing the names of the IPC routes for this class.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search

INDEX

A

arguments (*zonis.RequestPacket attribute*), 13

B

BaseZonisException (*class in zonis*), 9
block_until_closed() (*zonis.Client method*), 3
block_until_closed() (*zonis.Router method*), 15

C

Client (*class in zonis*), 3
ClientToServerPacket (*class in zonis*), 13
close() (*zonis.Client method*), 3
close() (*zonis.Router method*), 15
connect_client() (*zonis.Router method*), 15
connect_server() (*zonis.Router method*), 15

D

data (*zonis.ClientToServerPacket attribute*), 14
data (*zonis.IdentifyPacket attribute*), 13
data (*zonis.Packet attribute*), 13
disconnect() (*zonis.Server method*), 5
DuplicateConnection (*class in zonis*), 9
DuplicateRoute (*class in zonis*), 9

F

FastAPIWebsockets (*class in zonis*), 11

I

identifier (*zonis.ClientToServerPacket attribute*), 13
identifier (*zonis.IdentifyPacket attribute*), 13
identifier (*zonis.Packet attribute*), 13
IdentifyDataPacket (*class in zonis*), 13
IdentifyPacket (*class in zonis*), 13

L

load_routes() (*zonis.Client method*), 3
load_routes() (*zonis.RouteHandler method*), 16
load_routes() (*zonis.Server method*), 6

M

MissingReceiveHandler (*class in zonis*), 9

O

override_key (*zonis.IdentifyDataPacket attribute*), 13

P

Packet (*class in zonis*), 13
parse_identify() (*zonis.Server method*), 6

R

receive() (*zonis.FastAPIWebsockets method*), 11
receive() (*zonis.WebsocketProtocol method*), 11
receive() (*zonis.Websockets method*), 11
register_class_instance_for_routes() (*zonis.Client method*), 3
register_class_instance_for_routes() (*zonis.RouteHandler method*), 16
register_class_instance_for_routes() (*zonis.Server method*), 6
register_receiver() (*zonis.Router method*), 16
request() (*zonis.Client method*), 3
request() (*zonis.Server method*), 5
request_all() (*zonis.Server method*), 5
RequestFailed (*class in zonis*), 9
RequestPacket (*class in zonis*), 13
route (*zonis.RequestPacket attribute*), 13
route() (*in module zonis*), 15
route() (*zonis.Client method*), 3
route() (*zonis.RouteHandler method*), 16
route() (*zonis.Server method*), 6
RouteHandler (*class in zonis*), 16
Router (*class in zonis*), 15

S

secret_key (*zonis.IdentifyDataPacket attribute*), 13
send() (*zonis.FastAPIWebsockets method*), 11
send() (*zonis.Router method*), 15
send() (*zonis.WebsocketProtocol method*), 11
send() (*zonis.Websockets method*), 11
send_response() (*zonis.Router method*), 15
Server (*class in zonis*), 5
start() (*zonis.Client method*), 3

T

type (*zonis.ClientToServerPacket attribute*), 14
type (*zonis.IdentifyPacket attribute*), 13
type (*zonis.Packet attribute*), 13

U

UnhandledWebsocketType (*class in zonis*), 9
UnknownClient (*class in zonis*), 9
UnknownPacket (*class in zonis*), 9
UnknownRoute (*class in zonis*), 9

W

WebsocketProtocol (*class in zonis*), 11
Websockets (*class in zonis*), 11