

---

# **Mycroft Simple Documentation**

***Release 0.1.0***

**Matthew Scholefield**

**Jun 23, 2017**



---

## Contents:

---

<b>1</b>	<b>Subpackages</b>	<b>3</b>
1.1	mycroft.clients package . . . . .	3
1.1.1	mycroft.clients.mycroft_client module . . . . .	3
1.1.2	mycroft.clients.text_client module . . . . .	3
1.2	mycroft.engines package . . . . .	4
1.2.1	mycroft.engines.intent_engine module . . . . .	4
1.2.2	mycroft.engines.padatious_engine module . . . . .	4
1.3	mycroft.formats package . . . . .	5
1.3.1	mycroft.formats.dialog_format module . . . . .	5
1.3.2	mycroft.formats.mycroft_format module . . . . .	5
1.4	mycroft.managers package . . . . .	5
1.4.1	mycroft.managers.client_manager module . . . . .	5
1.4.2	mycroft.managers.format_manager module . . . . .	5
1.4.3	mycroft.managers.intent_manager module . . . . .	6
1.4.4	mycroft.managers.path_manager module . . . . .	6
1.4.5	mycroft.managers.query_manager module . . . . .	7
1.4.6	mycroft.managers.skill_manager module . . . . .	7
1.5	mycroft.skills package . . . . .	7
1.5.1	Subpackages . . . . .	7
1.5.2	mycroft.skills.mycroft_skill module . . . . .	8
<b>2</b>	<b>mycroft.mycroft_thread module</b>	<b>9</b>
<b>3</b>	<b>mycroft.util module</b>	<b>11</b>
	<b>Python Module Index</b>	<b>13</b>



*Mycroft, made simple*

This repository contains experimental code restructuring with things like Padatious integration.



### mycroft.clients package

#### mycroft.clients.mycroft\_client module

**class** `mycroft.clients.mycroft_client.MycroftClient` (*query\_manager*)

Bases: `object`

Provides common behavior like sending and receiving queries Examples clients include the voice client and text client

**on\_response** (*format\_manager*)

Called after `send_query`. Use `format_manager` to get outputted response

**quit** ()

Should send a signal to stop the main thread of the client

**run** ()

Executes the main thread for the client

**send\_query** (*query*)

Ask a question and trigger `on_response` when an answer is found

#### mycroft.clients.text\_client module

**class** `mycroft.clients.text_client.TextClient` (*\*args, \*\*kwargs*)

Bases: `mycroft.clients.mycroft_client.MycroftClient`

Interact with Mycroft via a terminal

**on\_response** (*format\_manager*)

**quit** ()

**run** ()

## mycroft.engines package

### mycroft.engines.intent\_engine module

```
class mycroft.engines.intent_engine.IntentEngine (path_manager)
    Bases: object

    Interface for intent engines

    calc_intents (query)
        Run the intent engine to determine the probability of each intent against the query :param query: input
        sentence as a single string :return: dict of intent: intent_data where

        Example return data: { 'name': 'TimeSkill:time.ask', 'confidence': '0.65', 'matches': { 'location': 'new
        york' } }

    on_intents_loaded ()
        Override to run code when all intents have been registered

    try_register_intent (*args, **kwargs)
        Attempt to register intent with given arguments :rtype str :returns intent name if parsed parameters, other-
        wise ""

mycroft.engines.intent_engine.extract_intent_name (namespaced_name)
    Ex. TimeSkill:time.ask -> time.ask

mycroft.engines.intent_engine.extract_skill_name (namespaced_name)
    Ex. TimeSkill:time.ask -> TimeSkill

mycroft.engines.intent_engine.make_namespaced (intent_name, skill_name)
    Mangle the intent name so that it doesn't conflict and to save the skill name in the same string
```

### mycroft.engines.padatious\_engine module

```
class mycroft.engines.padatious_engine.PadatiousEngine (path_manager)
    Bases: mycroft.engines.intent_engine.IntentEngine

    Interface for Padatious intent engine

    GIT_BRANCH = 'feature/mycroft-simple'

    GIT_URL = 'https://github.com/MatthewScholefield/padatious-mycroft.git'

    HOST = '127.0.0.1'

    PORT = 8014

    calc_intents (query)

    on_intents_loaded ()

    try_register_intent (skill_name, intent_name)
```



## mycroft.formats package

### mycroft.formats.dialog\_format module

**class** `mycroft.formats.dialog_format.DialogFormat` (*path\_manager*)

Bases: `mycroft.formats.mycroft_format.MycroftFormat`

Format data into sentences

**generate** (*name, results*)

### mycroft.formats.mycroft\_format module

**class** `mycroft.formats.mycroft_format.MycroftFormat` (*path\_manager*)

Bases: `object`

Base class to provide an interface for different types of “formats”

Formats are modes to display key-value data. For instance, the DialogFormat puts data into sentences The EnclosureFormat could put data into visual faceplate animations

**generate** (*name, results*)

Internally format the data from the results Depending on the format, this can be accessed different ways

#### Parameters

- **name** – namespaced intent name
- **results** – dict containing all data from the skill

**Returns** nothing

## mycroft.managers package

### mycroft.managers.client\_manager module

**class** `mycroft.managers.client_manager.ClientManager` (*client\_classes, \*args, \*\*kwargs*)

Bases: `object`

Holds all clients to start and stop them

**quit** ()

Sends a signal to all clients to quit. Does not wait for clients to exit (non blocking)

**start** ()

Starts all clients in different threads (non blocking)

### mycroft.managers.format\_manager module

**class** `mycroft.managers.format_manager.FormatManager` (*path\_manager*)

Bases: `object`

Holds all formats and provides an interface to access them

**as\_dialog**

Get data as a sentence

**generate** (*name, results*)

## mycroft.managers.intent\_manager module

**class** mycroft.managers.intent\_manager.**IntentManager** (*path\_manager*)

Bases: object

Used to handle creating both intents and intent engines

**calc\_results** (*query*)

Find the best intent and run the handler to find the results

**Parameters** **query** – input sentence

**Returns** name, results

**Rtype name** string (namespaced intent)

**Rtype results** dict

**on\_intents\_loaded** ()

**register\_fallback** (*handler*)

Register a function to be called as a general knowledge fallback

**Parameters** **handler** – function that receives query and returns a dict of results, one of which is ‘confidence’ note: register\_fallback in the MycroftSkill base class automatically manages results

**register\_intent** (*skill\_name, intent, handler*)

Register an intent via the corresponding intent engine It tries passing the arguments to each engine until one can interpret it correctly

**Parameters**

- **skill\_name** –
- **intent** – argument used to build intent; can be anything
- **handler** – function that receives intent\_data and returns a dict of results note: register\_intent in the MycroftSkill base class automatically manages results

**Returns** nothing

## mycroft.managers.path\_manager module

**class** mycroft.managers.path\_manager.**PathManager** (*base\_path*)

Bases: object

Retreives directories and files used by Mycroft

**dialog\_dir** (*skill\_name*)

**intent\_dir** (*skill\_name*)

**mod\_path**

**padatious\_exe**

The locally compiled Padatious executable

**skill\_dir** (*skill\_name*)

**skills\_dir**

**vocab\_dir** (*skill\_name*)

## mycroft.managers.query\_manager module

**class** `mycroft.managers.query_manager.QueryManager` (*intent\_manager, format\_manager*)

Bases: `object`

Launches queries in separate threads

**on\_response** (*callback*)

Assign a callback to be run whenever a new response comes in

**send\_query** (*query*)

Starts calculating a query in a new thread

## mycroft.managers.skill\_manager module

**class** `mycroft.managers.skill_manager.SkillManager` (*intent\_manager, path\_manager*)

Bases: `object`

Dynamically loads skills

**load\_skills** ()

Looks in the skill folder and loads the CamelCase equivalent class of the snake case folder This class should be inside the skill.py file. Example:

**skills/**

**time\_skill/** skill.py - class TimeSkill(MycroftSkill):

**weather\_skill/** skill.py - class WeatherSkill(MycroftSkill):

## mycroft.skills package

### Subpackages

**mycroft.skills.duck\_duck\_go\_skill package**

**mycroft.skills.duck\_duck\_go\_skill.skill module**

**class** `mycroft.skills.duck_duck_go_skill.skill.DuckDuckGoSkill` (*\*args, \*\*kwargs*)

Bases: `mycroft.skills.mycroft_skill.MycroftSkill`

Fallback skill that queries DuckDuckGo's instant answer API

**fallback** (*query*)

**fallback\_no\_question** (*query*)

**mycroft.skills.quit\_skill package**

### mycroft.skills.quit\_skill.skill module

```
class mycroft.skills.quit_skill.skill.QuitSkill(*args, **kwargs)
    Bases: mycroft.skills.mycroft_skill.MycroftSkill
```

### mycroft.skills.time\_skill package

#### mycroft.skills.time\_skill.skill module

```
class mycroft.skills.time_skill.skill.TimeSkill(*args, **kwargs)
    Bases: mycroft.skills.mycroft_skill.MycroftSkill

    date (intent_data)
    time (intent_data)
```

### mycroft.skills.unknown\_skill package

#### mycroft.skills.unknown\_skill.skill module

```
class mycroft.skills.unknown_skill.skill.UnknownSkill(*args, **kwargs)
    Bases: mycroft.skills.mycroft_skill.MycroftSkill

    calc_results (intent_data)
```

### mycroft.skills.mycroft\_skill module

```
class mycroft.skills.mycroft_skill.MycroftSkill(intent_manager)
    Bases: object
```

Base class for all Mycroft skills

**add\_result** (*key, value*)

Adds a result from the skill. For example:

**self.add\_result('time', '11:45 PM')** Except, of course, '11:45 PM' would be something generated from an API

**Results can be both general and granular. Another example:** **self.add\_result('time\_seconds', 23)**

**create\_handler** (*handler, skill\_name=None*)

Wrap the skill handler to return added results

**register\_fallback** (*handler*)

Same as register\_intent except the handler only receives a query and is only activated when all other Mycroft intents fail

**register\_intent** (*name, handler*)

Set a function to be called when the intent called 'name' is activated In this handler the skill should receive a dict called intent\_data and call self.add\_result() to add output data. Nothing should be returned from the handler

## CHAPTER 2

---

### mycroft.mycroft\_thread module

---

```
mycroft.mycroft_thread.quit()  
mycroft.mycroft_thread.set_quit_action(callback)
```



## CHAPTER 3

---

### mycroft.util module

---

`mycroft.util.split_sentences` (*text*)

Turns a string of multiple sentences into a list of separate ones As a side effect, .?! at the end of a sentence are removed

`mycroft.util.to_camel` (*snake*)

time\_skill -> TimeSkill

`mycroft.util.to_snake` (*camel*)

TimeSkill -> time\_skill





### m

`mycroft.clients.mycroft_client`, 3  
`mycroft.clients.text_client`, 3  
`mycroft.engines.intent_engine`, 4  
`mycroft.engines.pاداتious_engine`, 4  
`mycroft.formats.dialog_format`, 5  
`mycroft.formats.mycroft_format`, 5  
`mycroft.managers.client_manager`, 5  
`mycroft.managers.format_manager`, 5  
`mycroft.managers.intent_manager`, 6  
`mycroft.managers.path_manager`, 6  
`mycroft.managers.query_manager`, 7  
`mycroft.managers.skill_manager`, 7  
`mycroft.mycroft_thread`, 9  
`mycroft.skills.duck_duck_go_skill.skill`,  
7  
`mycroft.skills.mycroft_skill`, 8  
`mycroft.skills.quit_skill.skill`, 8  
`mycroft.skills.time_skill.skill`, 8  
`mycroft.skills.unknown_skill.skill`, 8  
`mycroft.util`, 11



## A

add\_result() (mycroft.skills.mycroft\_skill.MycroftSkill method), 8

as\_dialog (mycroft.managers.format\_manager.FormatManager attribute), 5

fallback\_no\_question() (mycroft.skills.duck\_duck\_go\_skill.skill.DuckDuckGoSkill method), 7

FormatManager (class in mycroft.managers.format\_manager), 5

## C

calc\_intents() (mycroft.engines.intent\_engine.IntentEngine method), 4

calc\_intents() (mycroft.engines.padatious\_engine.PadatiousEngine method), 4

calc\_results() (mycroft.managers.intent\_manager.IntentManager method), 6

calc\_results() (mycroft.skills.unknown\_skill.skill.UnknownSkill method), 8

ClientManager (class in mycroft.managers.client\_manager), 5

create\_handler() (mycroft.skills.mycroft\_skill.MycroftSkill method), 8

generate() (mycroft.formats.dialog\_format.DialogFormat method), 5

generate() (mycroft.formats.mycroft\_format.MycroftFormat method), 5

generate() (mycroft.managers.format\_manager.FormatManager method), 5

GIT\_BRANCH (mycroft.engines.padatious\_engine.PadatiousEngine attribute), 4

GIT\_URL (mycroft.engines.padatious\_engine.PadatiousEngine attribute), 4

## G

## H

HOST (mycroft.engines.padatious\_engine.PadatiousEngine attribute), 4

## D

date() (mycroft.skills.time\_skill.skill.TimeSkill method), 8

dialog\_dir() (mycroft.managers.path\_manager.PathManager method), 6

DialogFormat (class in mycroft.formats.dialog\_format), 5

DuckDuckGoSkill (class in mycroft.skills.duck\_duck\_go\_skill.skill), 7

intent\_dir() (mycroft.managers.path\_manager.PathManager method), 6

IntentEngine (class in mycroft.engines.intent\_engine), 4

IntentManager (class in mycroft.managers.intent\_manager), 6

## E

extract\_intent\_name() (in module mycroft.engines.intent\_engine), 4

extract\_skill\_name() (in module mycroft.engines.intent\_engine), 4

## F

fallback() (mycroft.skills.duck\_duck\_go\_skill.skill.DuckDuckGoSkill method), 7

## L

load\_skills() (mycroft.managers.skill\_manager.SkillManager method), 7

## M

make\_namespaced() (in module mycroft.engines.intent\_engine), 4

not\_GitSkill (mycroft.managers.path\_manager.PathManager attribute), 6

mycroft.clients.mycroft\_client (module), 3

mycroft.clients.text\_client (module), 3  
 mycroft.engines.intent\_engine (module), 4  
 mycroft.engines.padiatious\_engine (module), 4  
 mycroft.formats.dialog\_format (module), 5  
 mycroft.formats.mycroft\_format (module), 5  
 mycroft.managers.client\_manager (module), 5  
 mycroft.managers.format\_manager (module), 5  
 mycroft.managers.intent\_manager (module), 6  
 mycroft.managers.path\_manager (module), 6  
 mycroft.managers.query\_manager (module), 7  
 mycroft.managers.skill\_manager (module), 7  
 mycroft.mycroft\_thread (module), 9  
 mycroft.skills.duck\_duck\_go\_skill.skill (module), 7  
 mycroft.skills.mycroft\_skill (module), 8  
 mycroft.skills.quit\_skill.skill (module), 8  
 mycroft.skills.time\_skill.skill (module), 8  
 mycroft.skills.unknown\_skill.skill (module), 8  
 mycroft.util (module), 11  
 MycroftClient (class in mycroft.clients.mycroft\_client), 3  
 MycroftFormat (class in mycroft.formats.mycroft\_format), 5  
 MycroftSkill (class in mycroft.skills.mycroft\_skill), 8

## O

on\_intents\_loaded() (mycroft.engines.intent\_engine.IntentEngine method), 4  
 on\_intents\_loaded() (mycroft.engines.padiatious\_engine.PadiatiousEngine method), 4  
 on\_intents\_loaded() (mycroft.managers.intent\_manager.IntentManager method), 6  
 on\_response() (mycroft.clients.mycroft\_client.MycroftClient method), 3  
 on\_response() (mycroft.clients.text\_client.TextClient method), 3  
 on\_response() (mycroft.managers.query\_manager.QueryManager method), 7

## P

padiatious\_exe (mycroft.managers.path\_manager.PathManager attribute), 6  
 PadiatiousEngine (class in mycroft.engines.padiatious\_engine), 4  
 PathManager (class in mycroft.managers.path\_manager), 6  
 PORT (mycroft.engines.padiatious\_engine.PadiatiousEngine attribute), 4

## Q

QueryManager (class in mycroft.managers.query\_manager), 7  
 quit() (in module mycroft.mycroft\_thread), 9

quit() (mycroft.clients.mycroft\_client.MycroftClient method), 3  
 quit() (mycroft.clients.text\_client.TextClient method), 3  
 quit() (mycroft.managers.client\_manager.ClientManager method), 5  
 QuitSkill (class in mycroft.skills.quit\_skill.skill), 8

## R

register\_fallback() (mycroft.managers.intent\_manager.IntentManager method), 6  
 register\_fallback() (mycroft.skills.mycroft\_skill.MycroftSkill method), 8  
 register\_intent() (mycroft.managers.intent\_manager.IntentManager method), 6  
 register\_intent() (mycroft.skills.mycroft\_skill.MycroftSkill method), 8  
 run() (mycroft.clients.mycroft\_client.MycroftClient method), 3  
 run() (mycroft.clients.text\_client.TextClient method), 3

## S

send\_query() (mycroft.clients.mycroft\_client.MycroftClient method), 3  
 send\_query() (mycroft.managers.query\_manager.QueryManager method), 7  
 set\_quit\_action() (in module mycroft.mycroft\_thread), 9  
 skill\_dir() (mycroft.managers.path\_manager.PathManager method), 6  
 SkillManager (class in mycroft.managers.skill\_manager), 7  
 skills\_dir (mycroft.managers.path\_manager.PathManager attribute), 6  
 split\_sentences() (in module mycroft.util), 11  
 start() (mycroft.managers.client\_manager.ClientManager method), 5

## T

TextClient (class in mycroft.clients.text\_client), 3  
 time() (mycroft.skills.time\_skill.skill.TimeSkill method), 8  
 TimeSkill (class in mycroft.skills.time\_skill.skill), 8  
 to\_camel() (in module mycroft.util), 11  
 to\_snake() (in module mycroft.util), 11  
 try\_register\_intent() (mycroft.engines.intent\_engine.IntentEngine method), 4  
 try\_register\_intent() (mycroft.engines.padiatious\_engine.PadiatiousEngine method), 4

## U

UnknownSkill (class in my-croft.skills.unknown\_skill.skill), [8](#)

## V

vocab\_dir() (mycroft.managers.path\_manager.PathManager method), [6](#)