

Emscripten Development Environment Setup for Kali Linux

Install Prerequisites:

1:

EMCC uses a lot of python scripts in the background. Consequently, you need to install python3 if it's not installed already.

1. `sudo apt update`
2. `sudo apt-get install python3`

Install other packages that may be needed

1. `sudo apt-get install node, wget, cmake`

To install VScode IDE

1. `wget -qO- https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor > packages.microsoft.gpg`
2. `$ sudo install -o root -g root -m 644 packages.microsoft.gpg /etc/apt/trusted.gpg.d/`
3. `$ sudo sh -c 'echo "deb [arch=amd64 signed-by=/etc/apt/trusted.gpg.d/packages.microsoft.gpg] https://packages.microsoft.com/repos/vscode stable main" > /etc/apt/sources.list.d/vscode.list'`
4. `sudo apt install code`

EMSCRIPTEN Installation

To install emscripten sdk

1. `git clone https://github.com/emscripten-core/emsdk.git`
2. `cd emsdk`
3. `ls emsdk`

```
(kali@kali) ~[~/emsdk]
$ ls
bazel          emscripten-releases-tags.json  emsdk_env.bat  emsdk_env.ps1  emsdk.ps1  legacy-emscripten-tags.txt  node  test
docker         emsdk                        emsdk_env.csh  emsdk_env.sh   emsdk.py   LICENSE                    README.md  upstream
emcmdprompt.bat emsdk.bat                  emsdk_env.fish emsdk_manifest.json  legacy-binaryen-tags.txt  llvm-tags-64bit.txt  scripts  zips
```

1. `./emsdk install latest`
2. `./emsdk activate latest`
3. `source ./emsdk_env.sh`

```
Setting up EMSDK environment (suppress these messages with EMSDK_QUIET=1)
Adding directories to PATH:
PATH += /home/kali/emsdk
PATH += /home/kali/emsdk/upstream/emscripten
PATH += /home/kali/emsdk/node/14.18.2_64bit/bin

Setting environment variables:
PATH = /home/kali/emsdk:/home/kali/emsdk/upstream/emscripten:/home/kali/emsdk/node/14.18.2_64bit/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games
EMSDK = /home/kali/emsdk
EM_CONFIG = /home/kali/emsdk/.emscripten
EMSDK_NODE = /home/kali/emsdk/node/14.18.2_64bit/bin/node
```

NOTE:

Please not that you need to run `source <path to emsdk_env.sh>` if you change directory and you need to add `emcc` command to execution path. To simplify the solution, I add the source command `<source /home/kali/emsdk/emsdk_env.sh>` to my `zshrc` file.

Verifying Hello World Project

```
c hello_world.c > main()
1  #include <stdio.h>
2
3  int main() {
4      printf("hello, world!\n");
5      return 0;
6  }
```

- Run `emcc hello_world.c`

```
(kali㉿kali)-[~/hello_world]
$ emcc hello_world.c

(kali㉿kali)-[~/hello_world]
$ ls
a.out.js  a.out.wasm  hello_world.c

(kali㉿kali)-[~/hello_world]
$
```

2:

Code Base to Transpile:

https://github.com/epoell/openMP_examples_and_Matrix-Matrix-Multiplication

Code was refactored to suit test case.

matrix.h

```
1. class Matrix {
2.     public:
3.         Matrix();
4.         double multMatrix(int n);
5. };
6.
```

matrix.cpp

```
1. #include "matrix.h"
2. #include <iostream>
3. #include <sys/time.h>
4.
5. using namespace std;
6.
7. Matrix::Matrix() {
8.
9. }
10.
11.
12. double Matrix::multMatrix(int n) {
13.
14.     double A[n][n], B[n][n], C[n][n];
15.
16.     // Initialize Matrices
17.
18.     for(int i = 0; i < n; ++i)
19.         for(int j = 0; j < n; ++j)
20.             {
21.                 A[i][j] = (double)rand()/ (double)RAND_MAX;
22.                 B[i][j] = (double)rand()/ (double)RAND_MAX;
23.                 C[i][j] = 0;
24.             }
25.
26.     // Matrix multiplication
27.
28.     int i,j,k;
29.
30.     struct timeval start, end;
```

```

31.
32.  // start timer.
33.  gettimeofday(&start, NULL);
34.
35.  // unsync the I/O of C and C++.
36.  ios_base::sync_with_stdio(false);
37.
38.
39.  for(i = 0; i < n; ++i) {
40.      for(int k = 0; k < n; ++k) {
41.          for(j = 0; j < n; ++j) {
42.              C[i][j] += A[i][k] * B[k][j];
43.          }
44.      }
45.  }
46.
47.  gettimeofday(&end, NULL);
48.
49.  double time_taken;
50.
51.  time_taken = (end.tv_sec - start.tv_sec) * 1e6;
52.  time_taken = (time_taken + (end.tv_usec -
53.                          start.tv_usec)) * 1e-6;
54.
55.  return time_taken;
56.}
57.

```

The code takes in an integer value n , creates 3 matrix $n * n$ matrix, populates matrix with random floating-point values and then multiplies the 3 matrices together. The code returns time taken for the computation in secs.

Testing Base Code by compiling with emcc:

- after cloning class GitHub repo, create a **build** directory in the repo **lab-1-emscripten-eltopus/Code** to hold all cmake configurations.
- create a CMakeLists.txt in **lab-1-emscripten-eltopus/Code**

```

cmake_minimum_required(VERSION 3.10)

# set the project name
project(Lab1)

# add the executable
add_executable(matrix matrix.cpp)

```

- run `cmake --build ./build` from **lab-1-emscripten-eltopus/Code**
- This created the following files in the build folder

```
(kali㉿kali)-[~/lab-1-emsripten-eltopus/Code/build]
$ ls
CMakeCache.txt  CMakeFiles  cmake_install.cmake  Makefile
```

- modify Makefile to compile .cpp to web assembly
- run make compile

```
(kali㉿kali)-[~/lab-1-emsripten-eltopus/Code/build]
$ make compile
emcc /home/kali/lab-1-emsripten-eltopus/Code/matrix.cpp -o /home/kali/lab-1-emsripten-eltopus/Code/output
```

- Successful compilation should create 2 files in /home/kali/lab-1-emsripten-eltopus/Code/

```
$ ls -al
total 164
drwxr-xr-x 4 kali kali 4096 Sep 13 04:54 .
drwxr-xr-x 7 kali kali 4096 Sep 9 09:55 ..
-rw-r--r-- 1 kali kali 195 Sep 12 00:10 app.js
drwxr-xr-x 3 kali kali 4096 Sep 10 18:25 build
-rw-r--r-- 1 kali kali 130 Sep 9 09:50 CMakeLists.txt
-rw-r--r-- 1 kali kali 649 Sep 12 00:11 index.html
-rw-r--r-- 1 kali kali 911 Sep 3 22:25 index.js
-rw-r--r-- 1 kali kali 71 Sep 12 00:03 matrix_classes.idl
-rw-r--r-- 1 kali kali 1088 Sep 12 00:02 matrix.cpp
-rw-r--r-- 1 kali kali 63 Sep 11 15:10 matrix_glue_wrapper.cpp
-rw-r--r-- 1 kali kali 81 Sep 12 00:03 matrix.h
-rw-r--r-- 1 kali kali 552 Sep 3 14:26 matrix.py
drwxr-xr-x 61 kali kali 4096 Sep 3 21:52 node_modules
-rw-r--r-- 1 kali kali 82367 Sep 13 04:54 output
-rwxr-xr-x 1 kali kali 1679 Sep 13 04:54 output.wasm
-rw-r--r-- 1 kali kali 315 Sep 3 21:52 package.json
-rw-r--r-- 1 kali kali 18491 Sep 3 21:52 package-lock.json
```

3:

Build Code base to web assembly

- create `matrix_glue_wrapper.cpp`

```
#include <stddef.h>
#include "matrix.h"
#include "glue.cpp"
```

- make the following modifications to ***lab-1-emsripten-eltopus/Code/build/Makefile***
- `glue.cpp` does not currently exist but will be generated later

```
SHELL = /bin/sh
# The top-level source directory on which CMake was run.
CMAKE_SOURCE_DIR = /home/kali/lab-1-emsripten-eltopus/Code

# The top-level build directory on which CMake was run.
CMAKE_BINARY_DIR = /home/kali/lab-1-emsripten-eltopus/Code/build

MATRIX_CPP = /home/kali/lab-1-emsripten-eltopus/Code/matrix.cpp
WASM_OUTPUT = /home/kali/lab-1-emsripten-eltopus/Code/output
WASM_WASM = /home/kali/lab-1-emsripten-eltopus/Code/output.wasm
MATRIX_GLUE_WRAPPER = /home/kali/lab-1-emsripten-
eltopus/Code/matrix_glue_wrapper.cpp
MATRIX_GLUE = /home/kali/lab-1-emsripten-eltopus/Code/glue.cpp
NODE = /home/kali/lab-1-emsripten-eltopus/Code/app.js
OUTPUT_JS = /home/kali/lab-1-emsripten-eltopus/Code/output.js
GLUE_JS = /home/kali/lab-1-emsripten-eltopus/Code/glue.js
EMSDK_ROOT_FOLDER = /home/kali/emsdk/upstream/emsripten
REPO_CODE_DIR = /home/kali/lab-1-emsripten-eltopus/Code

compile:
    $(SHELL source ./home/kali/emsdk/emsdk_env.sh)
    $(EMSDK_ROOT_FOLDER)/emcc $(MATRIX_CPP) -o $(WASM_OUTPUT)

glue:
    python3 $(EMSDK_ROOT_FOLDER)/upstream/emsripten/tools/webidl_binder.py
$(REPO_CODE_DIR)/matrix_classes.idl $(REPO_CODE_DIR)/glue

build:
    $(EMSDK_ROOT_FOLDER)/emcc $(MATRIX_CPP) $(MATRIX_GLUE_WRAPPER) --post-js
$(GLUE_JS) -o $(OUTPUT_JS) -s EXPORTED_RUNTIME_METHODS=["ccall, cwrap"]

node:
    node $(NODE)

clean:
    rm -rf $(MATRIX_GLUE) $(GLUE_JS) $(WASM_OUTPUT) $(WASM_WASM) $(OUTPUT_JS)
```

- create matrix_classes.idl

```
interface Matrix {
    void Matrix();
    double multMatrix(long n);
};
```

- run make glue

```
(kali㉿kali)-[~/lab-1-emsripten-eltopus/Code/build]
$ make glue
python3 /home/kali/emscripten-upstream/emscripten/tools/webidl_binder.py /home/kali/lab-1-emsripten-eltopus/Code/matrix_classes.idl /home/kali/lab-1-emsripten-eltopus/Code/glue
```

- glue.cpp and glue.js files will be generated

```
(kali㉿kali)-[~/lab-1-emsripten-eltopus/Code]
$ ls -al
total 176
drwxr-xr-x  4 kali kali  4096 Sep 13 05:10 .
drwxr-xr-x  7 kali kali  4096 Sep  9 09:55 ..
-rw-r--r--  1 kali kali   195 Sep 12 00:10 app.js
drwxr-xr-x  3 kali kali  4096 Sep 10 18:25 build
-rw-r--r--  1 kali kali   130 Sep  9 09:50 CMakeLists.txt
-rw-r--r--  1 kali kali   882 Sep 13 05:10 glue.cpp
-rw-r--r--  1 kali kali  7491 Sep 13 05:10 glue.js
-rw-r--r--  1 kali kali   649 Sep 12 00:11 index.html
-rw-r--r--  1 kali kali   911 Sep  3 22:25 index.js
-rw-r--r--  1 kali kali    71 Sep 12 00:03 matrix_classes.idl
-rw-r--r--  1 kali kali  1088 Sep 12 00:02 matrix.cpp
-rw-r--r--  1 kali kali    63 Sep 11 15:10 matrix_glue_wrapper.cpp
-rw-r--r--  1 kali kali    81 Sep 12 00:03 matrix.h
-rw-r--r--  1 kali kali   552 Sep  3 14:26 matrix.py
drwxr-xr-x 61 kali kali  4096 Sep  3 21:52 node_modules
-rw-r--r--  1 kali kali 82367 Sep 13 04:54 output
-rwxr-xr-x  1 kali kali  1679 Sep 13 04:54 output.wasm
-rw-r--r--  1 kali kali   315 Sep  3 21:52 package.json
-rw-r--r--  1 kali kali 18491 Sep  3 21:52 package-lock.json
```

- run glue build

```
(kali㉿kali)-[~/lab-1-emsripten-eltopus/Code/build]
$ make build
emcc /home/kali/lab-1-emsripten-eltopus/Code/matrix.cpp /home/kali/lab-1-emsripten-eltopus/Code/matrix_glue_wrapper.cpp --post-js /home/kali/lab-1-emsripten-eltopus/Code/glue.js -o /home/kali/lab-1-emsripten-eltopus/Code/output.js -s EXPORTED_RUNTIME_METHODS=["ccall, cwrap"]
```

- output.js will be generated.

```
(kali@kali)-[~/lab-1-emsripten-eltopus/Code]
$ ls -al
total 276
drwxr-xr-x  4 kali kali  4096 Sep 13 05:20 .
drwxr-xr-x  7 kali kali  4096 Sep  9 09:55 ..
-rw-r--r--  1 kali kali   195 Sep 12 00:10 app.js
drwxr-xr-x  3 kali kali  4096 Sep 10 18:25 build
-rw-r--r--  1 kali kali   130 Sep  9 09:50 CMakeLists.txt
-rw-r--r--  1 kali kali   882 Sep 13 05:10 glue.cpp
-rw-r--r--  1 kali kali  7491 Sep 13 05:10 glue.js
-rw-r--r--  1 kali kali   649 Sep 12 00:11 index.html
-rw-r--r--  1 kali kali   911 Sep  3 22:25 index.js
-rw-r--r--  1 kali kali    71 Sep 12 00:03 matrix_classes.idl
-rw-r--r--  1 kali kali  1088 Sep 12 00:02 matrix.cpp
-rw-r--r--  1 kali kali    63 Sep 11 15:10 matrix_glue_wrapper.cpp
-rw-r--r--  1 kali kali    81 Sep 12 00:03 matrix.h
-rw-r--r--  1 kali kali   552 Sep  3 14:26 matrix.py
drwxr-xr-x 61 kali kali  4096 Sep  3 21:52 node_modules
-rw-r--r--  1 kali kali 82367 Sep 13 04:54 output
-rw-r--r--  1 kali kali 91778 Sep 13 05:20 output.js
-rwxr-xr-x  1 kali kali 11995 Sep 13 05:20 output.wasm
-rw-r--r--  1 kali kali   315 Sep  3 21:52 package.json
-rw-r--r--  1 kali kali 18491 Sep  3 21:52 package-lock.json

(kali@kali)-[~/lab-1-emsripten-eltopus/Code]
```

Use Html Output page using NodeJS server

- run npm init
- follow cli instructions
- run npm install express cors body-parser
- create index.html file

```
1. <!DOCTYPE html>
2. <html lang="en">
3.   <head>
4.     <title>Advanced Application Programming Lab 1</title>
5.     <meta charset="UTF-8">
6.     <meta name="viewport" content="width=device-width, initial-scale=1">
7.     <link rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.cs
      s">
8.   </head>
9.   <body>
10.    <div class="container">
11.      <h1>Matrix Multiplication Completion Time</h1>
12.      <form onsubmit="return calMatrix()">
13.        <div class="form-group">
14.          <label for="firstName">Matrix Size</label>
15.          <input type="text" class="form-control" id="size"
            placeholder="Enter matrix size" name="matrix_size">
16.        </div>
```



```

17.         <button type="submit" class="btn btn-primary">Submit</button>
18.     </form>
19. </body>
20. <script type="text/javascript" src="./output.js"> </script>
21. <script>
22.     function calMatrix() {
23.         const n = document.getElementById("size").value;
24.         var matrix = new Module.Matrix();
25.         const time_taken = matrix.multMatrix(n);
26.         console.log("Time taken is: " + time_taken + " sec");
27.         alert("Time taken is: " + time_taken.toFixed(4) + " secs");
28.     }
29.
30. </script>
31. </html>
32.

```

- create index.js file

```

1. const express = require('express'); //Import the express dependency
2. const cors = require('cors');
3. const bodyParser = require('body-parser')
4.
5. const app = express(); //Instantiate an express app, the main
   work horse of this server
6. const port = 5000; //Save the port number where your
   server will be listening
7.
8. var urlencodedParser = bodyParser.urlencoded({ extended: false })
9.
10. app.use(cors());
11. app.use('/', express.static('.'))
12.
13. app.get('/', (req, res) => { //get requests to the root ("/") will
   route here
14.     res.sendFile('index.html', {root: __dirname}); //server responds
   by sending the index.html file to the client's browser
15. //the .sendFile
   method needs the absolute path to the file, see:
   https://expressjs.com/en/4x/api.html#res.sendFile
16. });
17.
18. app.post('/', urlencodedParser, (req, res) => {
19.     console.log('Got body:', req.body);
20.     res.sendStatus(200);
21. });
22.
23. app.listen(port, () => { //server starts listening for any
   attempts from a client to connect at port: {port}
24.     console.log(`Now listening on port ${port}`);
25. });

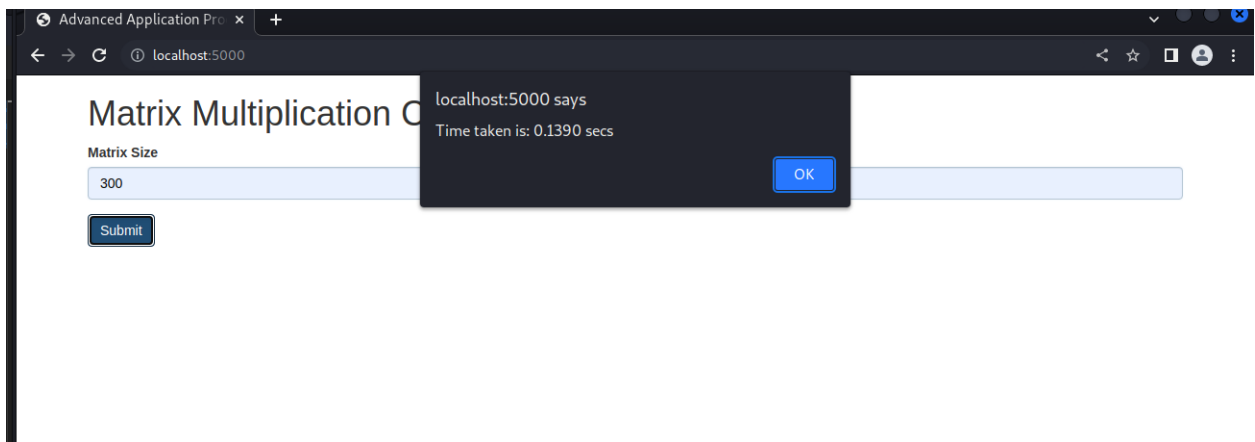
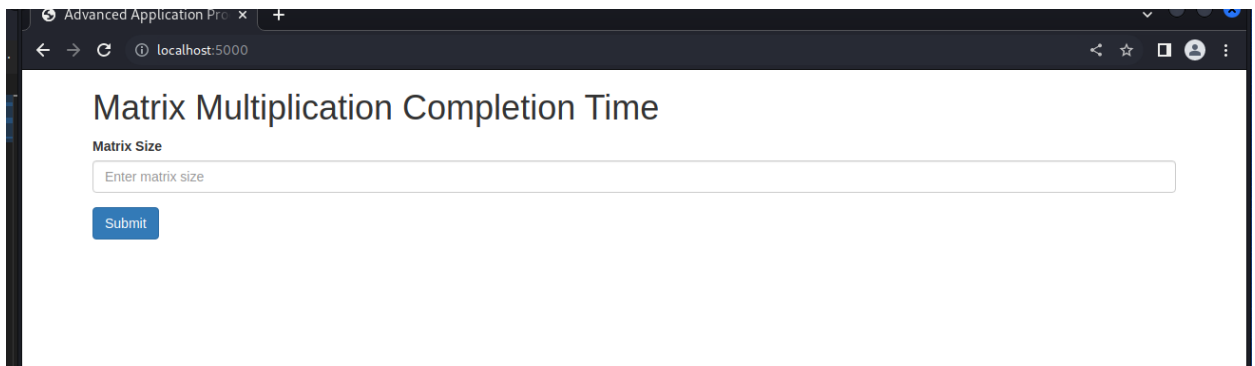
```

26.

- run make node

```
(kali@kali)-[~/lab-1-emscripten-eltopus/Code]
$ node index.js
Now listening on port 5000
```

- launch Google Chrome browser and navigate to localhost:5000



Compare Native Code Vs Javascript Code

- create app.sh

```
1. echo "statrting report generation...."
2. export DATA_DIR="/home/kali/lab-1-emsripten-eltopus/Data/report.txt"
3. rm $DATA_DIR
4. touch $DATA_DIR
5. g++ -o app.exe app.cpp
6. echo "Running C++ Native Code"
7. echo "Running C++ Native Code" >> $DATA_DIR
8. ./app.exe >> $DATA_DIR
9. echo "-----" >> $DATA_DIR
10. echo "Running Javascript Code"
11. echo "Running Javascript Code" >> $DATA_DIR
12. node app.js >> $DATA_DIR
13. echo "Report Generation Completed..."
14.
```

- run app.sh

Running C++ Native Code

Time taken for run #: 0: 0.177319 sec

Time taken for run #: 1: 0.118716 sec

Time taken for run #: 2: 0.094826 sec

Time taken for run #: 3: 0.140683 sec

Time taken for run #: 4: 0.106639 sec

Time taken for run #: 5: 0.141147 sec

Time taken for run #: 6: 0.138094 sec

Time taken for run #: 7: 0.095357 sec

Time taken for run #: 8: 0.179817 sec

Time taken for run #: 9: 0.114421 sec

Time taken for run #: 10: 0.137512 sec

Time taken for run #: 11: 0.120689 sec

Time taken for run #: 12: 0.122664 sec

Time taken for run #: 13: 0.100708 sec

Time taken for run #: 14: 0.146688 sec

Time taken for run #: 15: 0.135986 sec

Time taken for run #: 16: 0.148544 sec

Time taken for run #: 17: 0.094389 sec

Time taken for run #: 18: 0.115273 sec

Time taken for run #: 19: 0.206761 sec

Time taken for run #: 20: 0.186709 sec

Time taken for run #: 21: 0.143626 sec

Time taken for run #: 22: 0.130612 sec

Time taken for run #: 23: 0.124849 sec

Time taken for run #: 24: 0.111902 sec

Time taken for run #: 25: 0.117587 sec

Time taken for run #: 26: 0.103258 sec

Time taken for run #: 27: 0.118919 sec

Time taken for run #: 28: 0.126468 sec

Time taken for run #: 29: 0.10795 sec

Average Time taken for 30 runs is: 0.13027 sec

Running Javascript Code

Time taken for run #: 0: 0.4130 sec

Time taken for run #: 1: 0.0940 sec

Time taken for run #: 2: 0.1370 sec

Time taken for run #: 3: 0.1870 sec

Time taken for run #: 4: 0.0960 sec

Time taken for run #: 5: 0.1390 sec

Time taken for run #: 6: 0.1160 sec

Time taken for run #: 7: 0.1230 sec

Time taken for run #: 8: 0.1090 sec

Time taken for run #: 9: 0.1500 sec

Time taken for run #: 10: 0.1070 sec

Time taken for run #: 11: 0.1320 sec

Time taken for run #: 12: 0.1370 sec

Time taken for run #: 13: 0.1110 sec

Time taken for run #: 14: 0.1260 sec

Time taken for run #: 15: 0.0940 sec

Time taken for run #: 16: 0.1130 sec

Time taken for run #: 17: 0.0950 sec

Time taken for run #: 18: 0.1060 sec

Time taken for run #: 19: 0.1140 sec

Time taken for run #: 20: 0.1180 sec

Time taken for run #: 21: 0.1200 sec

Time taken for run #: 22: 0.1010 sec

Time taken for run #: 23: 0.1090 sec

Time taken for run #: 24: 0.0890 sec

Time taken for run #: 25: 0.1220 sec

Time taken for run #: 26: 0.0900 sec

Time taken for run #: 27: 0.0970 sec

Time taken for run #: 28: 0.1230 sec

Time taken for run #: 29: 0.1430 sec

Average Time taken for 30 runs is: 0.1270 sec

Average runtime for both native C++ and JavaScript for matrix multiplication of size 30 is 0.13 secs.

No difference in runtime.