

# REST

September 18, 2019

## 1 Install and use CherryPy

### 1.1 Install

To install CherryPy just open a terminal and type *"pip3 install cherrypy"*

### 1.2 "Hello World"

Let's create a new file called *"helloWorld.py"* and paste the following code inside it

```
[ ]: import cherrypy

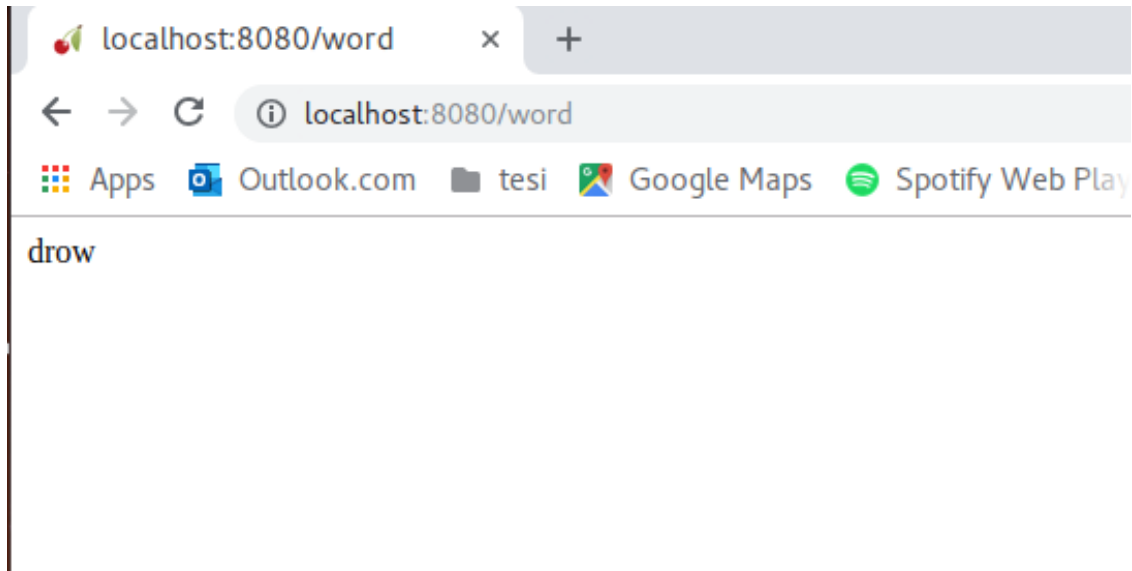
class HelloWorld(object):
    exposed=True
    def GET(self,*uri,**params):
        #Standard output
        output="Hello World"
        #Check the uri in the requests
        #<br> is just used to append the content in a new line
        #(<br> is the \n for HTML)
        if len(uri)!=0:
            output+='<br>uri: '+','.join(uri)
        #Check the parameters in the request
        #<br> is just used to append the content in a new line
        #(<br> is the \n for HTML)
        if params!={}:
            output+='<br>params: '+str(params)
        return output

if __name__=="__main__":
    #Standard configuration to serve the url "localhost:8080"
    conf={
        '/':{
            'request.dispatch':cherrypy.dispatch.MethodDispatcher(),
            'tool.session.on':True
        }
    }
```

```
cherry.py.quickstart>HelloWorld(), '/', conf)
```

### 1.3 Exercise 1

We want to create a web service that is able to read the uri we send as GET request and return the string reversed, the result is shown in the image below



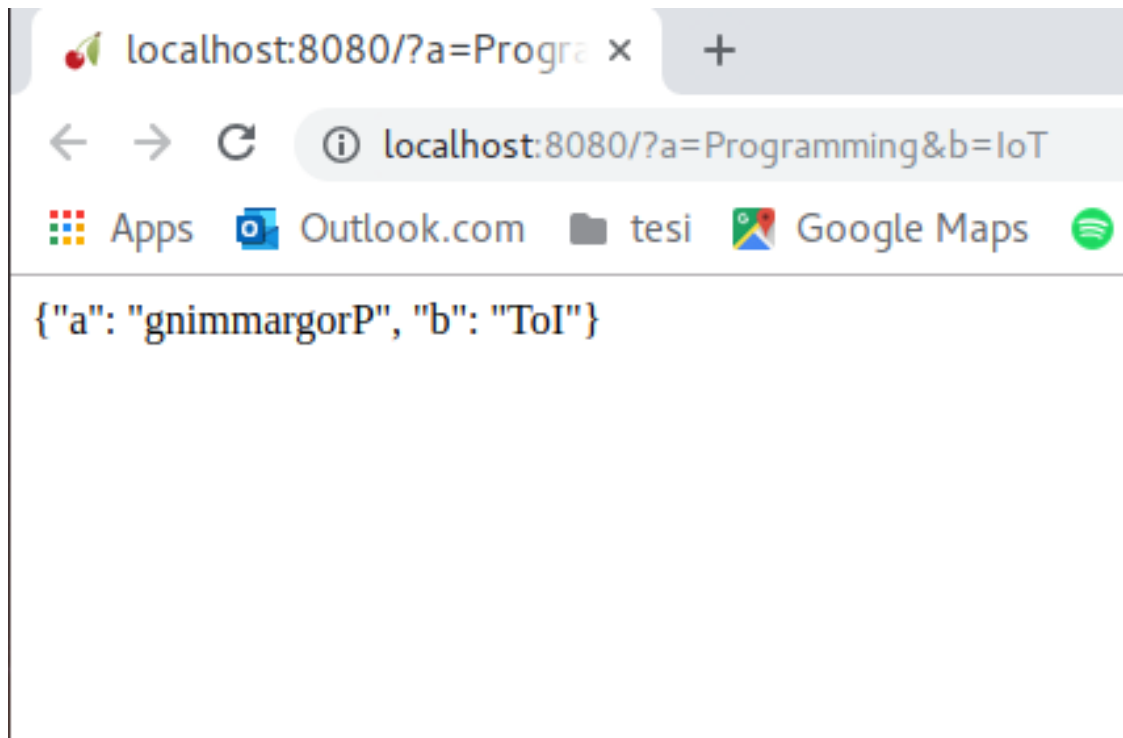
#### Tips and tricks

1. Once you've created your class you can use the code below inside the "main" function to configure the server (pay attention to the indentation!)

```
if __name__ == '__main__':
    conf={
        '/':{
            'request.dispatch':cherry.py.dispatch.MethodDispatcher(),
            'tool.session.on':True
        }
    }
    cherry.py.tree.mount(MyWebService(), '/', conf)
    cherry.py.engine.start()
    cherry.py.engine.block()
```

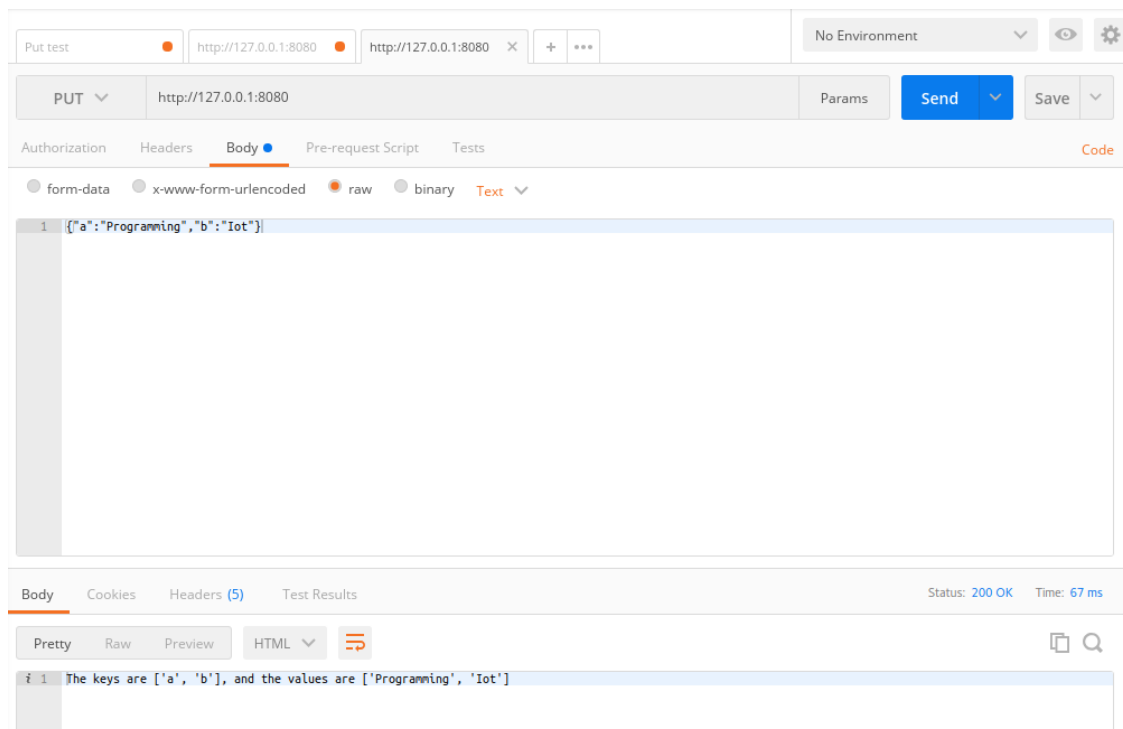
### 1.4 Exercise 2

Now let's create a web service similar to the one before but instead we want to pass the string to revert as parameter GET request. In this case the result should be a json that look like this:



### 1.5 Exercise 3

For this exercise we want to create a web services that receive with a PUT request with a json in the body and send us back the list of keys and the list of value of that json like in the picture below.



## Tips and tricks

1. You can use "Postman" available at [this link](#) to perform a PUT request to your service
2. When you create the json for the body remember to use double quotes (") to indicate the keys and the values to avoid errors.

## 1.6 Exercise 4

With this exercise we will try to become familiar with REST APIs. At [this link](#) you can find a list of free API, to make this exercise choose one of them that doesn't not require authentication (Auth=No). According to the one you choose you've to make a simple client to interact with it and do some basic search (try to find one that provides json as response and not images). Let's make an example. Let's chose the [exchange-rate api](#) and see how the API looks like The three main entry point of the API are:

1. **`https://api.exchangeratesapi.io/latest?base=...`**  
With this we need to specify the base currency to obtain the exchange rate
2. **`https://api.exchangeratesapi.io/YYYY-MM-YY`**  
To obtain the rate of a particular day
3. **`https://api.exchangeratesapi.io/history?start_at=YYYY-MM-YY&end_at=YYYY-MM-YY`**  
To obtain the rates in the specified interval

Knowing this we want to create a client that allows 3 function:

1. Obtain the latest exchange rates by specifying the base currency
2. Obtain the exchange rates of a specific day
3. Obtain the rates in an interval of days

The client looks like this:

```
[1]: run Exercise_4.py
```

Available command:

latest:latest change rate

history: historic exchange rates

quit:exit

latest

Which base currency you want:

E:Euro

U:USD

P:GBP

E

```
{
  "rates": {
    "CAD": 1.462,
    "HKD": 8.6266,
    "ISK": 136.6,
    "PHP": 57.712,
```

```

        "DKK": 7.4671,
        "HUF": 333.96,
        "CZK": 25.916,
        "AUD": 1.6122,
        "RON": 4.734,
        "SEK": 10.699,
        "IDR": 15536.74,
        "INR": 79.147,
        "BRL": 4.5207,
        "RUB": 70.7241,
        "HRK": 7.397,
        "JPY": 119.23,
        "THB": 33.701,
        "CHF": 1.0966,
        "SGD": 1.5175,
        "PLN": 4.3416,
        "BGN": 1.9558,
        "TRY": 6.3065,
        "CNY": 7.8249,
        "NOK": 9.8715,
        "NZD": 1.7426,
        "ZAR": 16.278,
        "USD": 1.1026,
        "MXN": 21.4919,
        "ILS": 3.9227,
        "GBP": 0.88813,
        "KRW": 1311.65,
        "MYR": 4.6149
    },
    "base": "EUR",
    "date": "2019-09-17"
}

```

Available command:

latest:latest change rate

history: historic exchange rates

quit:exit

history

Type D for a day and I for and interval

D

Write the year

2019

Write the month

06

Write the day

26

```

{
    "rates": {

```

```

        "CAD": 1.4947,
        "HKD": 8.8724,
        "ISK": 141.5,
        "PHP": 58.456,
        "DKK": 7.4651,
        "HUF": 323.5,
        "CZK": 25.486,
        "AUD": 1.6277,
        "RON": 4.722,
        "SEK": 10.5435,
        "IDR": 16097.68,
        "INR": 78.5705,
        "BRL": 4.3624,
        "RUB": 71.6399,
        "HRK": 7.3956,
        "JPY": 122.4,
        "THB": 34.955,
        "CHF": 1.1113,
        "SGD": 1.5387,
        "PLN": 4.2627,
        "BGN": 1.9558,
        "TRY": 6.55,
        "CNY": 7.8139,
        "NOK": 9.6733,
        "NZD": 1.7004,
        "ZAR": 16.2802,
        "USD": 1.1362,
        "MXN": 21.7972,
        "ILS": 4.0825,
        "GBP": 0.89603,
        "KRW": 1312.86,
        "MYR": 4.7124
    },
    "base": "EUR",
    "date": "2019-06-26"
}

```

Available command:

```

latest:latest change rate
history: historic exchange rates
quit:exit
quit

```

[ ]: