

---

# **discord-ext-ipcs**

***Release v0.0.5***

**mc-fdc**

**Jun 12, 2022**



CONTENTS:

<b>1</b>		<b>1</b>
1.1	.....	1
1.2	.....	1
1.3	.....	1
<b>2</b>	<b>api reference</b>	<b>3</b>
2.1	Submodules .....	3
2.2	discord.ext.ipcs.client module .....	3
2.3	discord.ext.ipcs.errors module .....	5
2.4	discord.ext.ipcs.items module .....	5
<b>3</b>	<b>Indices and tables</b>	<b>7</b>
	<b>Python Module Index</b>	<b>9</b>
	<b>Index</b>	<b>11</b>



## 1.1

websocket

## 1.2

```
{  
  "type": "",  
  "data": {}  
}
```

## 1.3

```
{  
  "type": "login",  
  "data": {  
    "password": ""  
  }  
}
```

```
{  
  "type": "ready",  
  "data": {}  
}
```

websocket



## API REFERENCE

### 2.1 Submodules

### 2.2 discord.ext.ipcs.client module

**class** discord.ext.ipcs.client.**Client**(*client: Client, secret\_key: str, \*, loop: Optional[AbstractEventLoop] = None, log: Optional[bool] = False*)

Bases: object

ipc client

#### Parameters

- **client** (*discord.Client*) – Discord client
- **secret\_key** (*str*) – secret key
- **loop** (*Optional[asyncio.AbstractEventLoop]*) – Event loop
- **log** (*Optional[bool]*) – show log

**async** **close**(*code: Optional[int] = 1000, message: Optional[str] = 'Bye'*) → None

Close from ipc server

#### Parameters

- **code** (*Optional[int]*) – Close code
- **message** (*Optional[str]*) – Close message

#### Raises

**ConnectionError** – If you already close, it will be raise.

#### Examples

```
await ipc_client.close()
```

**async** **connect**(*uri: str, \*, reconnect: Optional[bool] = True*) → None

Connect to ipc server

#### Parameters

- **uri** (*str*) – URI
- **reconnect** (*Optional[bool]*) – Reconnect when it was closed.

## Examples

`await ipc_client.connect("ws://localhost/ipc")`

### Raises

**`ConnectionError`** – If you already connect, it will be raise.

**`dispatch(eventtype: str, response: ResponseItem)`** → None

Run function

### Parameters

- **`eventtype`** (*str*) – Event type
- **`response`** (*ResponseItem*) – response item

**`listen(eventtype: str)`**

Catch data from ipc server

### Parameters

**`eventtype`** (*str*) – Event type.

**`login()`**

Login to ipc server

## Examples

`await ipc_client.login()`

**`print(content: str)`** → None

This can print like sanic

### Parameters

**`content`** (*str*) – content

**`async reconnect()`** → None

Reconnect from ipc server

## Examples

`await ipc_client.reconnect()`

**`async recv()`** → None

Get data and dispatch

**`async request(eventtype: str, data: Optional[dict] = {})`** → None

Send something to ipc server

### Parameters

- **`eventtype`** (*str*) – event type
- **`data`** (*Optional[dict]*) – The data you want to send.



### Examples

```
await ipc_client.request("hello", {"message": "What your name?"})
```

## 2.3 discord.ext.ipcs.errors module

**exception** discord.ext.ipcs.errors.**AsyncError**

Bases: Exception

**exception** discord.ext.ipcs.errors.**ConnectionError**

Bases: Exception

## 2.4 discord.ext.ipcs.items module

**class** discord.ext.ipcs.items.**ResponseItem**(*data: dict*)

Bases: object

Response data

**Parameters**

**data** (*dict*) – Some kind of data.

### Examples

```
response = ResponseItem({"hello": "world", "message": "What your name?"}) print(response.hello)
```



## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



## PYTHON MODULE INDEX

### d

`discord.ext.ipcs.client`, 3  
`discord.ext.ipcs.errors`, 5  
`discord.ext.ipcs.items`, 5



## INDEX

### A

`AsyncError`, 5

### C

`Client` (*class in discord.ext.ipcs.client*), 3

`close()` (*discord.ext.ipcs.client.Client method*), 3

`connect()` (*discord.ext.ipcs.client.Client method*), 3

`ConnectionError`, 5

### D

`discord.ext.ipcs.client`  
module, 3

`discord.ext.ipcs.errors`  
module, 5

`discord.ext.ipcs.items`  
module, 5

`dispatch()` (*discord.ext.ipcs.client.Client method*), 4

### L

`listen()` (*discord.ext.ipcs.client.Client method*), 4

`login()` (*discord.ext.ipcs.client.Client method*), 4

### M

module

`discord.ext.ipcs.client`, 3

`discord.ext.ipcs.errors`, 5

`discord.ext.ipcs.items`, 5

### P

`print()` (*discord.ext.ipcs.client.Client method*), 4

### R

`reconnect()` (*discord.ext.ipcs.client.Client method*), 4

`recv()` (*discord.ext.ipcs.client.Client method*), 4

`request()` (*discord.ext.ipcs.client.Client method*), 4

`ResponseItem` (*class in discord.ext.ipcs.items*), 5