ExtraREST

October 28, 2019

1 Extra

1.1 Response with .html files

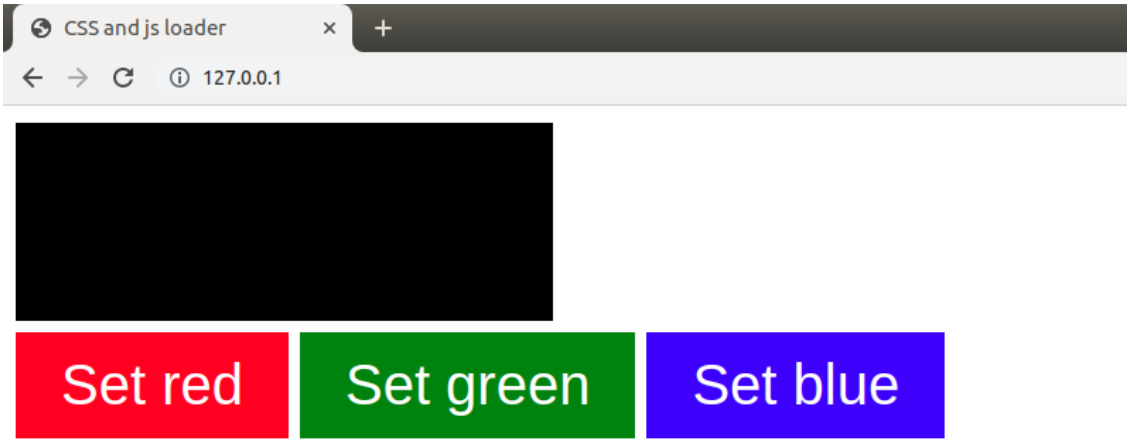
In most of the cases we want that our server reponds to the request with html files instead of strings, in order to have a proper formatting. To do this with CherryPy we can do as written in the code below

[ ]:
```
import cherrypy
import os

class Example(object):
        """docstring for Example"""
        exposed=True
        def __init__(self):
                self.id=1
        def GET(self):
                return open("index.html")

if __name__ == '__main__':
        conf={
                '/':{
                                'request.dispatch':cherrypy.dispatch.
 ↪MethodDispatcher(),
                                'tools.staticdir.root': os.path.abspath(os.
 ↪getcwd()),
                        }
        }
        cherrypy.tree.mount(Example(),'/',conf)
        cherrypy.engine.start()
        cherrypy.engine.block()
```

As you can see we can just return the result of the open function on the .html file (in the example *open('index.html')*).

1.2 Add css and javascript

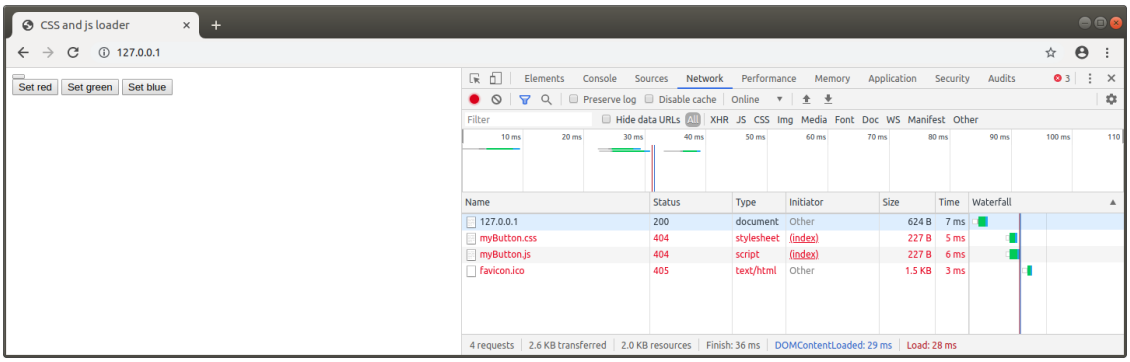
Sometimes we want to give a nicer look to our web page and maybe add some functions. The "styling" of the web page can be done with .css files while the function are written in javascript inside .js files. In the material you can find the file "*index.html*" and two folder called "*css*" and "*js*". These script contains all the code to obtain a web page that look like the one below, where clicking on a button set the color of the top rectangle to the selected one.



Inside the material you can find a zip file containing all the things you need to do what follow.

1. Copy the code above in the file *"main.py"* of the zip and launch it.
2. Then open a browser and activate the window "Developer Tools" (you can find it on the menu on the top right of your browser or you can press Ctrl+shift+i or Cmd+shift+i).
3. Once you've done that go on the "Network" tab of the developer tools window
4. than you can finally go on your website at the address 127.0.0.1:8080.

The result should be something like in the image below



As you can see from the error 404 in the *Network* panel of the *Developer Tools*, the browser is not able to find the stylesheet *"myButton.css"* that defines the colors of our button, it also can't find the file *"myButton.js"* that define the behaviour of the buttons when clicked. In order to avoid this errors we must configure our server in CherryPy in order to indicate where this file can be found.

Before doing this we want to know where our browser is searching this file. To do so, we can open our *"index.html"* with a code editor or we can look at it in the panel *"Elements"* of the window *"Developer Tools"*. In both the case at the beginning of the code we can find this lines

```
<link rel="stylesheet" type="text/css" href="/css/myButton.css">
<script type="text/javascript" src="/js/myButton.js"></script>
```

So we can see that our browser is expecting to find the .css file in the folder called *"css"* and the javascript file in the folder *"js"*. Once that we know that we can modify our python script in order to do so.

[ ]:
```
import cherrypy
import os

class Example(object):
        """docstring for Example"""
        exposed=True
        def __init__(self):
                self.id=1
        def GET(self):
                return open("index.html")


if __name__ == '__main__':
        conf={
                '/':{
                                'request.dispatch':cherrypy.dispatch.
 ↪MethodDispatcher(),
                                'tools.staticdir.root': os.path.abspath(os.
 ↪getcwd()),
                        },
        ##LINES ADDED TO PROVIDE THE PATH FOR CSS
                 '/css':{
```

```
                'tools.staticdir.on': True,
                'tools.staticdir.dir':'./css'
                },
        ##LINES ADDED TO PROVIDE THE PATH FOR JS
                '/js':{
                'tools.staticdir.on': True,
                'tools.staticdir.dir':'./js'
                },
        }
        cherrypy.tree.mount(Example(),'/',conf)
        cherrypy.engine.start()
        cherrypy.engine.block()
```

As you can see we added some lines to specify the path for the folder *"css"* and *"js"*. If you did this step correctly you should be able to see on your browser the correct results

[ ]: