

Eel: Get (simple) GUI for your Python _____ with no hassle

Yasuyuki “Yatch” Tanaka, EVA

Eel: Get (simple) GUI for your Python **script** with no hassle

Yasuyuki “Yatch” Tanaka, EVA

Python script...
CUI looks cool but it is not visually
appealing...



Would it be nice to have a chart showing the internal states of your Python script for demo or development purpose??



Of course, yes!

But,
it seems difficult to add
GUI to my script....



Don't worry; install “Eel” and use it!
It's super easy to add GUI to your script.



Let's see how to use Eel with SAMPLE CODE



The sample code is available at:

<https://gitlab.inria.fr/ytanaka/devmeetup-eel/>

“Hello DevMeetup!”

Python script

(hello devmeetup.py)

```
1 import time
2
3 to_whom = 'DevMeetup'
4
5
6 def keep_saying_hello():
7     global to_whom
8
9     while True:
10         current_time = time.strftime(
11             "%H:%M:%S, %d %b %Y",
12             time.localtime()
13         )
14         msg = 'Hello, {0}! ({1})'.format(to_whom, current_time)
15         print(msg)
16         time.sleep(1)
17
18
19 if __name__ == '__main__':
20     keep_saying_hello()
```


It keeps saying "Hello, DevMeetup"...

```
$ python hello_devmeetup.py  
Hello, DevMeetup! (11:07:39, 22 Jan 2019)  
Hello, DevMeetup! (11:07:40, 22 Jan 2019)  
Hello, DevMeetup! (11:07:41, 22 Jan 2019)  
Hello, DevMeetup! (11:07:42, 22 Jan 2019)  
Hello, DevMeetup! (11:07:43, 22 Jan 2019)  
Hello, DevMeetup! (11:07:44, 22 Jan 2019)  
Hello, DevMeetup! (11:07:45, 22 Jan 2019)  
Hello, DevMeetup! (11:07:46, 22 Jan 2019)  
Hello, DevMeetup! (11:07:47, 22 Jan 2019)  
Hello, DevMeetup! (11:07:48, 22 Jan 2019)  
Hello, DevMeetup! (11:07:49, 22 Jan 2019)  
Hello, DevMeetup! (11:07:50, 22 Jan 2019)  
Hello, DevMeetup! (11:07:51, 22 Jan 2019)  
Hello, DevMeetup! (11:07:52, 22 Jan 2019)  
Hello, DevMeetup! (11:07:53, 22 Jan 2019)  
Hello, DevMeetup! (11:07:54, 22 Jan 2019)  
Hello, DevMeetup! (11:07:55, 22 Jan 2019)  
Hello, DevMeetup! (11:07:56, 22 Jan 2019)  
Hello, DevMeetup! (11:07:57, 22 Jan 2019)  
Hello, DevMeetup! (11:07:58, 22 Jan 2019)  
Hello, DevMeetup! (11:07:59, 22 Jan 2019)  
█
```

OK, put a GUI to this simple script!

- I want to change “DevMeetup” part from the GUI
- I want to see the latest message on the GUI

```
$ python hello_devmeetup.py
Hello, DevMeetup! (11:07:39, 22 Jan 2019)
Hello, DevMeetup! (11:07:40, 22 Jan 2019)
Hello, DevMeetup! (11:07:41, 22 Jan 2019)
Hello, DevMeetup! (11:07:42, 22 Jan 2019)
Hello, DevMeetup! (11:07:43, 22 Jan 2019)
Hello, DevMeetup! (11:07:44, 22 Jan 2019)
Hello, DevMeetup! (11:07:45, 22 Jan 2019)
Hello, DevMeetup! (11:07:46, 22 Jan 2019)
Hello, DevMeetup! (11:07:47, 22 Jan 2019)
Hello, DevMeetup! (11:07:48, 22 Jan 2019)
Hello, DevMeetup! (11:07:49, 22 Jan 2019)
Hello, DevMeetup! (11:07:50, 22 Jan 2019)
Hello, DevMeetup! (11:07:51, 22 Jan 2019)
Hello, DevMeetup! (11:07:52, 22 Jan 2019)
Hello, DevMeetup! (11:07:53, 22 Jan 2019)
Hello, DevMeetup! (11:07:54, 22 Jan 2019)
Hello, DevMeetup! (11:07:55, 22 Jan 2019)
Hello, DevMeetup! (11:07:56, 22 Jan 2019)
Hello, DevMeetup! (11:07:57, 22 Jan 2019)
Hello, DevMeetup! (11:07:58, 22 Jan 2019)
Hello, DevMeetup! (11:07:59, 22 Jan 2019)
```



Add these lines and prepare an HTML file accordingly....

```
3 import eel
4
5
6 to_whom = 'DevMeetup'
7
8
9 @eel.expose
10 def set_to_whom(new_value):
11     global to_whom
12     to_whom = new_value
13
```

This function can be called from Javascript as `eel.set_to_whom()`.

```
14
15 def keep_saying_hello():
16     global to_whom
17
18     eel.init('public')
19     eel.start('main.html', block=False)
20
```

Specify the directory having the HTML file and the HTML file name

```
21 while True:
22     current_time = time.strftime(
23         "%H:%M:%S, %d %b %Y",
24         time.localtime()
25     )
26     msg = 'Hello, {0}! ({1})'.format(to_whom, current_time)
27     print(msg)
28     eel.printGreeting(msg)
29     eel.sleep(1)
30
```

`eel.printGreeting()` is a function which will be provided by the Javascript code.
`time.sleep()` needs to be replaced with `eel.sleep()`

Tada

```
Hello, DevMeetup! (12:40:13,  
Hello, DevMeetup! (12:40:14,  
Hello, DevMeetup! (12:40:15,  
Hello, DevMeetup! (12:40:16,  
Hello, DevMeetup! (12:40:17,  
Hello, DevMeetup! (12:40:18,  
Hello, DevMeetup! (12:40:19,  
Hello, DevMeetup! (12:40:20,  
Hello, DevMeetup! (12:40:21,  
Hello, DevMeetup! (12:40:22,  
Hello, DevMeetup! (12:40:23,  
Hello, DevMeetup! (12:40:24,  
Hello, DevMeetup! (12:40:25,  
Hello, DevMeetup! (12:40:26,  
Hello, DevMeetup! (12:40:27,
```

Hello, DevMeetup!
(12:40:27, 22 Jan 2019)

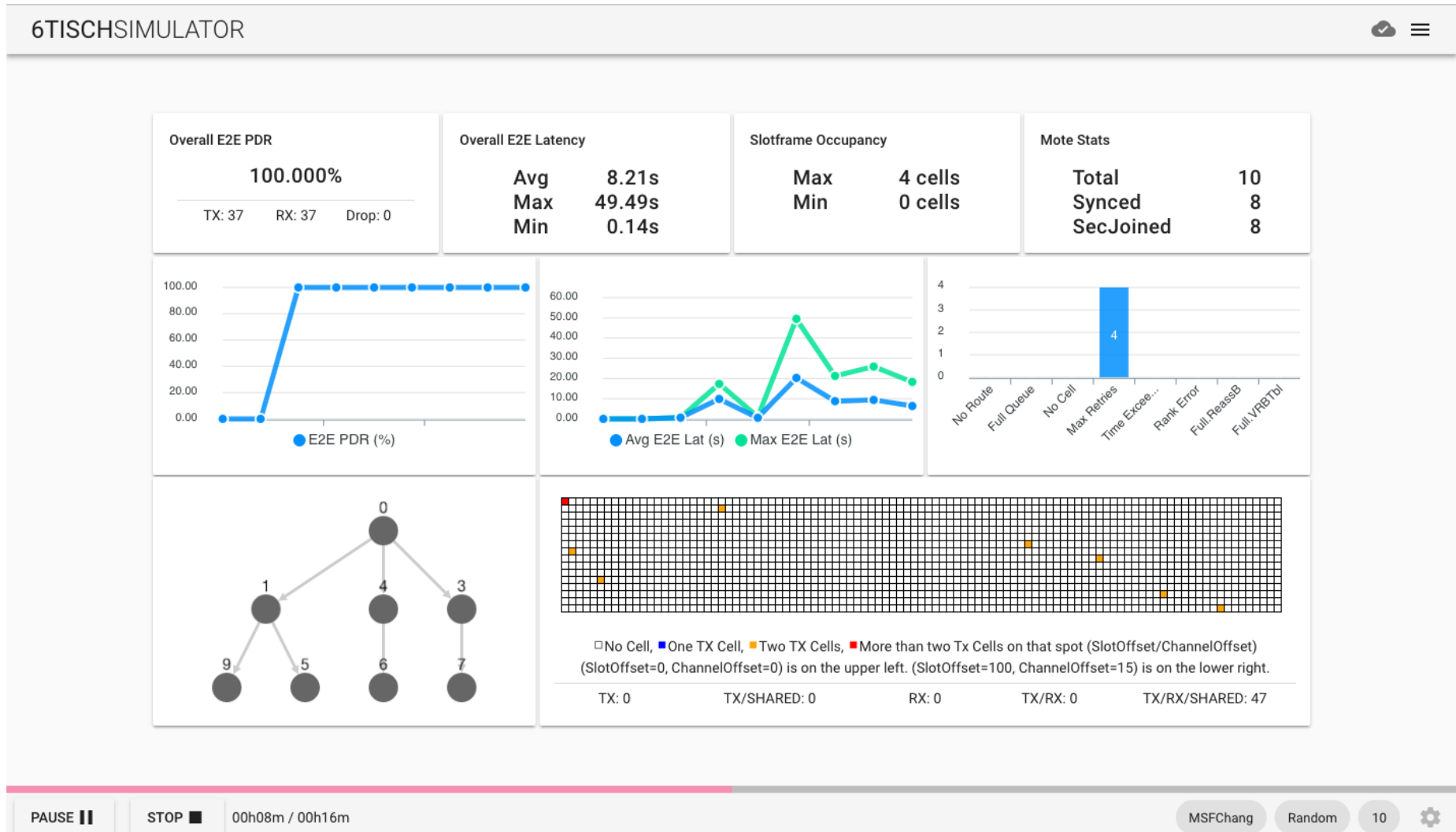
What is so great about Eel?

- Eel does all bothersome tasks to integrate your Python script with the Javascript code
- All you need to do is
 - install Eel
 - add some lines of code
 - to expose functions in the Python code
 - to call functions in the Javascript code
 - to start Eel main routine, which runs as the backend server
 - implement a GUI with HTML/CSS/Javascript
- You can use millions of Javascript libraries for your cool GUI!

For instance,
thanks to Eel, our Python script has...

```
$ python runSim.py
parameters 1/1, run 1/1
  slotframe_iteration: 0/999
  slotframe_iteration: 1/999
  slotframe_iteration: 2/999
  slotframe_iteration: 3/999
  slotframe_iteration: 4/999
  slotframe_iteration: 5/999
  slotframe_iteration: 6/999
  slotframe_iteration: 7/999
  slotframe_iteration: 8/999
  slotframe_iteration: 9/999
  slotframe_iteration: 10/999
  slotframe_iteration: 11/999
  slotframe_iteration: 12/999
  slotframe_iteration: 13/999
  slotframe_iteration: 14/999
  slotframe_iteration: 15/999
  slotframe_iteration: 16/999
```

... this beautiful GUI ☺



Eel: Get (simple) GUI for your Python **script** with no hassle

Yasuyuki “Yatch” Tanaka, EVA

backup:
step-by-step
explanation

The first feature to implement:

I want to change
“DevMeetup” part from my
web browser!

Let's have set_to_whom()

....

```
1 import time
2
3 import eel
4
5
6 to_whom = 'DevMeetup'
7
8
9 @eel.expose
10 def set_to_whom(new_value):
11     global to_whom
12     to_whom = new_value
13
14
15 def keep_saying_hello():
16     global to_whom
17
18     while True:
19         current_time = time.strftime('%H:%M:%S', time.localtime())
```

Magic happens here!

@eel.expose decorator makes the function able to be called from Javascript as eel.set_to_whom() function

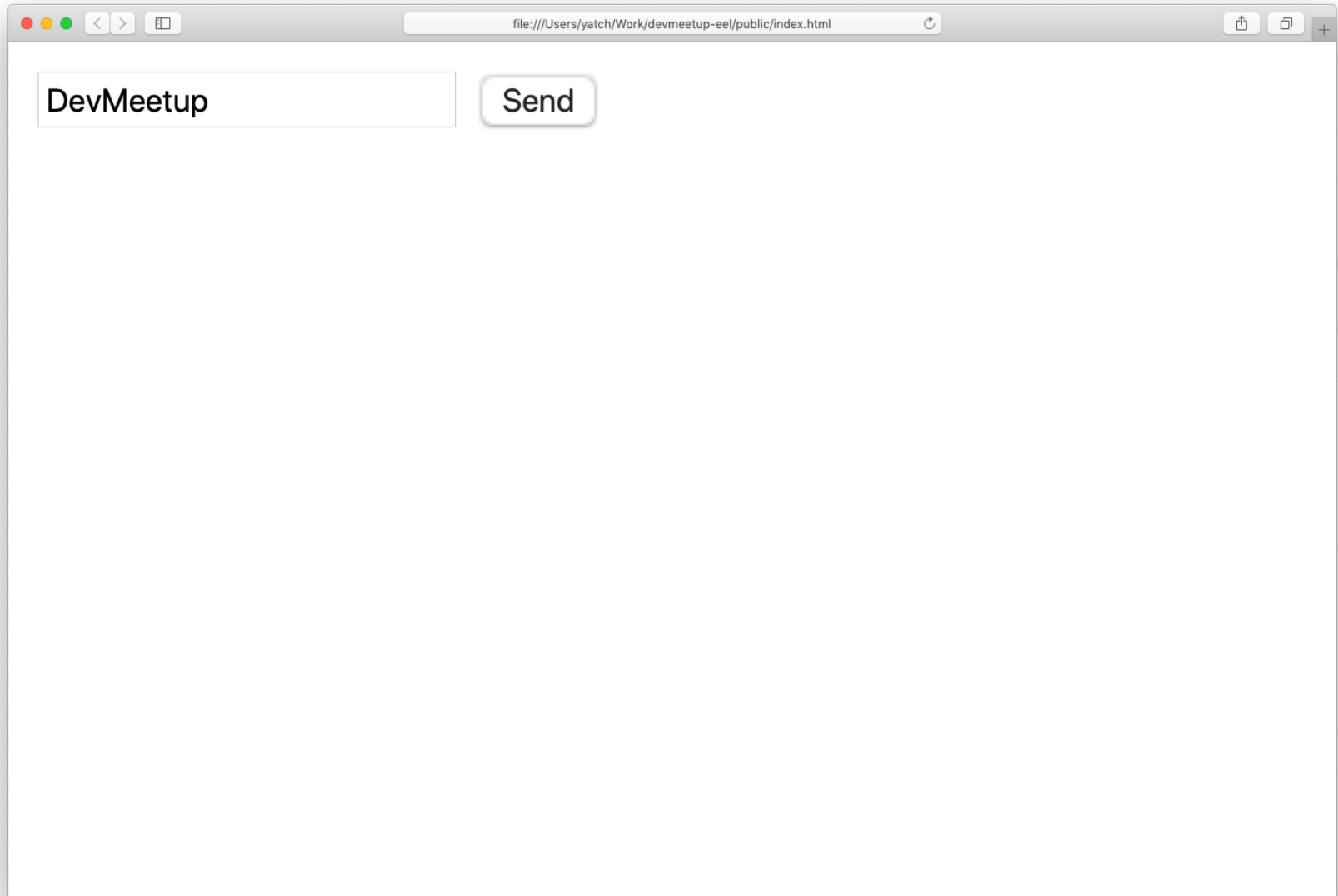
Prepare an HTML file...

(public/main.html)

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Hello DevMeetup!</title>
5     <script type='text/javascript' src='/eel.js'></script>
6     <script type='text/javascript'>
7
8       function setToWhom() {
9         const newToWhom = document.getElementById('input-to-whom').value
10        eel.set_to_whom(newToWhom)
11      }
12
13    </script>
14  </head>
15
16  <body>
17    <div>
18      <input
19        id='input-to-whom'
20        placeholder='Type "to_whome" value here'
21        value='DevMeetup'
22      >
23      <button onclick='setToWhom()'>Send</button>
24    </div>
25  </body>
26 </html>
```

On clicking the button,
eel.set_to_whom() is called, which ends
up calling the function we made in the
previous page.

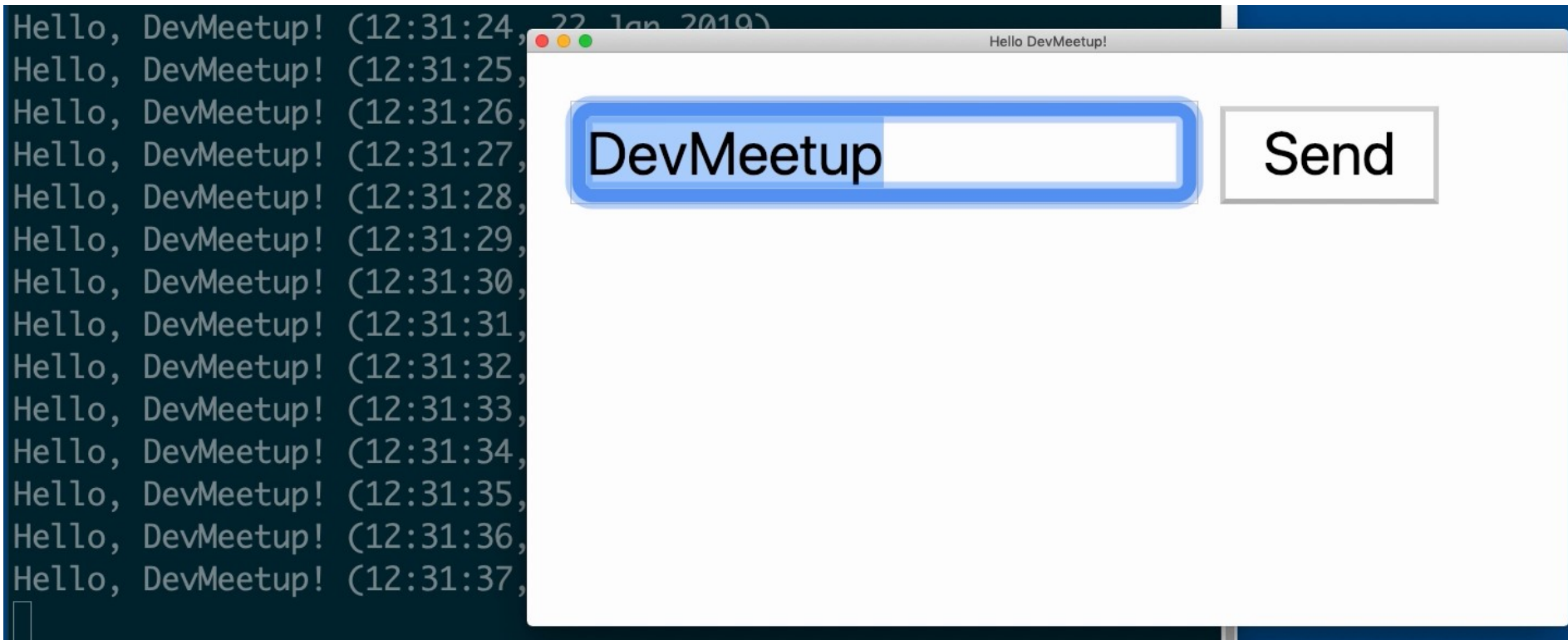
It's rendered like this....



Apply Eel to the script body...

```
14
15 def keep_saying_hello():
16     global to_whom
17     specify the directory having the HTML
        file and the HTML file name
18     eel.init('public')
19     eel.start('main.html', block=False)
20
21     while True:
22         current_time = time.strftime(
23             "%H:%M:%S, %d %b %Y",
24             time.localtime()
25         )
26         msg = 'Hello, {0}! ({1})'.format(to_whom, current_time)
27         print(msg)
28         eel.sleep(1)
29         time.sleep() needs to be replaced with
            eel.sleep()
30
31 if __name__ == '__main__':
32     keep_saying_hello()
```

Now I can change “DevMeetup” part from my browser 🎉



The second feature to implement:

I want to see the message on my browser as well....

Add printGreeting() to Javascript part... and use it in the Python script. That's it!

public/main.html

```
12
13     eel.expose(printGreeting)
14     function printGreeting(message) {
15         const greeting_div = document.getElementById('greeting')
16         greeting_div.innerHTML = message
17     }
18
19     </script>
20 </head>
21
22     <body>
23         <div id='greeting'>---</div>
24         <div>
25             <input
```

Add printGreeting() function to receive
message from the Python script as
eel.printGreeting()

hello_devmeetup.py

```
26     msg = 'Hello, {0}! ({1})'.format(to_whom, current_time)
27     print(msg)
28     eel.printGreeting(msg)
29     eel.sleep(1)
30
```

Call that function from the Python script

Now I can see the (latest) message 🎉

```
Hello, DevMeetup! (12:40:13,  
Hello, DevMeetup! (12:40:14,  
Hello, DevMeetup! (12:40:15,  
Hello, DevMeetup! (12:40:16,  
Hello, DevMeetup! (12:40:17,  
Hello, DevMeetup! (12:40:18,  
Hello, DevMeetup! (12:40:19,  
Hello, DevMeetup! (12:40:20,  
Hello, DevMeetup! (12:40:21,  
Hello, DevMeetup! (12:40:22,  
Hello, DevMeetup! (12:40:23,  
Hello, DevMeetup! (12:40:24,  
Hello, DevMeetup! (12:40:25,  
Hello, DevMeetup! (12:40:26,  
Hello, DevMeetup! (12:40:27,
```

Hello, DevMeetup!
(12:40:27, 22 Jan 2019)