

## Lab 6

- Study the GitHub repository Lesson 6
- Install Node.js and run hello-world.js, hello.js, and http.js

```
Hello, World!
```

```
--use-openssl-ca      failure, reporting it to stderr) or
--v8-options          'silent' (map and silently ignore failure)
--v8-pool-size=...    use OpenSSL's default CA store (default)
--v, --version        print V8 command line options
--zero-fill-buffers   set V8's thread pool size
                     print Node.js version
                     automatically zero-fill all newly allocated
                     buffer and SlowBuffer instances

Environment variables:
NODE_DEBUG            ','-separated list of core modules that
                     should print debug information
NODE_DEBUG_NATIVE     ','-separated list of C++ core debug
                     categories that should print debug output
NODE_DISABLE_COLORS   set to 1 to disable colors in the REPL
NODE_EXTRA_CA_CERTS   path to additional CA certificates file
NODE_NO_WARNINGS      set to 1 to silence process warnings
NODE_OPTIONS          set CLI options in the environment via a
                     space-separated list
NODE_PATH             ':'-separated list of directories prefixed
                     to the module search path
NODE_PENDING_DEPRECATION set to 1 to emit pending deprecation
                     warnings
NODE_PENDING_PIPE_INSTANCES set the number of pending pipe instance
                     handles on Windows
NODE_PRESERVE_SYMLINKS set to 1 to preserve symbolic links when
                     resolving and caching modules
NODE_REDIRECT_WARNINGS write warnings to path instead of stderr
NODE_REPL_HISTORY      path to the persistent REPL history file
NODE_TLS_REJECT_UNAUTHORIZED set to 0 to disable TLS certificate
                     validation
NODE_V8_COVERAGE       directory to output v8 coverage JSON to
OPENSSL_CONF           load OpenSSL configuration from file
SSL_CERT_DIR           sets OpenSSL's directory of trusted
                     certificates when used in conjunction with
                     --use-openssl-ca
SSL_CERT_FILE          sets OpenSSL's trusted certificate file
                     when used in conjunction with
                     --use-openssl-ca
UV_THREADPOOL_SIZE     sets the number of threads used in libuv's
                     threadpool

Documentation can be found at https://nodejs.org/
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak$ cd iot
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot$ cd lesson6
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ node hello-world.js
Server running at http://127.0.0.1:3000/
```

```
Hello World!
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/WindowsPowerShell

PS C:\Users\nehak> wsl.exe
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak$ node -v
v12.22.9
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak$ npm -v
9.5.0
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak$ cd lesson 6
-bash: cd: too many arguments
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak$ cd lesson6
-bash: cd: lesson6: No such file or directory
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak$ cd iot
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot$ cd lesson6
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ node hello-world.js
Server running at http://127.0.0.1:3000/
^C
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ node hello.js
Server running at http://127.0.0.1:8080/
response end call done
request end event fired
^C
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ node hello.js
Server running at http://127.0.0.1:8080/
response end call done
request end event fired
```

This page was refreshed 7 times!

```
-bash: cd: too many arguments
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak$ cd lesson6
-bash: cd: lesson6: No such file or directory
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak$ cd iot
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot$ cd lesson6
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ node hello-world.js
Server running at http://127.0.0.1:3000/
^C
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ node hello.js
Server running at http://127.0.0.1:8080/
response end call done
request end event fired
^C
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ node hello.js
Server running at http://127.0.0.1:8080/
response end call done
request end event fired
^C
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ node http.js
0
1
2
3
4
5
6
```

- Refresh the webpage to see server activities
- *Install Pystache and run say\_hello.py that uses the template in say\_hello.mustache*

```
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ cat say_hello.mustache
Hello, {{to}}!
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ cat say_hello.py
# https://github.com/defunkt/pystache
import pystache
print(pystache.render('Hi {{person}}!', {'person': 'Alexa'}))

# Create dedicated view classes to hold view logic
class SayHello(object):
    def to(self):
        return "World"
hello = SayHello()

# Use template in say_hello.mustache
renderer = pystache.Renderer()
print(renderer.render(hello))

# Pre-parse a template
parsed = pystache.parse('Hey {{#who}}{{.}}!{{/who}}')
print(parsed)
print(renderer.render(parsed, {'who': 'Google'}))
print(renderer.render(parsed, {'who': 'Siri'}))
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$ python3 say_hello.py
Hi Alexa!
Hello, World!

['Hey ', _SectionNode(key='who', index_begin=12, index_end=18, parsed=[_EscapeNode(key='.'), '!'])]
Hey Google!
Hey Siri!
nkundra@DESKTOP-4TNRFB2:/mnt/c/Users/nehak/iot/lesson6$
```