Instalación de CLion IDE de C++ con Licencia Estudiantil

Departamento de Computación, FCEyN, Universidad de Buenos Aires.

12 de Agosto de 2019

CLion

- En el Taller de Algo-1 vamos a utilizar un entorno de desarrollo de C++ (IDE) denominado CLion.
- Esta IDE permite crear proyectos, editar archivos, compilar, *debuggear*, entre otras cosas.
- CLion es desarrollado, soportado, y distribuido por JETBRAINS (www.jetbrains.com).
- Es una herramienta que se puede descargar bajo licencia estudiantil, que permite su utilización gratuita por un año.

CLion

El CLion va a estar instalado en las PCs del laboratorio. Si alguien quiere descargarlo e instalarlo, debe seguir los siguientes pasos.

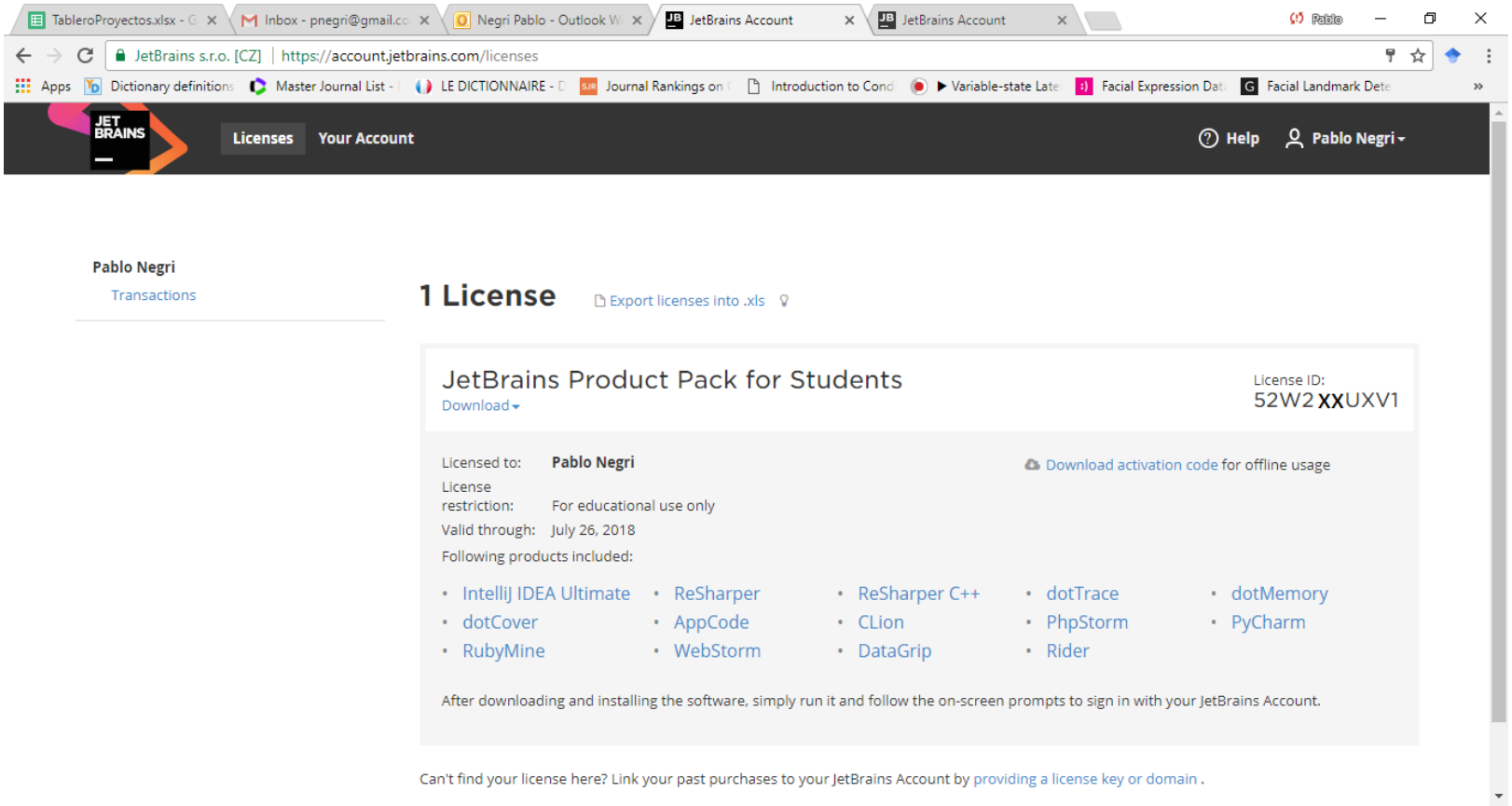
- En la pagina `https://www.jetbrains.com/student/` es posible pedir la licencia estudiantil usanto el botón Ápply Now'.
- Este botón lleva a la página `https://www.jetbrains.com/shop/eform/students`, donde hay que llenar los campos, y lo que es muy importante, poner su dirección estudiantil '@dc.uba.ar'.

- El sistema envia un correo avisando el pedido de licencia estudiantil.
- En un correo siguiente se envia un enlace para activar la licencia. Se crea un usuario automáticamente.
- Se accede a la página de Licencias, desde donde:
  - Descargar la licencia en formato xls.
  - Descargar el cógido de activación ("Download Activation Code").

CLion

- El sistema envia un correo avisando el pedido de licencia estudiantil.
- En un correo siguiente se envia un enlace para activar la licencia. Se crea un usuario automáticamente.
- Se accede a la página de Licencias, desde donde:
  - Descargar la licencia en formato xls.
  - Descargar el cógido de activación ("Download Activation Code").
- Desde la página es posible descargar cualquiera de las aplicaciones de JetBrains.
- Elegir CLion (pueden bajar otras, por supuesto).

CLion


TableroProyectos.xlsx - G
Inbox - pnegri@gmail.co
Negri Pablo - Outlook W
JetBrains Account
JetBrains Account
Pablo
JetBrains s.r.o. [CZ] | https://account.jetbrains.com/licenses
Apps
Dictionary definitions
Master Journal List -
LE DICTIONNAIRE - D
Journal Rankings on
Introduction to Cond
Variable-state Late
Facial Expression Dat
Facial Landmark Dete
JET BRAINS
Licenses
Your Account
Help
Pablo Negri
Pablo Negri
Transactions
1 License
Export licenses into .xls
JetBrains Product Pack for Students
Download
License ID:
52W2XXUXV1
Licensed to: Pablo Negri
License restriction: For educational use only
Valid through: July 26, 2018
Following products included:
Download activation code for offline usage
IntelliJ IDEA Ultimate
ReSharper
ReSharper C++
dotTrace
dotMemory
dotCover
AppCode
CLion
PhpStorm
PyCharm
RubyMine
WebStorm
DataGrip
Rider
After downloading and installing the software, simply run it and follow the on-screen prompts to sign in with your JetBrains Account.
Can't find your license here? Link your past purchases to your JetBrains Account by providing a license key or domain .

CLion

En el caso de utilizar Windows, es necesario instalar MinGW. Esta aplicación tiene el **gcc** necesario para la compilación. En `https://www.jetbrains.com/help/clion/quick-tutorial-on-configuring-clion-on-windows.html` estan los pasos para la instalción en Windows.

CLion

- En el Taller de Algo-1 vamos a utilizar un entorno de desarrollo de C++ (IDE) denominado CLion.
- Esta IDE permite crear proyectos, editar archivos, compilar, *debuggear*, entre otras cosas.
- Es una herramienta que se puede descargar bajo licencia estudiantil, y está instalada en las PCs del laboratorio.

Pantalla de bienvenida del CLion

- Para lanzar el CLion en las PCs del laboratorio, abrir una consola (boton derecho del mouse sobre el escritorio), y escribir:

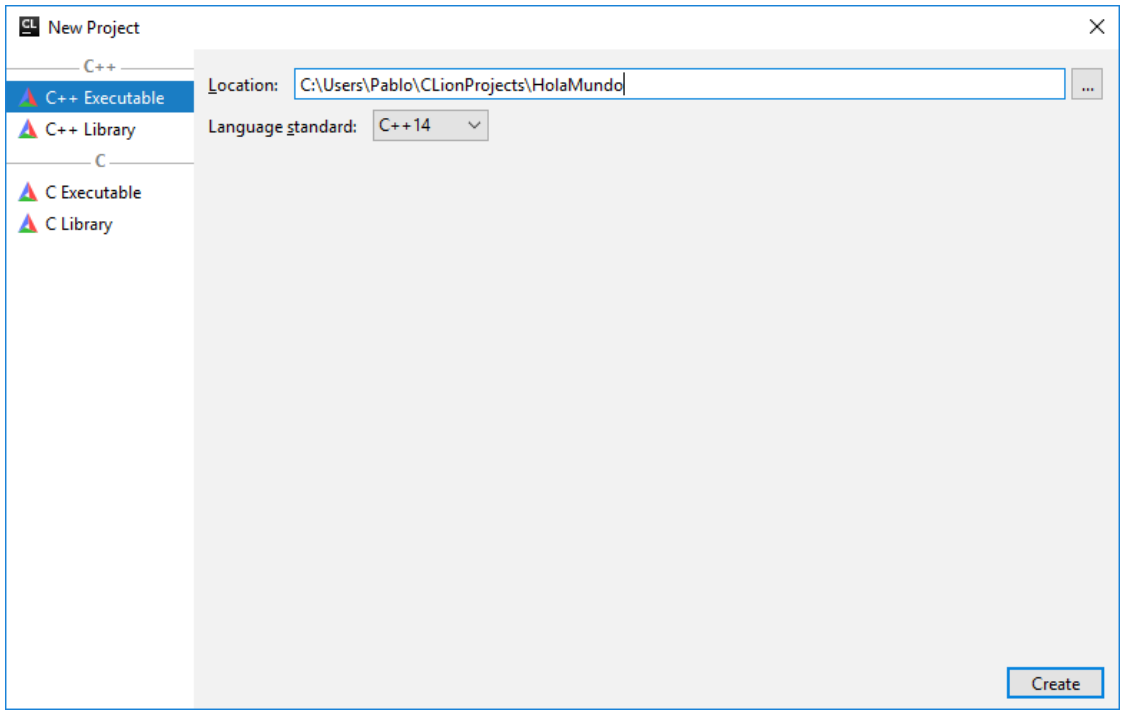
  `clion.sh`

Pantalla de bienvenida del CLion

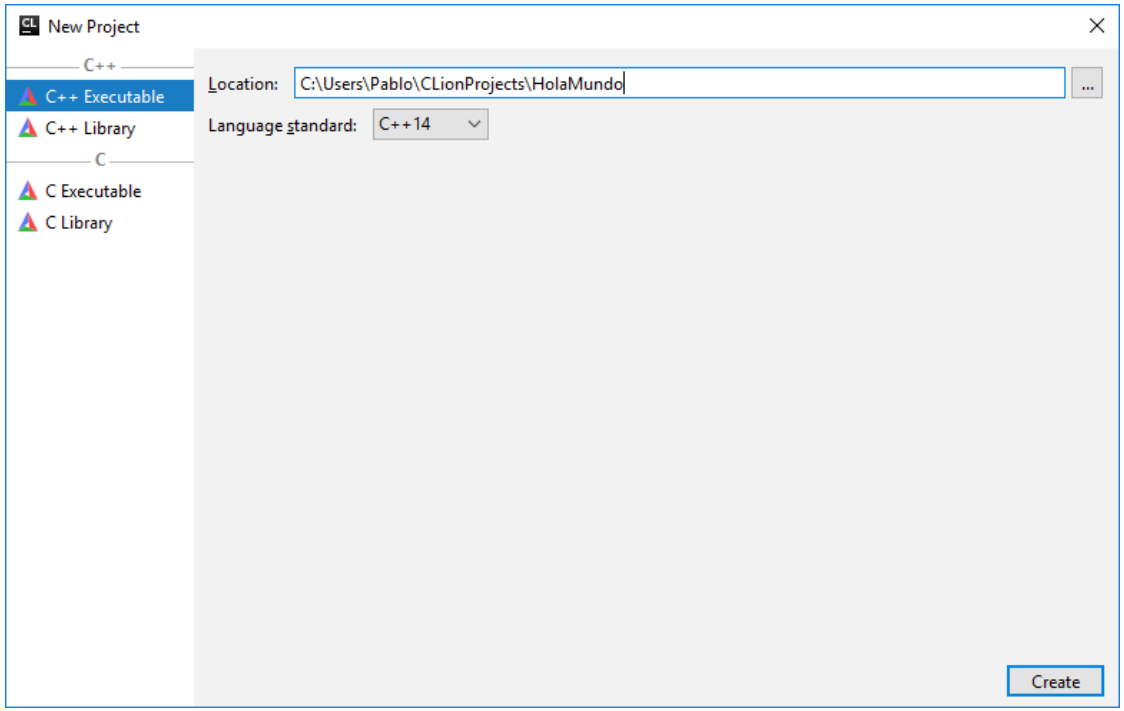
- Para lanzar el CLion en las PCs del laboratorio, abrir una consola (boton derecho del mouse sobre el escritorio), y escribir:

  `clion.sh`

- La ventana de inicio puede mostrar:



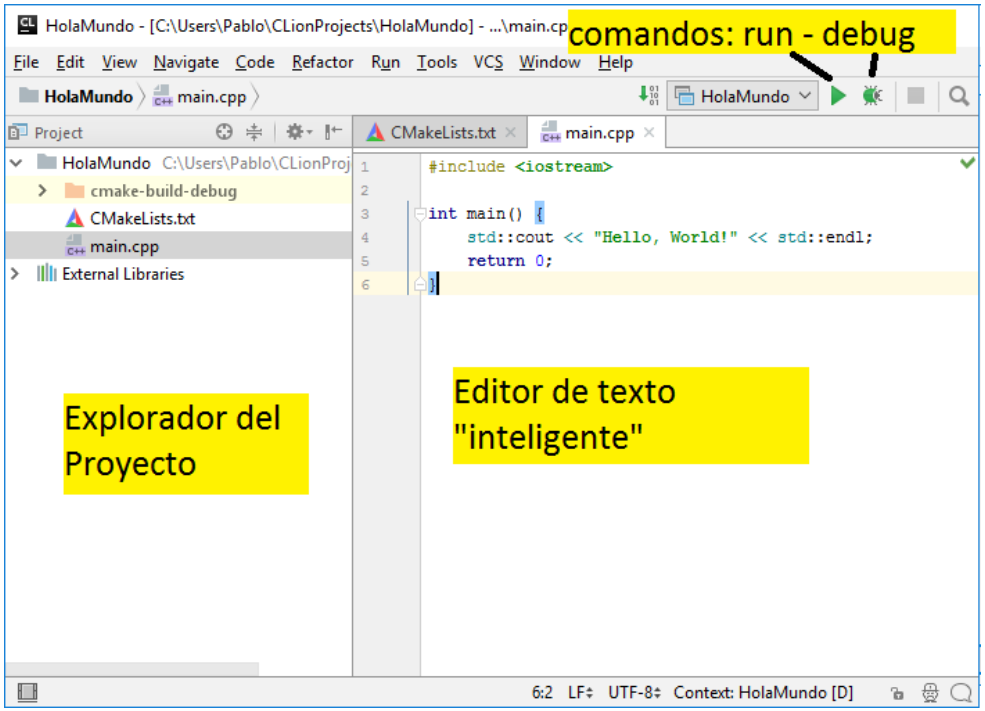
Pantalla de bienvenida del CLion



- La ventana de nuevo proyecto permite seleccionar el directorio de destino y el tipo de aplicación.
- En el ejemplo, el directorio destino es "HolaMundo", y el tipo de aplicación "C++ Executable".

Proyecto Hola Mundo

- La IDE contiene dos paneles principales:
  1. Explorador de los archivos del proyecto
  2. Editor de texto

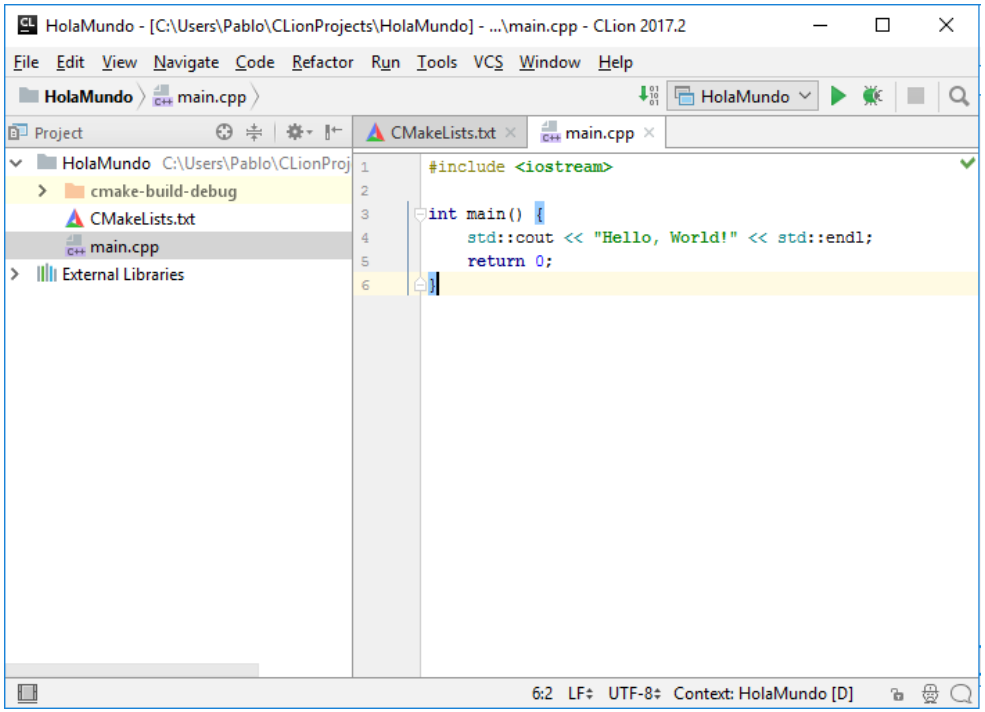


Proyecto Hola Mundo

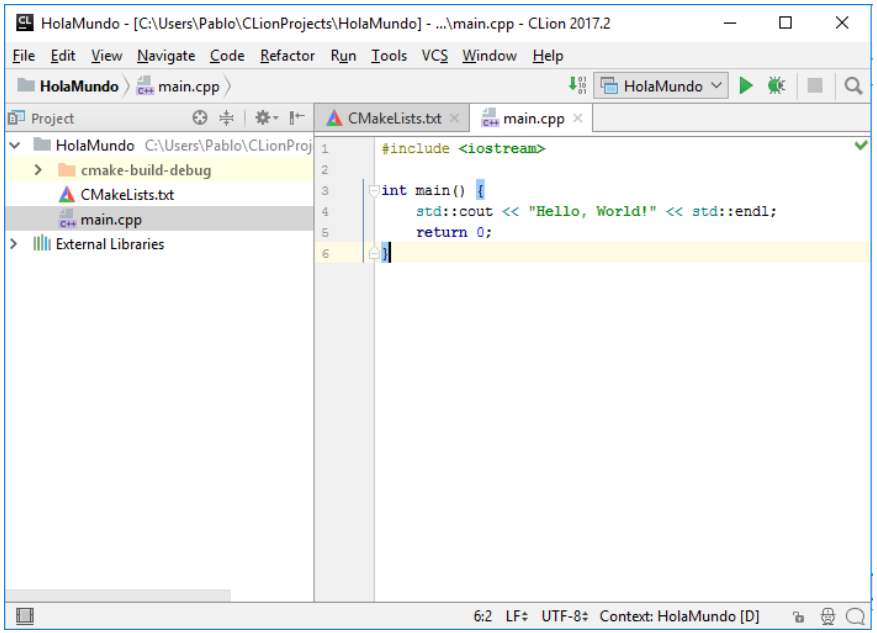
- CLion genera autoáticamente dos archivos para el proyecto:
  1. main.cpp
  2. CMakeList.txt

Proyecto Hola Mundo

- CLion genera autoáticamente dos archivos para el proyecto:
  1. main.cpp
  2. CMakeList.txt
- y muestra al usuario la IDE



Proyecto Hola Mundo



- Primera sorpresa: En el editor de texto, podemos ver que el archivo main.cpp no está vacio.
- `#include <iostream>`

```
int main() {
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

Proyecto Hola Mundo

```
#include <iostream>

int main() {
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

¿Que hace el programa?

Proyecto Hola Mundo

```
#include <iostream>

int main() {
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

- `#include <iostream>` sirve para incorporar al programa un set de funciones.
  1. "#" es un caracter que indica al compilador una instrucción especial
  2. "include" instrucción que incluye la librería que puede estar entre llaves o parentesis
  3. "iostream" es una libreria de funciones para el manejo de entrada/salida de c++

Proyecto Hola Mundo

```
#include <iostream>

int main() {
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

- `int main() {` define el inicio del bloque de la función principal del programa. Por convención, el programa siempre inicia su ejecución en esta funcion. Podemos ver dos características:
  1. Esta función main no recibe parámetros "()"
  2. La función devuelve una variable de tipo **int** o sea entero.
  3. { Es la llave de apertura de la función.

Proyecto Hola Mundo

```
#include <iostream>

int main() {
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

- `std::cout << "Hello, World!" << std::endl;`
  1. `std` es una librería de funciones definida dentro de iostream. Para acceder a una de esas funcions se usan dos puntos consecutivos `::`
  2. `cout` es una instrucción para imprimir en pantalla el mensaje entre << y <<, aqui es un mensaje de tipo String, entre comillas ", pero es capaz de imprimir todo tipo de variables.
  3. La instrucción `endl` indica el fin de la línea (retorno de carro)
  4. `;` finaliza la instrucción. El compilador lo precisa para identificar que termina la línea de instrucción. Olvidarse el punto y coma representa el 50 % de los errores de compilación en C++ :-)

Proyecto Hola Mundo

```
#include <iostream>

int main() {
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

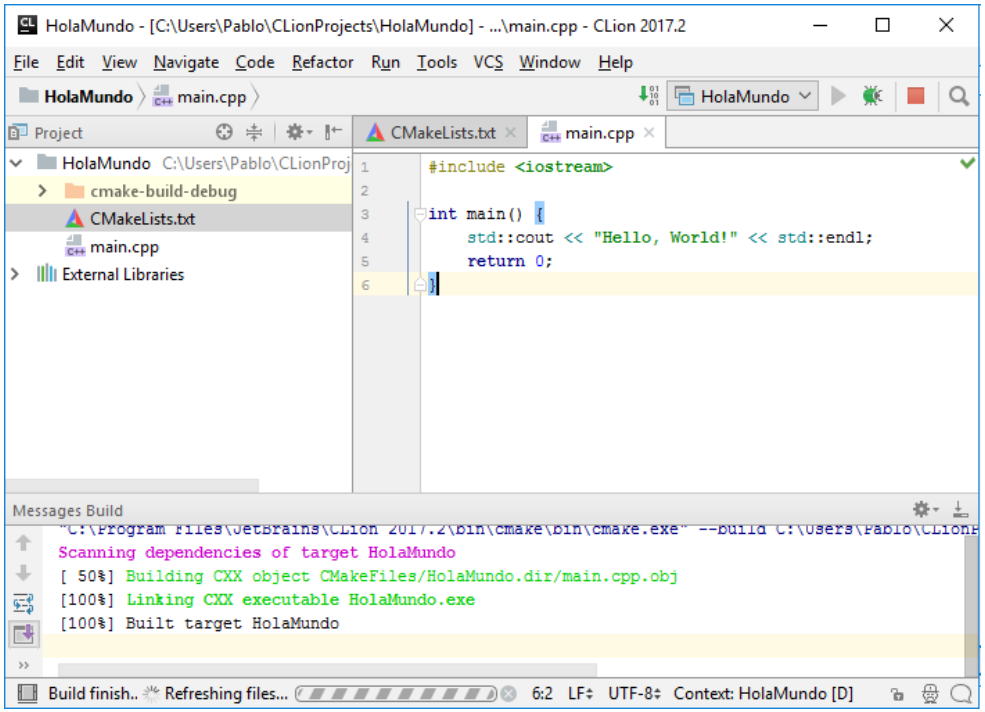
- `return 0;` El programa termina retornando 0, que significa que no hubo errores en la ejecución.
- `}` La llave de cierre termina el bloque de la función main. Olvidarse de los cierres de bloques es otra fuente común de error de compilación.

Proyecto Hola Mundo

- Ahora vamos a compilar el programa para generar el archivo ejecutable.

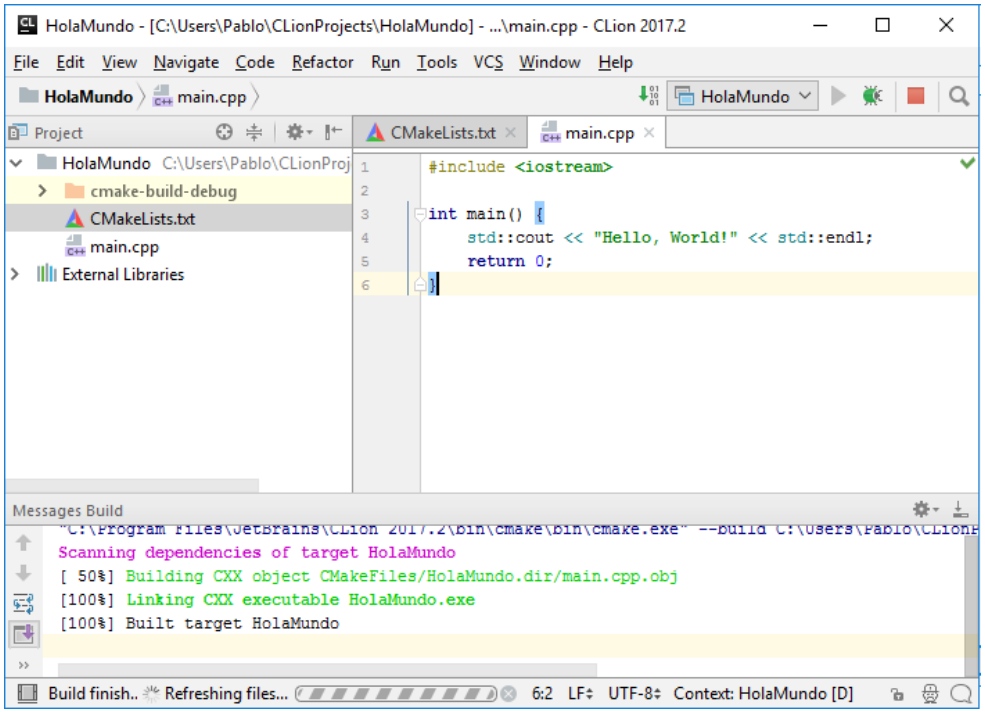
Proyecto Hola Mundo

- Ahora vamos a compilar el programa para generar el archivo ejecutable.
- Esto se hace con CLion utilizando el icono verde de RUN.
- La IDE abre un nuevo panel que muestra el avance y estado de la compilación.



Proyecto Hola Mundo

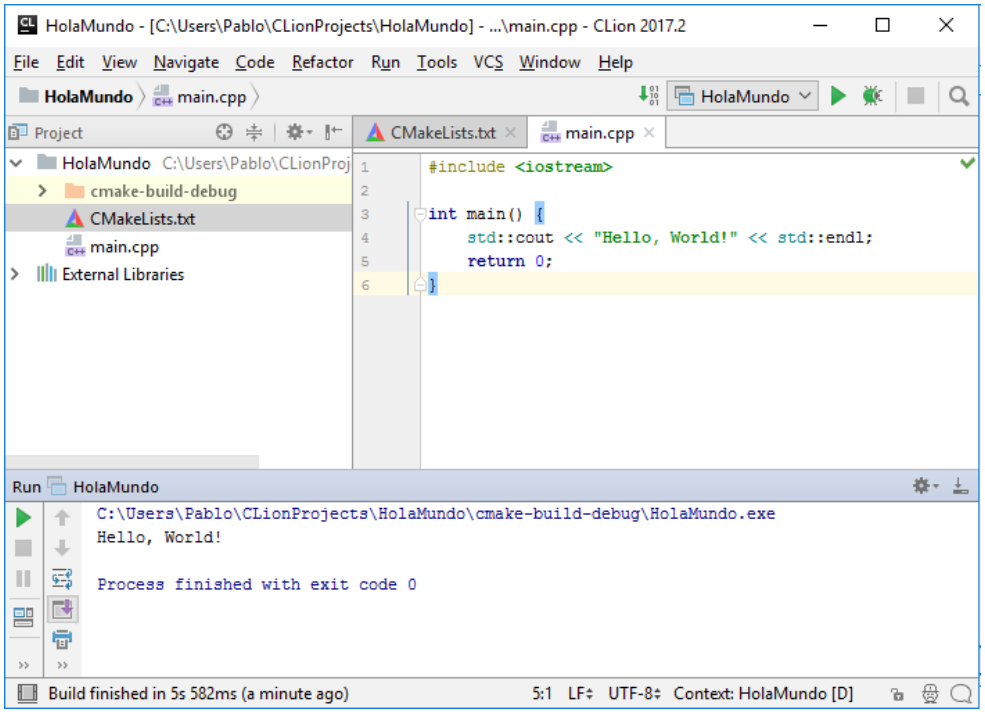
- Ahora vamos a compilar el programa para generar el archivo ejecutable.
- Esto se hace con CLion utilizando el icono verde de RUN.
- La IDE abre un nuevo panel que muestra el avance y estado de la compilación.



- Podemos ver que el cmake ejecutó secuencialmente una compilación, un linking y terminó por construir el ejecutable.

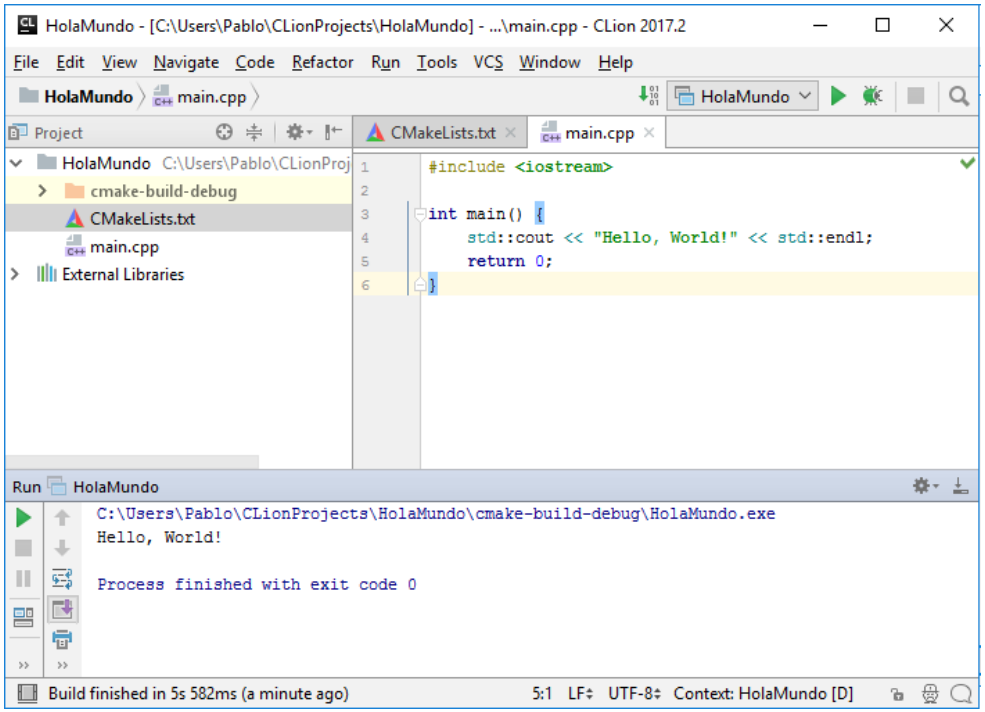
Proyecto Hola Mundo

- Luego de la compilación, el IDE ejecuta automáticamente el programa y muestra el resultado en un nuevo panel.



Proyecto Hola Mundo

- Luego de la compilación, el IDE ejecuta automáticamente el programa y muestra el resultado en un nuevo panel.



- En este panel podemos ver efectivamente, el mensaje generado por la función `cout`.

Proyecto Hola Mundo

- Que pasa cuando la compilación falla por un error?

Proyecto Hola Mundo

- Que pasa cuando la compilación falla por un error?
- Ejecutemos el programa con errores de sintaxis:
  1. Quitar el punto y coma al final de la linea del `cout`.
  2. Borrar la llave de cierre del `main`.
- Analizar el mensaje de error que nos devuelve el IDE.
- Es útil para resolver el problema?.