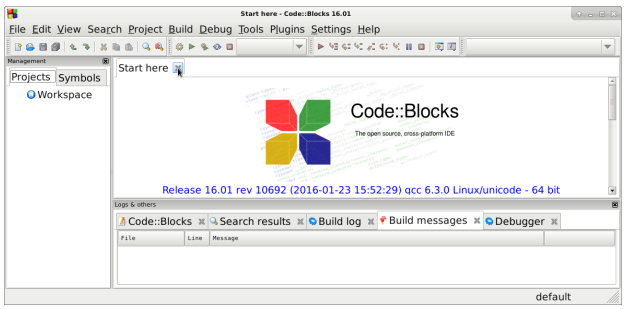
Guía del Entorno de Programación CodeBlocks

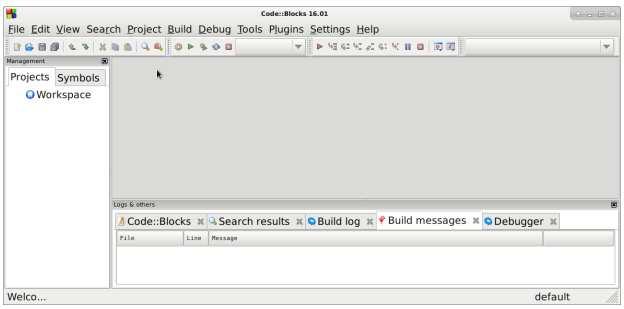
Dpto. Lenguajes y Ciencias de la Computación. E.T.S.I. Informática.

Introducción a la Programación

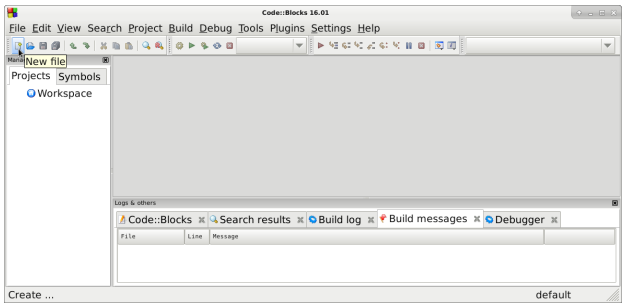
Si tiene problemas con el funcionamiento del **compilador de C++** del *entorno de programación* **CodeBlocks**, se recomienda realizar las acciones mostradas en la sección de **Configuración**, al final de este documento.

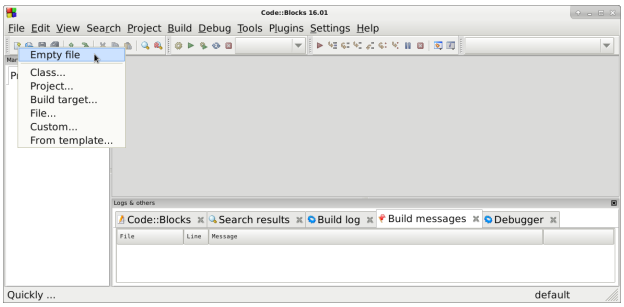
El *entorno de programación* (IDE) **CodeBlocks** nos permite crear, editar y compilar ficheros de código fuente en C++.



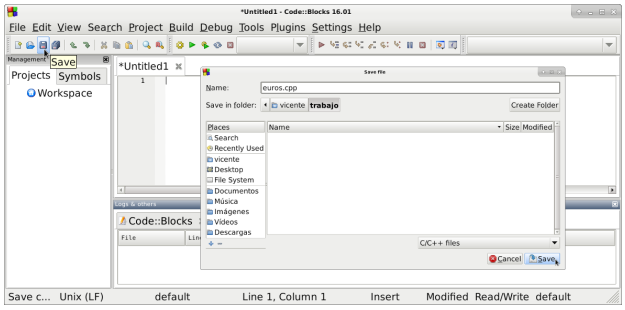


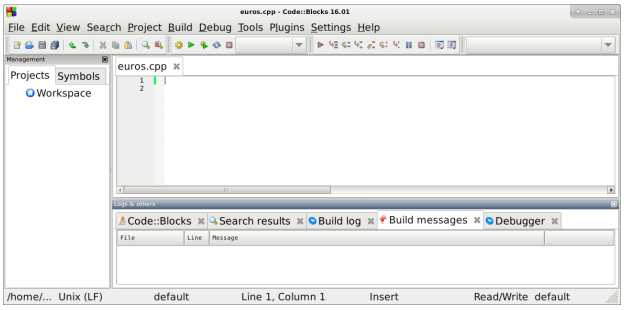
Pulsar sobre el icono **NewFile** y seleccionar **EmptyFile** nos permite crear un nuevo fichero de código fuente en C++.





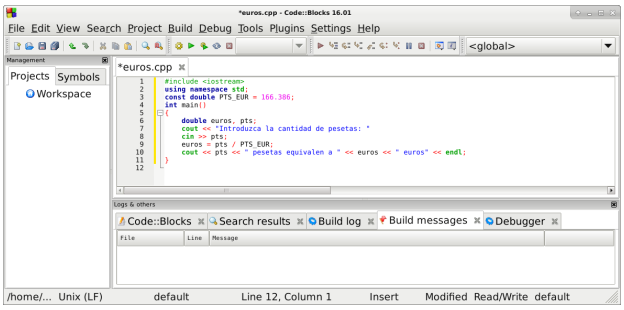
Es conveniente que a continuación pulsemos sobre el icono de **Save** (**Guardar**) para asignarle un nombre de fichero al código fuente en C++ que vamos a desarrollar. Además, habrá que seleccionar la carpeta donde se almacenará el fichero. Todos los ficheros en C++ que desarollaremos deberán tener la extensión `.cpp`. Guardar el fichero C++ con la extensión `.cpp` permitirá que el código sea coloreado para facilitar su edición y legibilidad, así como permitirá que el código fuente sea compilado como código **C++** (si no se especifica explícitamente la extensión `.cpp`, entonces se le asigna automáticamente la extensión `.c` y la compilación sera errónea). En este caso concreto, el fichero se llamará `euros.cpp`.





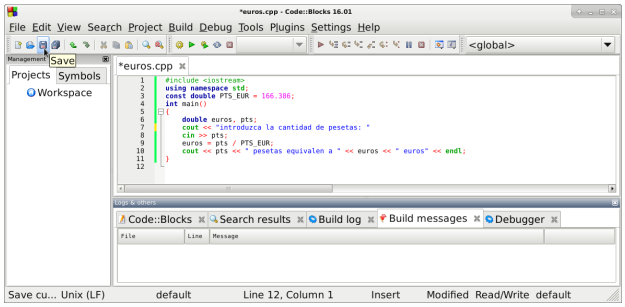
Codifique, utilizando el editor de textos del entorno de desarrollo, el siguiente programa C++, exactamente igual que aparece en el siguiente texto.

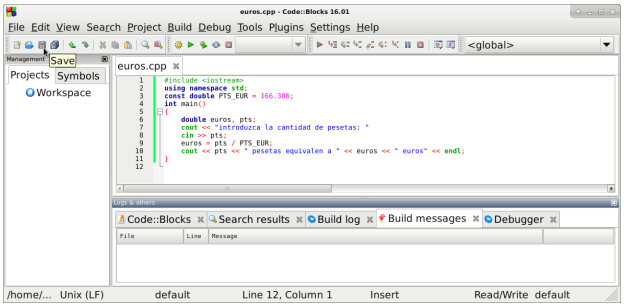
```
#include <iostream>
using namespace std;
const double PTS_EUR = 166.386;
int main()
{
    double euros, pts;
    cout << "Introduzca la cantidad de pesetas: "
    cin >> pts;
    euros = pts / PTS_EUR;
    cout << pts << " pesetas equivalen a " << euros << " euros" << endl;
}
```



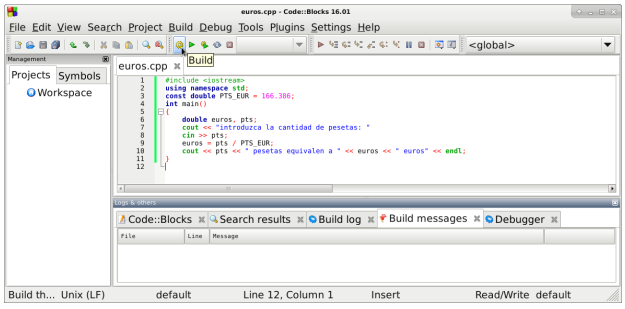
Mientras se edita un programa, si la versión actual no ha sido **guardada** en el fichero en memoria secundaria (permanente), entonces aparece un símbolo asterisco (*) en la pestaña junto al nombre del fichero.

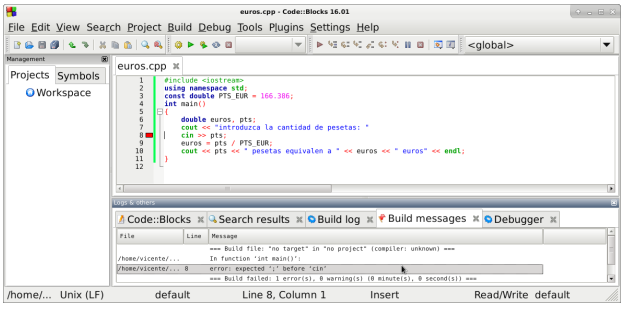
Para guardar en memoria secundaria el programa modificado, es necesario pulsar el icono de **Save** (**Guardar**) o pulsar la combinación de teclas **CTRL-S** (nótese como desaparece el símbolo * de la pestaña). Si el fichero es nuevo, en la operación de guardar habrá que seleccionar la carpeta y el nombre del fichero (con la extensión `.cpp`).





Podremos compilar el código fuente en C++ almacenado en el fichero **`euros.cpp`** pulsando en el icono **`Build`**, y producirá el fichero ejecutable **`euros`** en caso de que la compilación sea correcta:

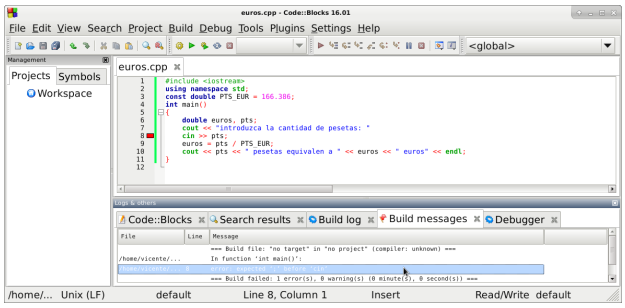


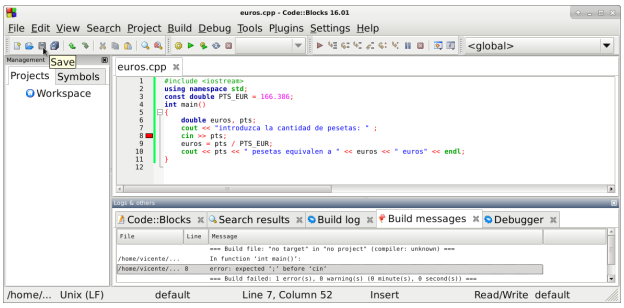


En este caso, existe un error en el programa, que el compilador señala en la línea **8**, indicando que se esperaba encontrar un símbolo punto y coma (**;**) antes de la palabra **cin**. En realidad, el error se encuentra en la **línea anterior**, ya que la sentencia de la línea **7** no ha sido terminada con el símbolo punto y coma (**;**).

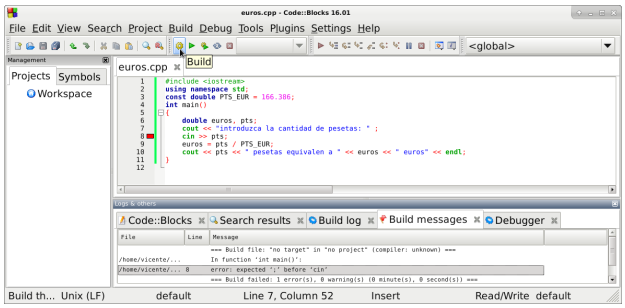
> A veces puede suceder que el compilador detecte y marque el error en una determinada línea, pero realmente el error se encuentra en algunas líneas anteriores. Adicionalmente, también puede suceder que los errores en algunas líneas anteriores también afecten y provoquen errores en las líneas siguientes, por lo que de forma práctica, sólo intentaremos resolver los primeros errores resultado de la compilación, y procederemos iterativamente hasta que no haya errores de compilación.

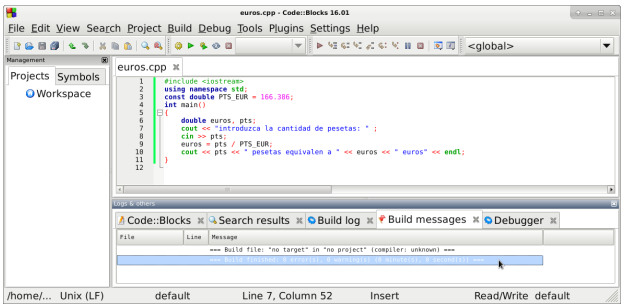
Por lo tanto deberemos corregir el error (añadiendo el símbolo **;** al final de la línea **7**) y **Guardar** en memoria secundaria la nueva versión corregida del programa.



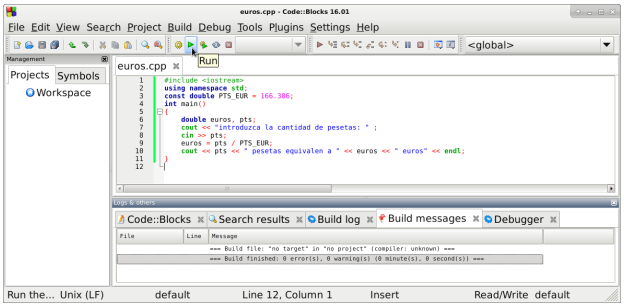


Si volvemos a compilar el fichero **euros.cpp** (ya corregido) pulsando en el icono **Build**, ahora la compilación del programa C++ es correcta, y el resultado de la compilación muestra que no hay errores, por lo que se generará un fichero ejecutable (programa) denominado **euros**.

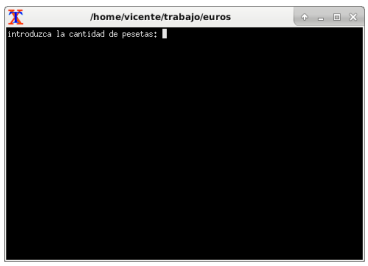


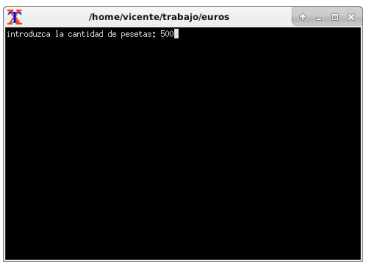


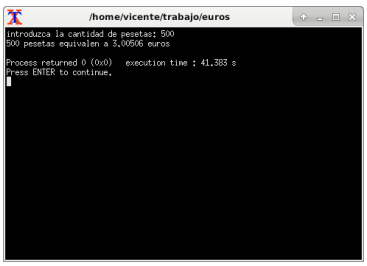
Podemos ejecutar el programa anterior compilado correctamente si pulsamos en el icono **Run**.



En la ejecución de este programa, se muestra un mensaje solicitando la introducción de una cantidad de pesetas, el usuario debera teclear la cantidad que desee (por ejemplo 500) y pulsar la tecla **Enter**, y finalmente el programa mostrará el resultado de la computación (3.00506 en este caso). Para finalizar la ejecución del programa, el usuario debera pulsar la tecla **Enter**.

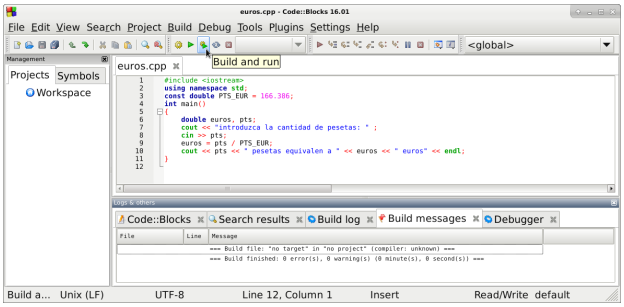




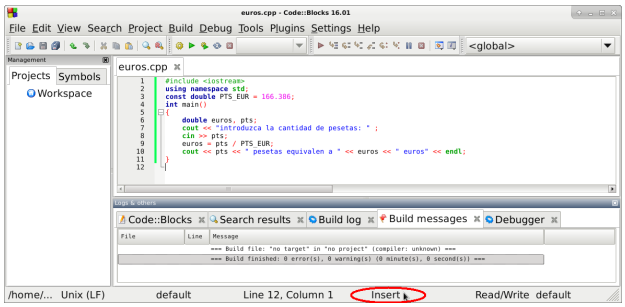


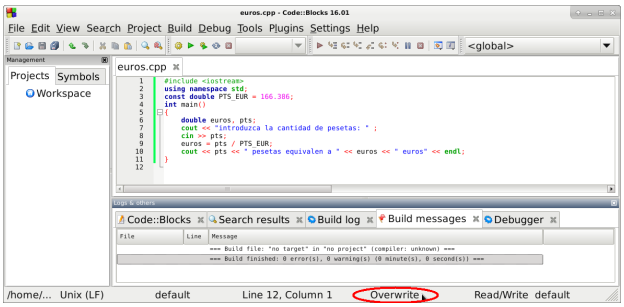
Algunas Acciones Útiles

Podemos realizar las dos acciones (**Compilar** y **Ejecutar**) consecutivamente si pulsamos en el icono **Build and Run** (o la tecla **F9**).

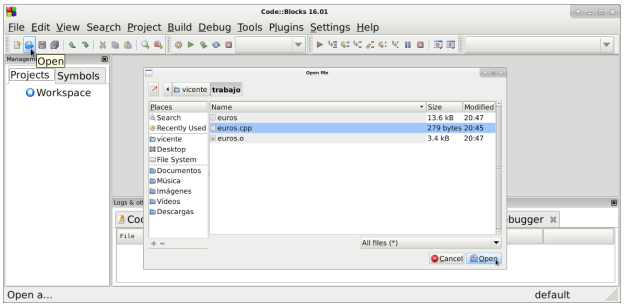


Durante la edición de un programa, el programador puede utilizar la tecla **Insert** para conmutar entre el modo **Insert** (insertar) y el modo **Overwrite** (sobreescribir).

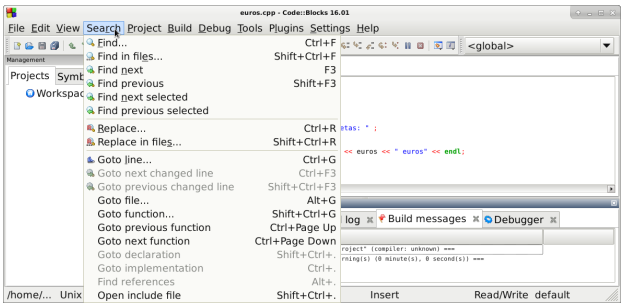




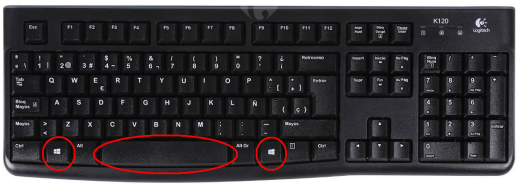
Además de utilizar el icono **NewFile** para crear un nuevo fichero, el programador tambien puede utilizar el icono **Open** para abrir y editar un fichero ya existente.



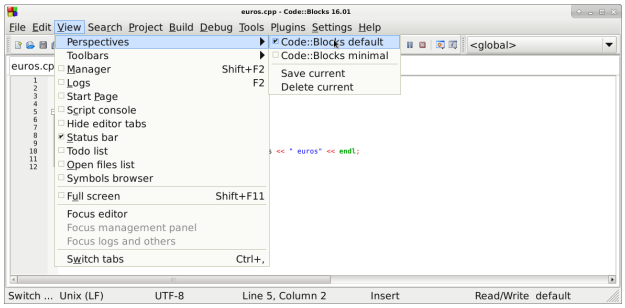
La pestaña **Search** de la barra de menú superior permite realizar búsquedas y reemplazamientos en todo el texto del fichero fuente.

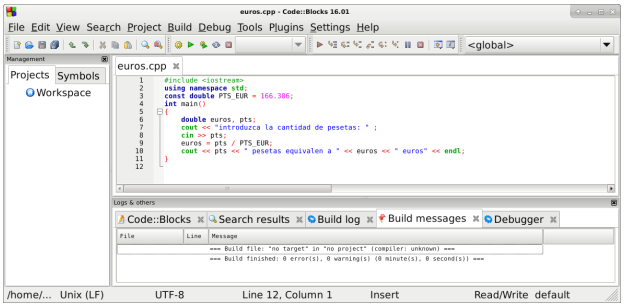


La combinación de una tecla **Windows** y la **Barra Espaciadora** (pulsadas simultáneamente) permite commutar el idioma del teclado.

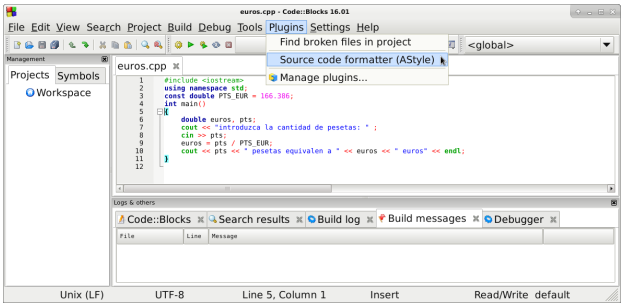


La opción **Code::Blocks default** del menú **Perspectives** de la pestaña **View** permite restaurar el estado de visualización a la perspectiva por defecto.

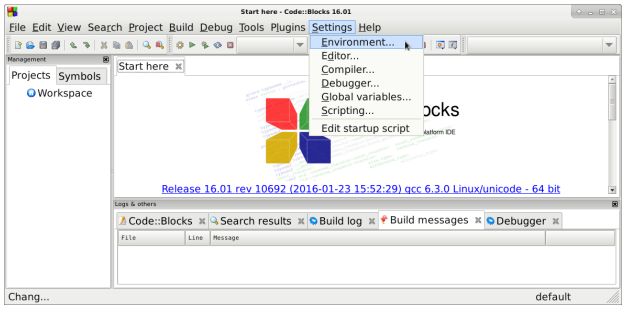


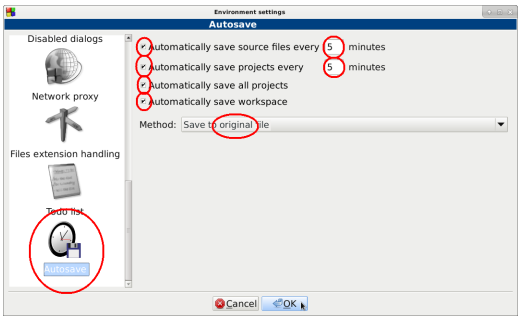


Aunque es recomendable que el programador codifique el programa en C++ en el formato y estilo adecuado, a veces nos podemos encontrar con código que no este formateado adecuadamente. En este caso, la opción **SourceCodeFormatter** de la pestaña **Plugins** aplicará un formato automático al código fuente.



También es posible configurar el entorno de desarrollo para que **guarde automáticamente** el contenido de los ficheros de forma periódica cada cierto intervalo de tiempo. En este caso, los ficheros se guardan automáticamente sobre el fichero original.





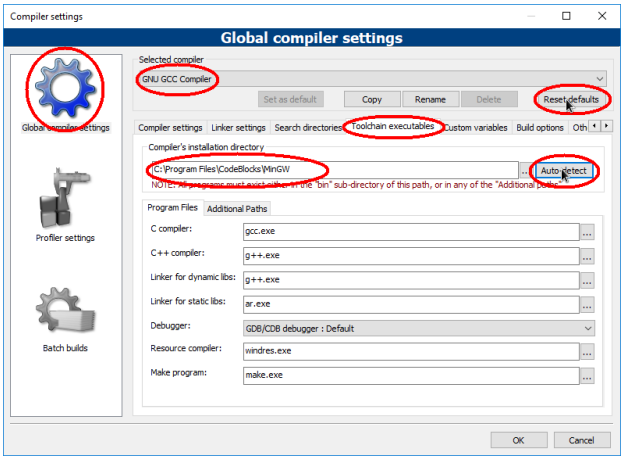
Tildes en el Sistema Operativo Windows

En el sistema operativo Windows, el *IDE Code::Blocks* suele dar problemas en el tratamiento y manipulación de caracteres con tildes. Para resolver este problema, se puede incluir la biblioteca `<windows.h>` e invocar a los procedimientos que se indican en el siguiente ejemplo:

```
#include <iostream>
#include <windows.h>
/* resto del programa */
int main()
{
    SetConsoleOutputCP(1252);
    SetConsoleCP(1252);
    /* resto del programa */
}
```

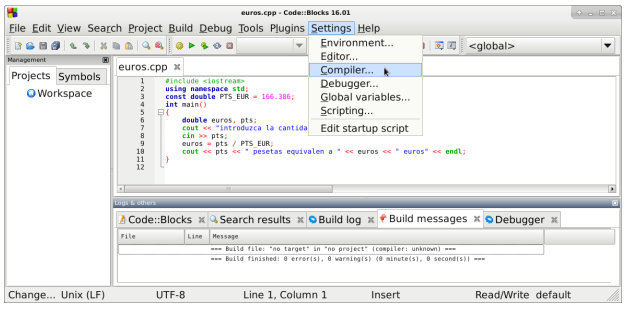
Configuración

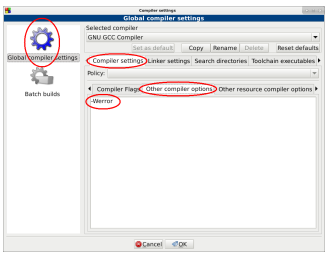
A veces, el entorno de desarrollo *Code::Blocks* no está bien configurado, y no encuentra el compilador de C++ para poder compilar los programas, o el compilador no trabaja adecuadamente. En este caso, debemos seleccionar las siguientes opciones en la pestaña **Settings** de la barra de menú superior, seleccionando la opción **Compiler**, seleccionando la opción **GlobalCompilerSettings**. Pulsaremos sobre el botón **Reset-defaults**, y además, seleccionamos las opciones marcadas en la siguiente imagen y pulsamos sobre el botón **Auto-detect**. Finalmente, seleccionaremos las opciones de compilación que se indican en la siguiente sección.

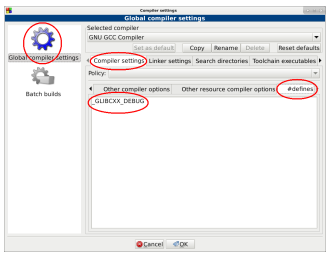


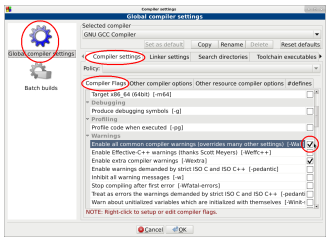
Opciones de Compilación

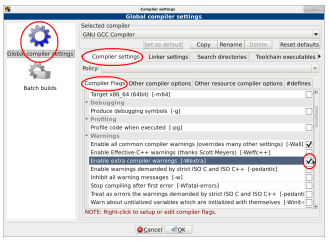
Podemos especificar las opciones de compilación en la pestaña **Settings** de la barra de menú superior, seleccionando la opción **Compiler**, y seleccionando la opción **GlobalCompilerSettings**. Cuando se compila código C++, es importante que las opciones marcadas en las siguientes imágenes sean seleccionadas.

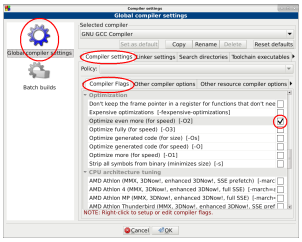












Nótese que la opción de compilación `-Werror` hace que la compilación del código fuente en C++ sea más estricta, y que determinados **avisos** de *posibles problemas* que indica el compilador, ahora serán marcados como **errores**, que obligarán al programador a **corregirlos** antes de poder generar el código ejecutable. Esta opción permite desarrollar software más seguro, ya que permite detectar y corregir durante la compilación errores que de otra forma tendrían que ser detectados durante la ejecución del programa. No obstante, a veces el compilador también marcará como *error* código que realmente no sea erróneo ni peligroso (por ejemplo que una determinada variable no sea utilizada), aunque en la mayoría de las situaciones, corregir estos *errores* y *avisos* durante la compilación permitirá desarrollar software más seguro.