
Minimal C kernel for Jupyter

Use with Docker (recommended)

- `docker pull brendanrius/jupyter-c-kernel`
- `docker run -p 8888:8888 brendanrius/jupyter-c-kernel`
- Copy the given URL containing the token, and browse to it. For instance:

```
1 Copy/paste this URL into your browser when you connect for the first
   time,
2 to login with a token:
3 http://localhost:8888/?token=66750
   c80bd0788f6ba15760aadz53beb9a9fb4cf8ac15ce8
```

Manual installation

Works only on Linux and OS X. Windows is not supported yet. If you want to use this project on Windows, please use Docker.

- Make sure you have the following requirements installed:
 - gcc
 - jupyter
 - python 3
 - pip

Step-by-step:

- `pip install jupyter-c-kernel`
- `install_c_kernel`
- `jupyter-notebook`. Enjoy!

Example of notebook

Valid code

```
In [1]: #include <stdio.h>

int main() {
    printf("Hello world\n");
}
```

Hello world

But the kernel will also display compilations warnings and errors if we have some. Let's try to remove the `#include <stdio.h>` statement, which is necessary for `printf` to work.

Code generating warnings during compilation

```
In [2]: int main() {
        printf("Hello world\n");
    }
```

```
/var/folders/3l/qct57s856n5f0b2r4tgbmnlc0000gn/T/tmp0Qan0v.c:2:5: warning: implicitly declaring library function 'printf' with type 'int (const char *, ...)' [-Wimplicit-function-declaration]
    printf("Hello world\n");
    ^
/var/folders/3l/qct57s856n5f0b2r4tgbmnlc0000gn/T/tmp0Qan0v.c:2:5: note: include the header <stdio.h> or explicitly provide a declaration for 'printf'
1 warning generated.
```

Hello world

You can see that the warnings are displayed, but the code is still executed.

Code generating errors during compilation

```
In [3]: int main() {
        printf("Hello world")
    }
```

```
/var/folders/3l/qct57s856n5f0b2r4tgbmnlc0000gn/T/tmpw4Kz3c.c:2:5: warning: implicitly declaring library function 'printf' with type 'int (const char *, ...)' [-Wimplicit-function-declaration]
    printf("Hello world")
    ^
/var/folders/3l/qct57s856n5f0b2r4tgbmnlc0000gn/T/tmpw4Kz3c.c:2:5: note: include the header <stdio.h> or explicitly provide a declaration for 'printf'
/var/folders/3l/qct57s856n5f0b2r4tgbmnlc0000gn/T/tmpw4Kz3c.c:2:26: error: expected ';' after expression
    printf("Hello world")
    ^
    ;
1 warning and 1 error generated.
[C kernel] GCC exited with code 1, the executable will not be executed
```

You can see that the errors and warnings are shown, but the code is not executed. The return code is also added to the end of `stderr`

Custom compilation flags

You can use custom compilation flags like so:

```
In [8]: #!/cflags:-lm

#include <stdio.h>
#include <math.h>

int main() {
    printf("sqrt(67) = %f", sqrt(67));
    return 0;
}

sqrt(67) = 8.185353
```

Here, the `-lm` flag is passed so you can use the math library.

Contributing

The docker image installs the kernel in editable mode, meaning that you can change the code in real-time in Docker. For that, just run the docker box like that:

```
1 git clone https://github.com/brendan-rius/jupyter-c-kernel.git
2 cd jupyter-c-kernel
3 docker run -v $(pwd):/jupyter/jupyter_c_kernel/ -p 8888:8888
  brendanrius/jupyter-c-kernel
```

This clones the source, run the kernel, and binds the current folder (the one you just cloned) to the corresponding folder in Docker. Now, if you change the source, it will be reflected in `http://localhost:8888` instantly. Do not forget to click “restart” the kernel on the page as it does not auto-restart.

License

MIT