Introducción al Dev-Cpp

El entorno de desarrollo Dev-C++

Es gratis y te lo puedes bajar de aquí para practicar en casa:

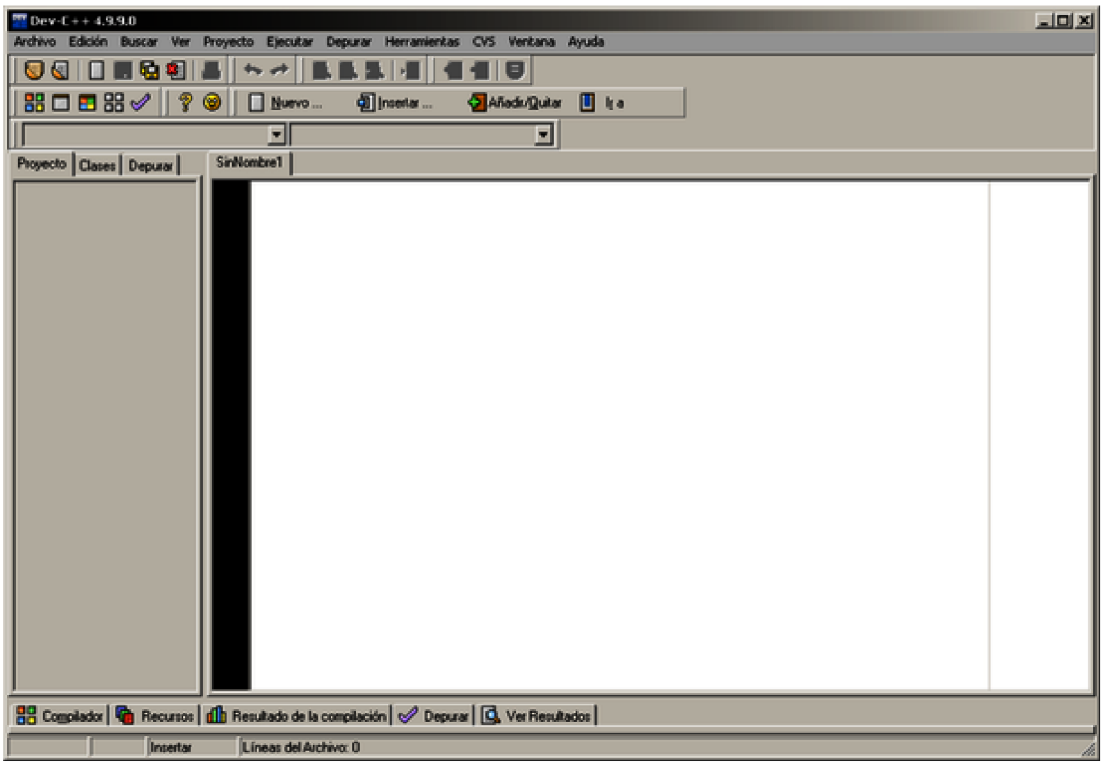
> http://www.bloodshed.net/devcpp.html
>
> (también puedes buscar `devcpp` en Google)

Es lo que se llama un IDE o *entorno integrado de desarrollo*, incluye:

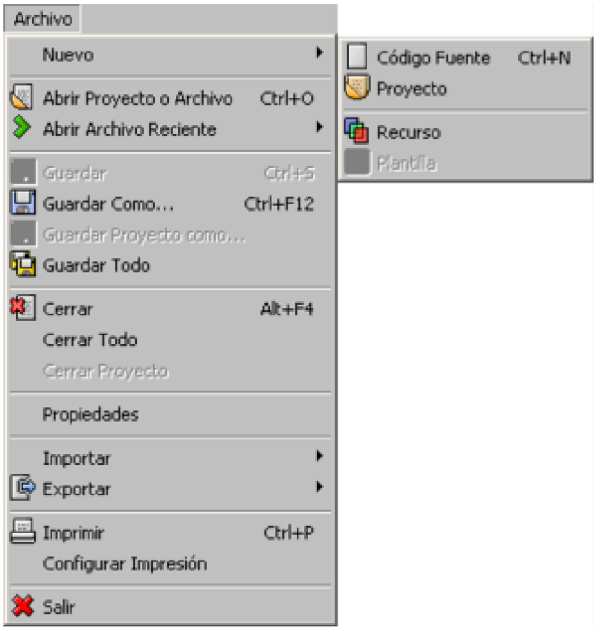
- Un editor, es como un procesador de texto, pero resulta más adecuado para escribir código.
- Compilador y enlazador para el lenguaje C (también C++ pero no lo vamos a utilizar).

A falta de un entorno integrado, podrías utilizar por separado un editor de textos (en windows tienes el `textpad`, por ejemplo), un compilador, etc. pero el entorno resulta mucho más cómodo.

Visión del entorno Dev-C++



Editor



- Para crear un archivo fuente nuevo (tecla **Ctrl+N**):

  Archivo -> Nuevo -> Código Fuente

- Dispone de una ventana en la que se edita el **código fuente** de nuestros programas. Puedes tener varios ficheros abiertos, cada uno en una pestaña.

Tipos de ficheros

- **Código Fuente (.c):** fichero de texto con el programa.
- **Objeto (.o .obj):** Son ficheros intermedios creados a partir de la compilación. Olvídate porque son transparentes al programador.
- **Biblioteca (.a .lib):** Similar a los ficheros objeto. Suelen agrupar funciones estándar.
- **Cabecera (.h):** Contienen definiciones y declaraciones. Suelen corresponder a funciones estándar de las librerías.
- **Ejecutables (.exe):** nuestro programa transformado en formato ejecutable.

¡IMPORTANTE!

- Utiliza la extensión **.c** para el código fuente, **evita la extensión .cc**
- Evita poner espacios en blanco en el nombre de tus programas.
- Salva periódicamente el programa, usando **Ctrl+S**.

El primer programa en C

Introduce el siguiente código fuente en el editor de Dev-C++ y guárdalo como "`hola.c`"

```
#include <stdio.h>
#include <conio.h>
int main() { /* cabecera de la funcion principal */
  fprintf(stdout,"Hola!\n"); /* imprime en pantalla */
  getch(); /* espera a que pulses una tecla */
  return 0; /* hace falta porque main lleva int */
}
```

¡Es normal casi todo te suene a chino! *No te preocupes, todo está bajo control ;)*

- Busca el fichero con el **explorador de archivos** (tecla de windows + 'E') y ábrelo con un doble click

Como ves, se trata de un fichero de texto que puedes escribir con un editor cualquiera

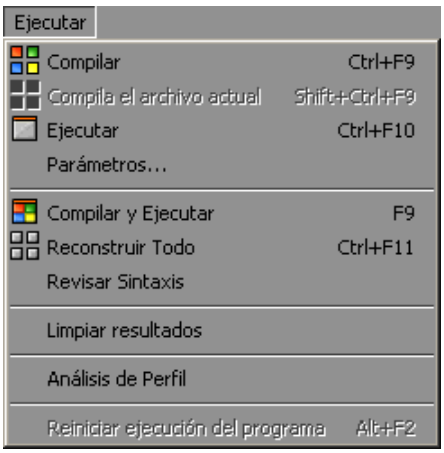
El proceso de la compilación

- El compilador es un traductor que convierte el **código fuente** de tu programa en **código máquina**.
- Tu programa normalmente incluye **bibliotecas** para poder utilizar funciones matemáticas, de entrada y salida (leer de teclado, escribir en pantalla, etc.).
- Tras enlazar estas bibliotecas, se genera un fichero con el programa **ejecutable**.

Visión sencilla:

- El compilador genera un **ejecutable** a partir del **codigo fuente** .

Compilación y ejecución



- Para compilar y ejecutar (tecla **F9**):

  Ejecutar -> Compilar y Ejecutar

- Para compilar un archivo fuente (tecla **Ctrl+F9**):

  Ejecutar -> Compilar

Vamos a probar

Ahora que ya tienes este programa en el fichero "`hola.c`"

```
#include <stdio.h>
#include <conio.h>
int main() { /* cabecera de la funcion principal */
  fprintf(stdout,"Hola!\n"); /* imprime en pantalla */
  getch(); /* espera a que pulses una tecla */
  return 0; /* hace falta porque main lleva int */
}
```

¡Vamos a probarlo!

- Dale a compilar y a ejecutar (tecla **F9**) para ver lo que hace el programa, pulsa una tecla para terminar.
- Busca en la misma carpeta de antes y verás que aparece un fichero llamado "`hola.exe`", haz doble click sobre él

---

Aprendamos algo de C

Ahora modifica el programa para que quede así:

```
#include <stdio.h>
#include <conio.h>
int main() { /* cabecera de la funcion principal */
  fprintf(stdout,"Hola!\n");  /* imprime en pantalla */
  fprintf(stdout,"mundo!\n"); /* imprime en pantalla */
  getch(); /* espera a que pulses una tecla */
  return 0; /* hace falta porque main lleva int */
}
```

- Lo que está escrito entre `/*` y `*/` son **comentarios** y son **ignorados por el compilador**, ¡los podrías quitar sin problema!
- Si lo ejecutas, verás que el programa ejecuta una instrucción `fprintf` y después la otra, **en orden**.
- Prueba a quitar el `!\n` que está después de `Hola!` y vuelve a compilar y ejecutar. Como has podido comprobar, `\n` es la forma de indicar un **cambio de línea**, quizás te convenga añadir un espacio en blanco.
- Elimina la línea `getch();` y vuelve a ejecutar ¿qué ha pasado?

Errores :(

- Evidentemente pueden producirse errores en el proceso.
- La ventana del compilador está en la parte inferior de la pantalla, apareciendo si ocurre algún error al compilar.
- También aparece si el compilador ve algo que quizás no sea un error, pero es sospechoso: un *warning* o advertencia, es la manera que tiene el compilador de decirte:

    "no es incorrecto pero me huele mal"

Tipos de error

Errores sintácticos.

- Los lenguajes de programación siguen una sintaxis. Si compilador detecta defectos de sintaxis muestra este tipo de errores. Ej.: Dejarse un punto y coma al acabar una instrucción.

Vamos a forzar un error sintáctico

- Elimina algún punto y coma "`;`" y dale a compilar.

    Fijate en la ventana inferior, es importante aprender a interpretar los mensajes del compilador.

Tipos de error

Errores en el enlace.

- Se suele tratar de errores a la hora de nombrar las funciones, en los tipos o número de parámetros o del lugar donde se encuentran al llamar a una función…

Vamos a forzar un error de enlazado

- Pon `fprint` en vez de `fprintf`. Trata de compilar y fíjate en el mensaje de la ventana de error.
- Vuelve a poner `fprintf` pero cambia `stdio` por otro nombre. Trata de compilar y fíjate en el mensaje de la ventana de error.
- Las líneas que empiezan con `#` son **directivas del preproceso**, en particular `#include` sirve para incluir una biblioteca. `stdio` es la biblioteca de **entrada y salida estándar** (**st**an**d**ard **i**nput **o**utput).

  > Ahí es donde se encuentra la función `fprintf`, por eso sin esa biblioteca no sabe enlazarlo para generar el ejecutable :(

Tipos de error

Errores en ejecución.

- Se dan durante la ejecución. Ejs.: División por cero, la raíz cuadrada de un valor negativo…

Errores semánticos.

- Discrepancias entre lo que hace el programa y lo que se pretende que haga ¡El entorno no puede ayudarnos!

Otro programa en C

```
#include <stdio.h>
#include <conio.h>
int main() {
  fprintf(stdout,"2+2 = %d\n", 2+2);
  getch();
  return 0;
}
```

Fíjate:

- la línea con `fprintf` es una **llamada a una función**.
- dentro de los paréntesis se encuentran los **argumentos de la función**:
    - `stdout` es un fichero que corresponde a la salida estándar o la pantalla del ordenador.
    - `"2+2 = %d\n"` es la **CADENA DE FORMATO** donde `%d` significa "**pon ahí un número entero**"
    - `2+2` es una expresión matemática, **todo lo que va tras la cadena de formato se usa en expresiones tipo** `%d`**

Forzando un error de ejecución

Modifica el programa:

```
#include <stdio.h>
#include <conio.h>
int main() {
  fprintf(stdout,"2/0 = %d\n", 2/0); /* OUCH! :( */
  getch();
  return 0;
}
```

¿Qué ocurre al ejecutarlo?

Precedencia y asociatividad

Modifica el programa:

```
#include <stdio.h>
#include <conio.h>
int main() {
  fprintf(stdout,"%d %d\n", 2+3*5, (2+3)*5);
  getch();
  return 0;
}
```

¿Por qué salen dos números diferentes?

Los caracteres son números

Prueba este programa:

```
#include <stdio.h>
#include <conio.h>
int main() {
  fprintf(stdout,"%c\n", 'A');  /* %c indica CARACTER */
  fprintf(stdout,"%c\n", 'A'+1);/* %c indica CARACTER */
  fprintf(stdout,"%d\n", 'A');  /* %d indica ENTERO   */
  fprintf(stdout,"%c\n", 65);   /* %c indica CARACTER */
  getch();
  return 0;
}
```

Y POR HOY YA BASTA :)