

ESTRUCTURA DE LENGUAJE DE PROGRAMACIÓN C++



1 0 1 0 1 1 1 1 1 0
1 0 1 0 1 1 1 1 1 0
0 1 0 1 0 0 0 0 0 1
0 0 0 0 1 0 0 0 0 0
0 1 0 1 1 0 0 0 0 1
1 1 1 1 1 1 1 1 1 1
1 0 1 0 1 1 1 1 1 0
1 0 0 0 1 0 0 0 0 0
0 1 0 1 0 0 0 0 0 1
0 0 1 0 1 1 1 1 1 0
0 1 0 1 1 0 0 0 1
0 1 0 1 0 0 0 0 1
0 1 0 0 1 1 1 1 0
0 1 0
1 1

Unidad 1



Elementos básicos del lenguaje de C++

Tabla de Contenido

1. Introducción	3
2. Estructura de contenido	4
3. Instalando el compilador.....	5
4. Componentes y tipos de datos	9
4.1. Archivos de cabecera	13
4.2. Comentarios	13
4.3. Identificadores	13
4.4. Palabras reservadas	14
4.5. Signos de puntuación	14
4.6. Tipos de datos	14
5. Declaración de variables	16
5.1. Variables locales	17
5.2. Variables globales	17
6. Funciones	17
6.1. Función main	17
6.2. Funciones declaradas por el usuario	17
7. Material de apoyo	19
8. Glosario	20
9. Referencias bibliográficas	21
10. Control del documento	21
Créditos	22
Creative Commons	22

1. Introducción

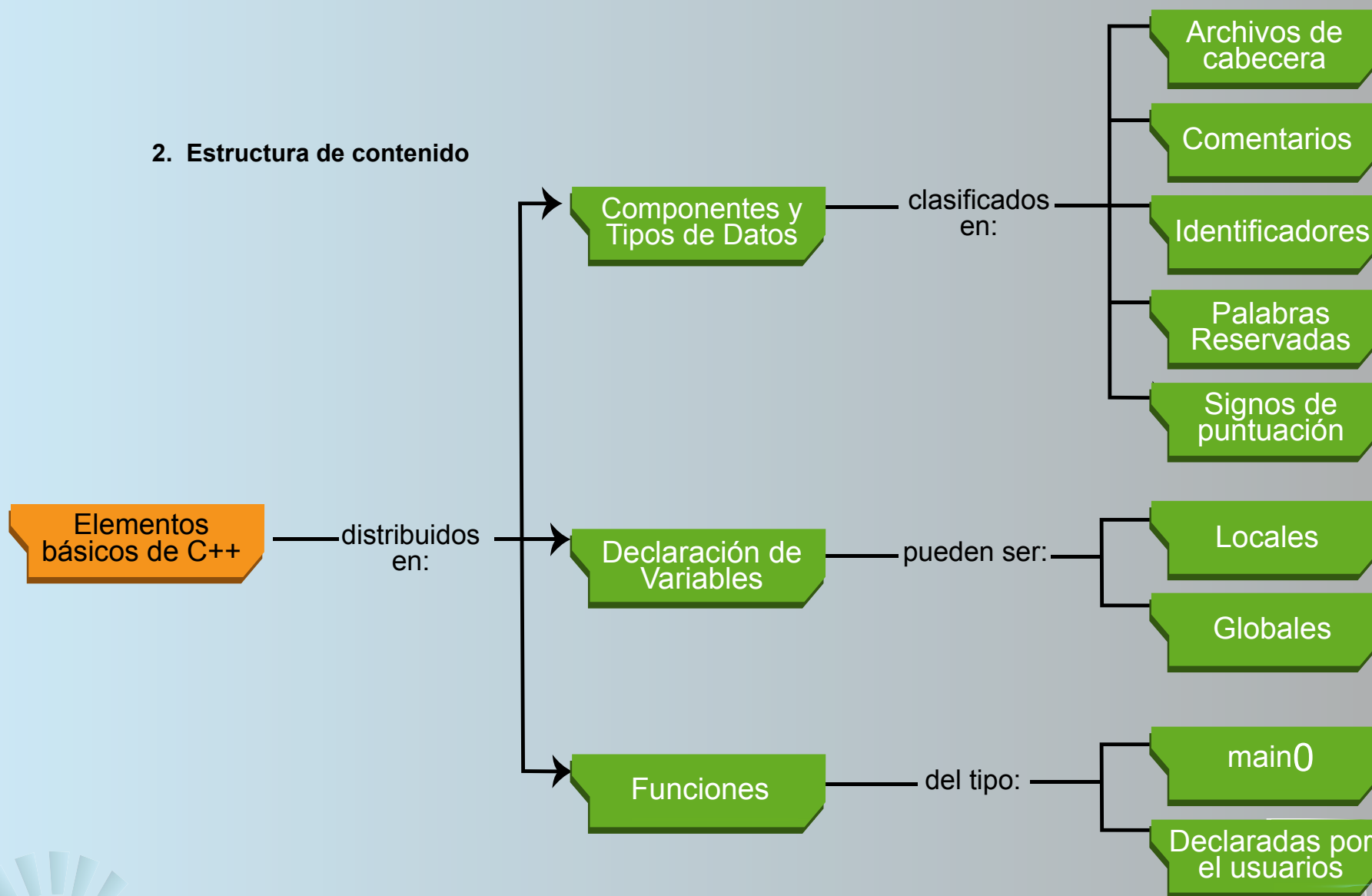
La comprensión de un lenguaje de programación está dada por la apropiación de conceptos básicos, terminología, sintaxis y semántica propia; que una vez adoptadas se deben poner en práctica a través del desarrollo del ejercicio.

Disponer de las herramientas esenciales de software y hardware permitirán al aprendiz avanzar exitosamente en el proceso de formación. El lenguaje C++ requiere para su desarrollo un entorno de edición y compilación, que permite la traducción al lenguaje máquina y a su vez la comprobación de la estructura de los programas que se desarrollen; dentro de ella, la secuencia de instrucciones, la definición de variables globales y locales, la declaración de funciones tanto propias del lenguaje, como definidas por el usuario, la incorporación de librerías, entre otros.

De igual forma, la identificación de los tipos de datos para la declaración de variables es fundamental en el desarrollo de aplicaciones, dado que de ello depende el flujo eficiente de la información, tanto de entrada como de salida. Esto se refiere, a los datos que requiere el computador para ser procesados con el fin de mostrar unos nuevos en la interfaz de usuario.



2. Estructura de contenido



3. Instalando el Compilador

Cuando se habla de un IDE en lenguajes de programación se refiere al acrónimo en inglés Integrated Development Environment o como se mencionaba en la introducción un Entorno de Desarrollo Integrado. En esta oportunidad se brindarán las orientaciones para instalar Dev C++ que es un entorno fácil de manejar con unos requisitos mínimos.

Lo primero es dirigirse a la página oficial de descarga en bloodshet <http://www.bloodshed.net/>.

Una vez allí debe dar clic en la versión más reciente Dev-C++ 5 Beta 9.2 (4.9.9.2). La página tiene el siguiente aspecto:



En la zona de descargas debe seleccionar la primera opción dando clic en SourceForge:

Downloads

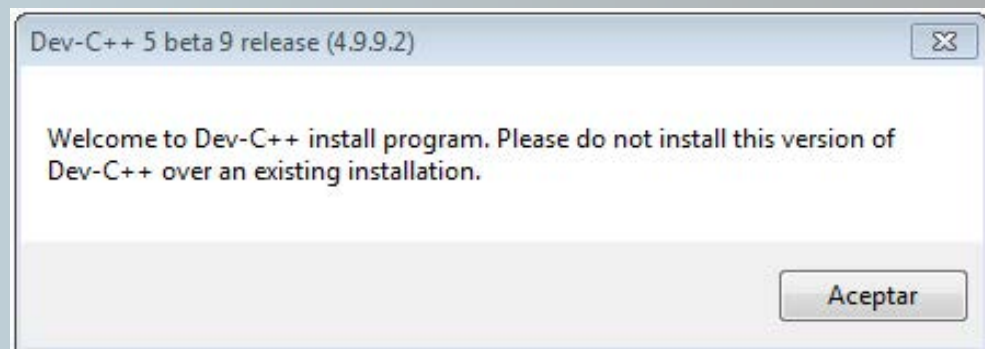


Dev-C++ 5.0 beta 9.2 (4.9.9.2) (9.0 MB) with Mingw/GCC 3.4.2
Dev-C++ version 4.9.9.2, includes full Mingw compiler system with GCC 3.4.2 and GD changes in this release.

Download from:

- [SourceForge](#)

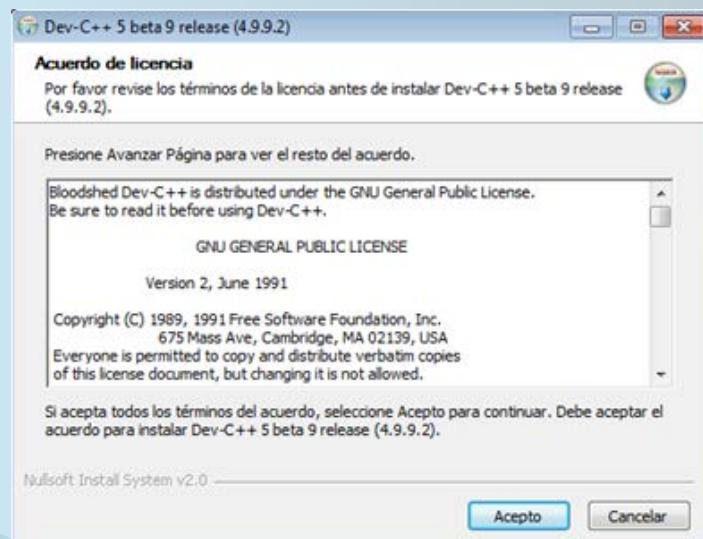
Una vez descargado el archivo debe ejecutarlo y seguir las instrucciones para la instalación. En la primer ventana de clic en Aceptar:



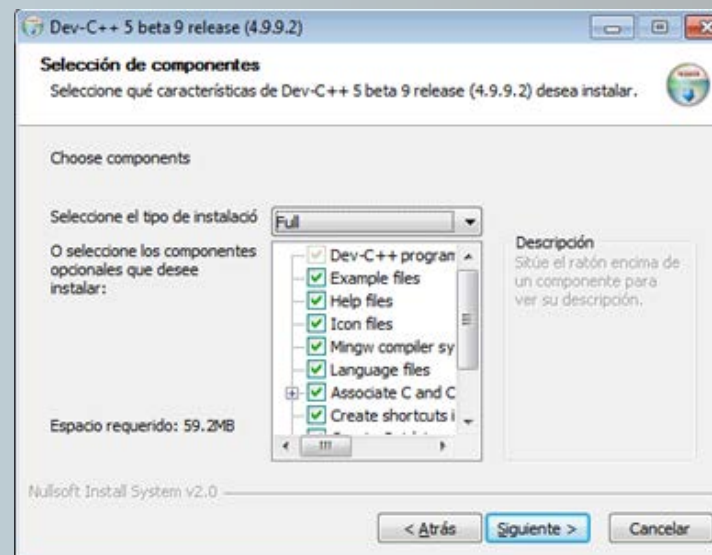
En la segunda ventana seleccione el idioma y de clic en el botón OK:



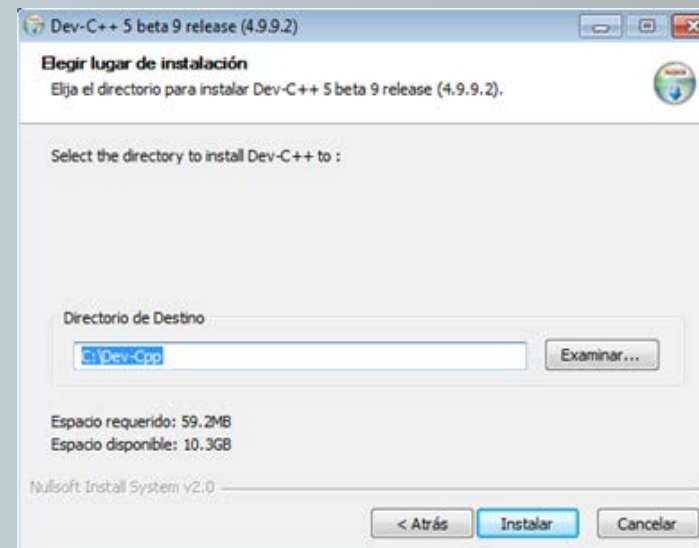
Continúe aceptando los términos del acuerdo de licencia dando clic en Acepto:



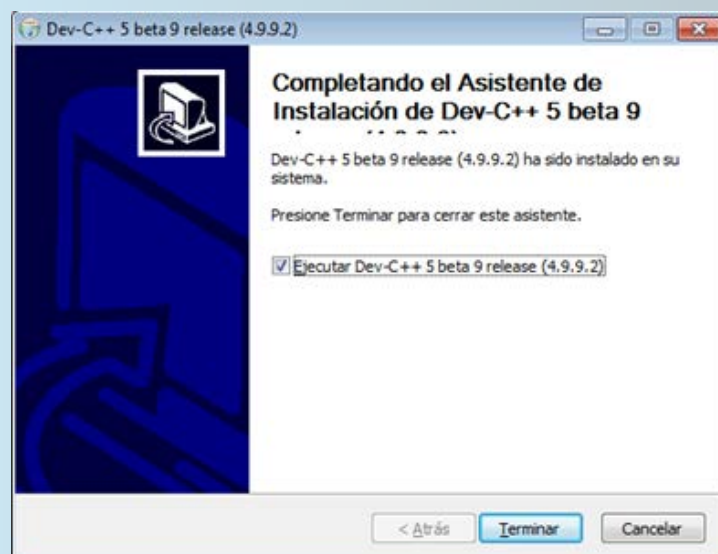
A continuación de clic en siguiente:



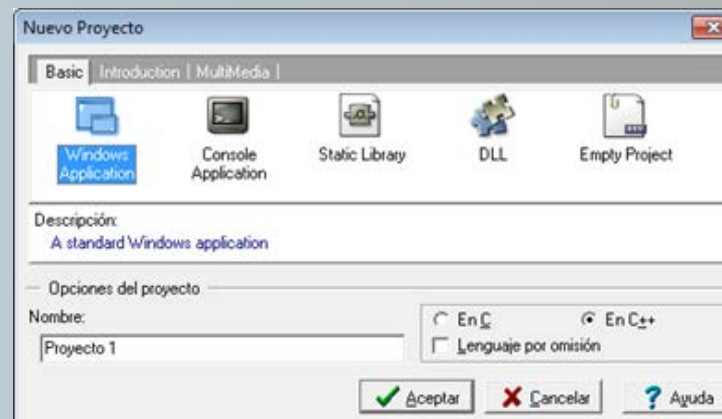
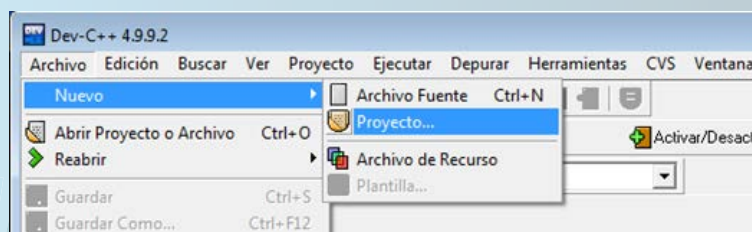
En la siguiente ventana de clic en Instalar:



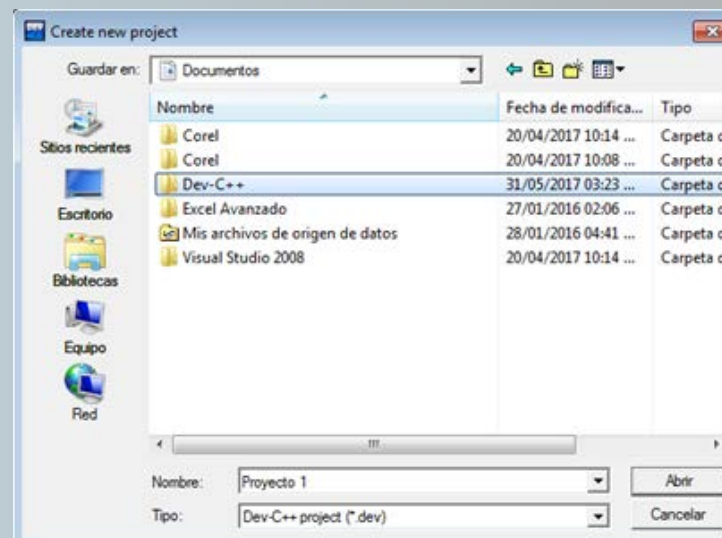
La siguiente imagen muestra la completa instalación del Entorno de Desarrollo Integrado. Para iniciar de clic en Terminar.



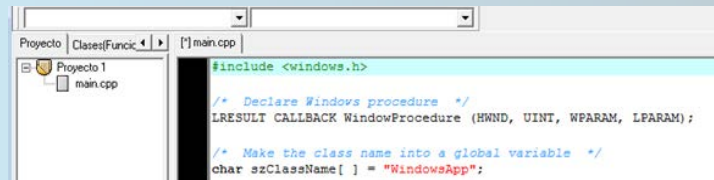
Una vez instalado, realice una prueba para verificar el correcto funcionamiento de la aplicación. Al abrir debe crear un nuevo proyecto del tipo **Windows Application**:



Abra el explorador de Windows para seleccionar la carpeta donde se va a guardar (Se recomienda crear una carpeta en “Mis Documentos” donde se almacenen todos los ejercicios que se desarrollen en este curso):



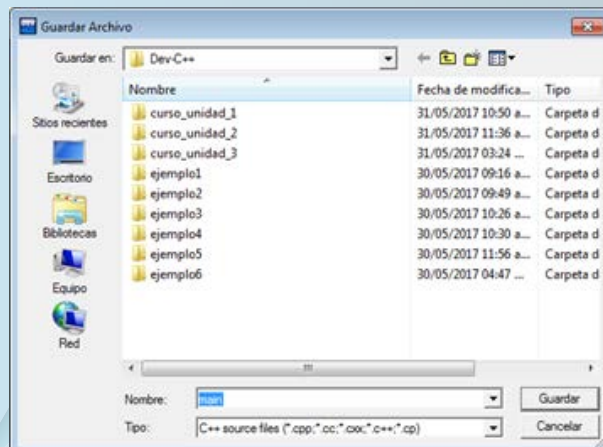
Al dar Clic en Guardar se crea un proyecto predeterminado que contiene únicamente un archivo llamado main.cpp que al ser compilado y ejecutado crea una ventana de Windows.



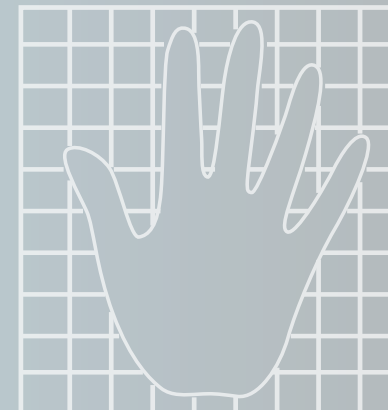
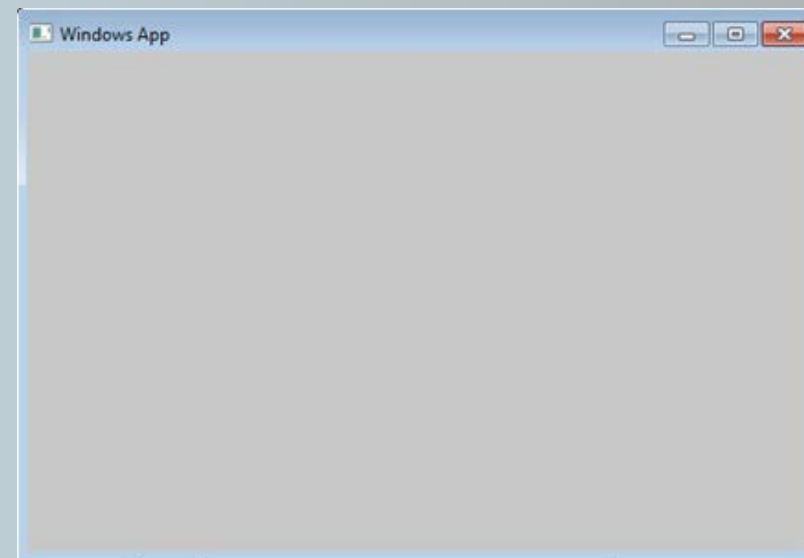
Dev C++ cuenta con una barra de herramientas compuesta por botones de uso frecuente.



Allí debe dar clic en guardar y de nuevo “guardar” en la ventana que se abre.



En la barra de herramientas también se encuentran las opciones para compilar y ejecutar proyectos. En este caso, debe dar clic en el botón con el ícono que compila y seguidamente ejecuta la aplicación. La siguiente ventana corrobora la exitosa instalación del programa:

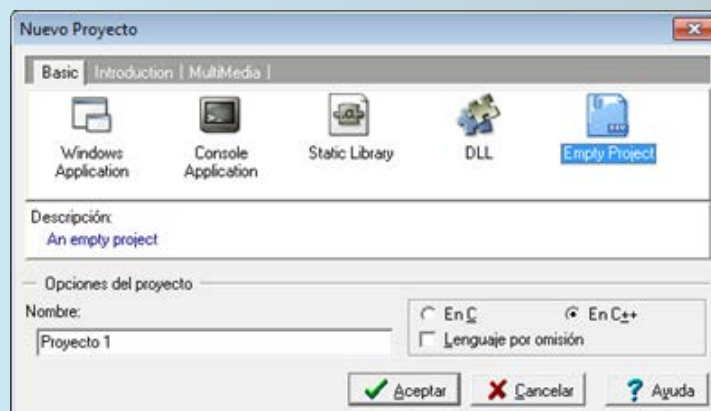


4. Componentes y tipos de datos

Los componentes básicos de un programa en C++ son: archivos de cabecera, comentarios, identificadores, palabras reservadas y signos de puntuación. Son elementos que se irán exponiendo a través de un ejemplo.

De ahora en adelante, al crear un proyecto se deben seleccionar la opción de proyecto vacío. A continuación se presentará un ejemplo y se analizarán los componentes de dicho código. Se recomienda seguir los siguientes pasos:

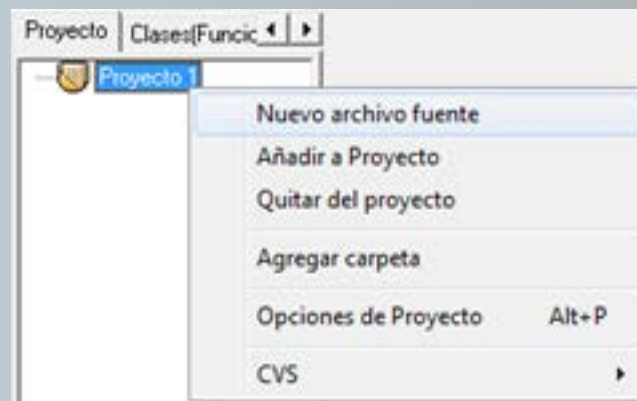
En el menú dar clic en Archivo > Proyecto y seleccionar “Empty Project”.

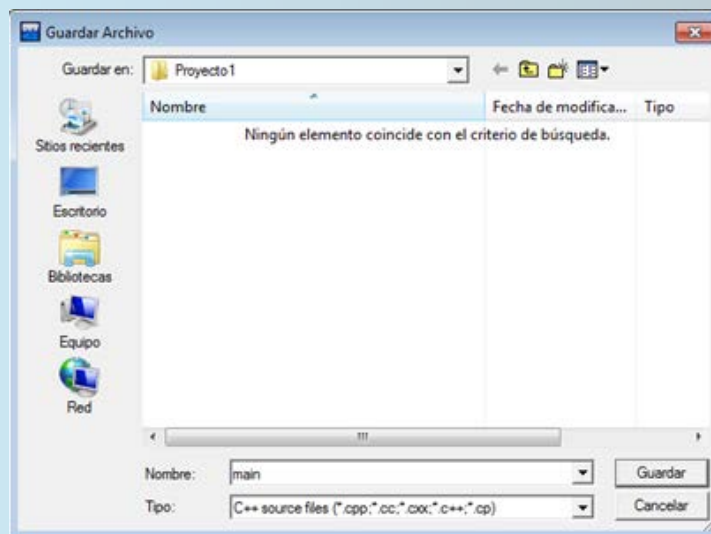


Buscar la ubicación donde desea guardar el proyecto:



En el explorador de proyectos dé clic derecho en el nombre del proyecto y seleccione “Nuevo archivo fuente” y guárdelo con el nombre de main:





Digite el siguiente código. Tenga en cuenta que encontrará caracteres como: el numeral (#), signos de mayor (>) y menor (<), punto y comas (;), paréntesis () y llaves ({ }). Signos de puntuación propios del lenguaje C++ que cumplen determinada función. Por lo tanto es imperiosa su digitación.

```
1 #include <iostream>
2
3 using namespace std;
4 //Definición de la función main
5 int main()
6 {
7     int a, b;
8     cout << "Ingresa el primer numero"<<endl;
9     cin >> a;
10    cout << "Ingresa el segundo numero"<<endl;
11    cin >> b;
12
13    cout <<"La suma de los numeros es: "<< a+b <<endl;
14    system("pause");
15    return EXIT_SUCCESS;
16 }
17
```

Al compilar y ejecutar el anterior programa, éste debe pedir al usuario que digite dos números y como resultado mostrará en pantalla la sumatoria entre ellos. Recuerde que cada vez que digite un dato debe dar enter para indicarle al programa que quiere avanzar.

En el código anterior se muestra la estructura básica de un programa en C++. Se observan varios componentes que se irán identificando a través del número de la línea, donde se brindará una explicación de cada uno de ellos.

```
1 #include <iostream>
```

En la línea 1 se encuentra la directiva del preprocesador `#include` que permite incluir en el programa una serie de archivos de cabecera que contienen funciones predefinidas que realizan tareas específicas. Por ejemplo `<iostream>` es un componente que permite al programador utilizar funciones para recibir y mostrar datos. De ahí su nombre input/output stream.

```
3 using namespace std;
```

En la línea 3 se hace uso del espacio de nombres `std`: `using namespace std`. Que permite agregar las palabras reservadas del lenguaje C++. En la línea 8 y 9 encontramos los objetos `cout` y `cin` respectivamente. Si no se hiciera uso del espacio de nombres, el compilador no reconocería estas palabras reservadas y generaría un error.

```
4 //Definición de la función main
```

En la línea cuatro se registra un comentario, el cual no interfiere en la codificación de la aplicación. Se utiliza como ayuda para el programador al momento de realizar la lectura del código.

```
5 int main()  
6 {
```

En la línea 5 se encuentra la cabecera de la función `main()`. Como se observa, está antecedita por la palabra `int` que es el tipo de la función y el valor que retornará (línea 15). El uso de esta función es indispensable en todos los programas que se desarrollen en C++. Es de uso obligatorio.

Para definir la función se abren y se cierran paréntesis vacíos y en la siguiente línea se abren y se cierran llaves. Dentro de estas llaves va el código de la aplicación.

```
7 int a, b;
```

En la línea 7 se presenta la primera declaración de variables. Se ha definido `a` y `b` como variables de tipo entero. Esto quiere decir que pueden almacenar números enteros entre el rango de -32768 y 32767 (Más adelante se profundizará en el tipo de datos adoptado por el lenguaje C++). Estos caracteres también se conocen como identificadores y no son palabras reservadas de lenguaje. Es de aclarar que la definición de variables no debe empezar por valores numéricos y el único signo de puntuación permitido es el guión bajo. Ejemplo: `nume_1`, `nombre_usuario`, `a1`, `c`, etc.

Otro detalle para observar en esta línea es que las variables están separadas por una coma (,) y al final hay un punto y coma (;) que debe ir de forma obligatoria al cierre de cada línea de código.

```
8 cout << "Ingresa el primer numero"<<endl;
```



En la línea 8 se encuentra la aplicación del objeto cout que permite mostrar en pantalla un mensaje o algún tipo de información. Éste debe ir entre signos dobles de menor (<<).

En este caso se requiere mostrar una frase solicitando al usuario que digite un número, por lo cual, ésta debe ir entre comillas dobles (" "). Al final la etiqueta endl o salto de línea permite pasar el cursor al siguiente renglón.

```
9      cin >> a;
```

En la línea 9 se hace uso del objeto cin que permite recibir lo que digite el usuario en la consola. Debe ir seguido de signos dobles de mayor (>>) y la variable que se ha dispuesto para guardar dicho valor. Al finalizar siempre el punto y coma (;).

```
10     cout << "Ingresa el segundo numero"<<endl;  
11     cin >> b;
```

En las líneas 10 y 11 se repiten las instrucciones para recibir el segundo número por parte del usuario.

```
13     cout <<"La suma de los numeros es: "<< a+b <<endl;
```

En la línea trece se utilizan de nuevo el objeto cout con el fin de mostrar el resultado. Como se observa, hay una combinación de datos de salida, como lo son, la frase entre comillas y la suma de las variables. Para concatenarlas, éstas deben ir unidas por signos dobles de menor (<<).

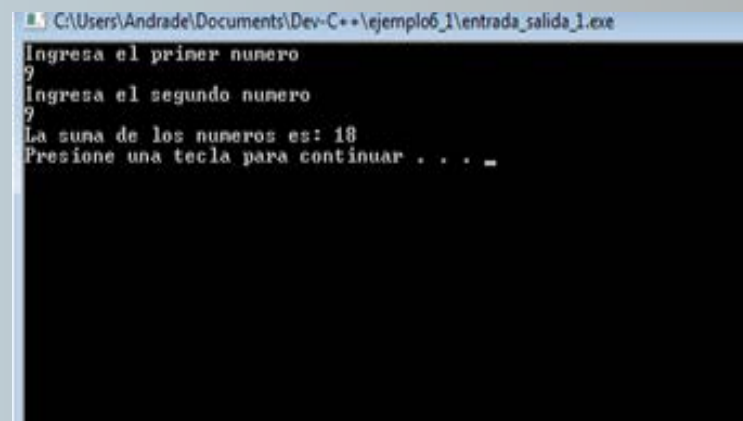
```
14     system("pause");
```

La sentencia system("pause") en la línea 14 obliga al sistema a mantener la consola abierta hasta que se presione una tecla. De lo contrario ésta se cerraría y no se alcanzaría a ver el resultado de la suma de los dos números.

```
15     return EXIT_SUCCESS;  
16 }
```

En la línea 15 la sentencia returnEXIT_SUCCESS devuelve el control al sistema operativo y confirma que la aplicación finalizó correctamente. En la línea 16 se encuentra la llave de cierre de la función main().

La salida en pantalla de la aplicación es la siguiente:



4.1. Archivos de cabecera

Como se mencionó anteriormente, un archivo de cabecera contiene objetos y funciones que pueden ser incluidos en el código del programa, dentro de los más comunes se pueden encontrar:

Tabla 1 Algunas Librerías de C++.

Librería	Descripción
iostream	Es un componente que permite al programador utilizar funciones de entrada y salida de datos.
string	Está compuesta por funciones de manipulación de cadenas de caracteres.
math	Cuenta con funciones para facilitar cálculos matemáticos.
cstdlib	Esta cabecera define varias funciones de propósito general, incluyendo gestión de memoria dinámica, generación de números aleatorios, comunicación con el entorno, aritmética entera, búsqueda, clasificación y conversión.

4.2. Comentarios

En C++ se pueden definir los comentarios de dos formas, la primera como se presentó en el ejemplo anterior, iniciando la línea con doble barra (//), de esta forma el comentario se limita a una sola línea. La segunda forma es dando inicio con barra y asterisco (/*), con la cual se pueden cubrir varias líneas; para cerrar el comentario se digita asterisco y barra (*/).

Ejemplo:

```
/*
```

```
Este es un comentario  
de dos líneas
```

```
*/
```

4.3. Identificadores

Los identificadores son los caracteres o grupo de caracteres definidos por el programador, éstos no deben ser palabras reservadas del lenguaje C++. Como se explicaba anteriormente, no pueden empezar con un número y el único signo de puntuación válido es el guión bajo.

Tabla 2 Ejemplo de Identificadores en C++.

Identificador válido	Identificador no válido
nombre	0Nombre
Apellido_uno	Apellido uno
dia_mes_ano	1er_dia
i	1_a
cargo	"cargo"

4.4. Palabras reservadas

C++ cuenta con muchas palabras reservadas, por lo cual no pueden ser usadas como indicadores porque cada una establece el llamado a determinada función o brinda unas características especiales. Son algunas palabras reservadas las siguientes:

Tabla 3 Palabras reservadas en C++.

bool	break	case	catch	char
const	default	delete	do	double
else	float	for	goto	if
inline	int	namespace	new	private
protected	public	return	short	sizeof
static	switch	this	try	typedef
void	while			

4.5. Signos de puntuación

A parte de los signos que se vieron en el ejemplo anterior, C++ cuenta con otros más que se exponen a continuación:

Tabla 4 Signos de puntuación en C++.

!	%	^	&	*	()	-	+	=	{	}	~
[]	\	;	'	:	<	>	?	,	.	/	

4.6. Tipos de datos

Los tipos de datos son aquellos que definen una variable para almacenar en memoria un dato. Los tipos de dato básicos en C++ se muestran a continuación:

Tabla 5 Tipos de datos en C++.

Tipo	Subtipo	Rango	Espacio en memoria
Entero	int	-32.768 ... +32.767	2 bytes
	unsignedint	0 ... 65.535	
	short int	-32.768 ... +32.767	
	long	-2147483648 ... 2147483647	
	unsignedlong	0 ... 4294967295	
En coma flotante	float	$3.4 \times 10^{-38} \dots 3.4 \times 10^38$	4 bytes
	double	$2.7 \times 10^{-308} \dots 2.7 \times 10^{308}$	8 bytes
	long	$3.4 \times 10^{-4932} \dots 2.1 \times 10^{4932}$	10 bytes
Caracteres	char	Los caracteres se almacenan como números enteros en el rango de -128 ... +127	1 byte
	unsignedchar	0 ... 255 para representar a todos los caracteres del código ASCII	1 byte
bool	bool	Verdadero (cualquier entero positivo) o Falso (cero)	1 byte

En el siguiente ejemplo se declaran variables del tipo carácter (char). En Dev C++ elabore un nuevo proyecto vacío y guárdelo como proyecto 2. De clic en el menú archivo - nuevo para crear un archivo fuente, guárdelo con el nombre de main y digite el siguiente código:

```
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6     char nombre_apellido[30];
7     char sexo;
8     int edad;
9
10    cout<<"Digita tu nombre y apellido..." << endl;
11    cin.getline(nombre_apellido, 30);
12    cout<<"Digita tu edad..."<<endl;
13    cin>>edad;
14    cout<<"Digita tu Sexo M o F..."<<endl;
15    cin>>sexo;
16
17    system("cls");
18
19    cout<<"Tu Nombre es: "<< nombre_apellido << endl;
20    cout<<"Tu Edad es: "<< edad <<" años" << endl;
21    cout<<"Tu Sexo es: "<< sexo << endl << endl;
22
23    system("pause");
24    return EXIT_SUCCESS;
25 }
```

Ilustración 22. Código Proyecto 2.
Fuente: Sena.

A continuación, se explicarán las líneas que son nuevas para el material de estudio:

```
6     char nombre_apellido[30];
7     char sexo;
```

En las líneas 6 y 7 se declaran dos variables del tipo carácter. Se espera que la primera almacene el nombre y el apellido del usuario, por lo que al finalizar, se agregan corchetes y se define la dimensión de la variable, que en este caso podrá almacenar hasta 30 caracteres incluyendo espacios.

La segunda variable (sexo) al ser definida sin un tamaño específico tiene la capacidad de guardar solamente un carácter (F para femenino ó M para masculino).

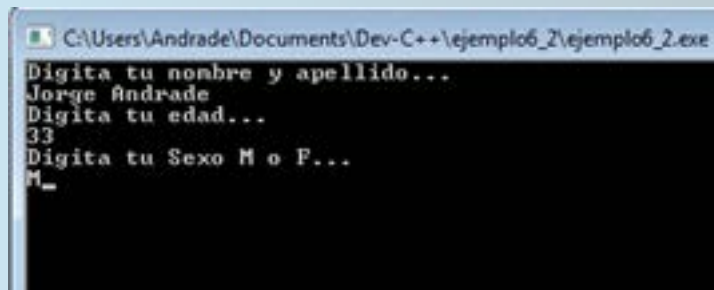
```
10    cout<<"Digita tu nombre y apellido..." << endl;
11    cin.getline(nombre_apellido, 30);
```

En la línea 11 se recibe la cadena de caracteres digitada por el usuario, la cual hace referencia al nombre y al apellido. Para recibirla de forma completa se utiliza la función getline() en compañía con el objeto cin; dado que al utilizar simplemente cin y los signos de << la variable almacenará la cadena de texto hasta que se encuentre con un espacio.

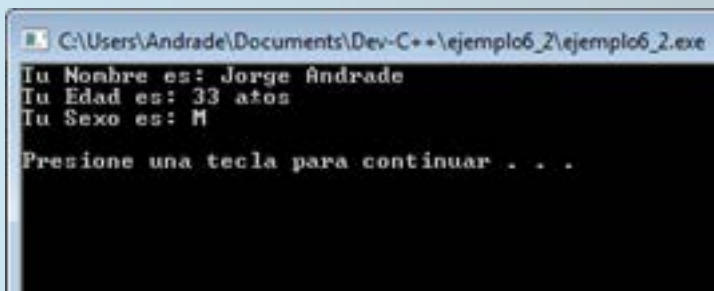
```
17    system("cls");
```

En la línea 17 se encuentra una sentencia que le indica al sistema que debe limpiar la pantalla.

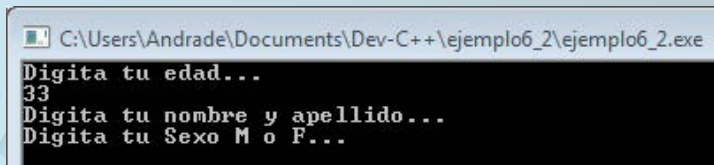
La salida en pantalla de la aplicación es la siguiente:



Después de limpiar pantalla muestra lo siguiente:



Importante: Si en el ejercicio anterior se hubiera leído primero la edad y después el nombre, el resultado en pantalla sería el siguiente:



Como se observa, no se dio espacio para que el usuario digitará el nombre y el apellido. Lo anterior ocurre porque la función `getline()` registra la tecla enter que se presionó después de digitar la edad. La tecla enter en el lenguaje C++ está representada por el signo “\n” el cual debe ser eliminado antes de usar la función `getline()` con la aplicación de la función `cin.ignore()` que requiere dos parámetros: el primero representa el número máximo de caracteres a ignorar y el segundo parámetro representa al caracter delimitador, en este caso el enter.

Por consiguiente, el código quedaría así:

```
10  cout<<"Digita tu edad..."<<endl;
11  cin>>edad;
12  cin.ignore(256, '\n');
13  cout<<"Digita tu nombre y apellido..." << endl;
14  cin.getline(nombre_apellido, 30);
15  cout<<"Digita tu Sexo M o F..."<<endl;
16  cin>>sexo;
```

5. Declaración de variables

En el ejemplo anterior se declararon las variables `a` y `b`. Esto quiere decir que se asignó una posición en memoria y se le asoció un identificador (`a` y `b`). Existen dos tipos de variables: locales y globales.



5.1. Variables locales

Las variables locales son aquellas que se definen dentro de una función, lo cual quiere decir que solamente pueden ser asignadas o modificadas dentro de ella, éstas variables son desconocidas para el exterior.

5.2. Variables globales

Las variables globales son aquellas que se definen por fuera de las funciones y por consiguiente son visibles para todas ellas; de este modo, pueden acceder para obtener su valor o para modificarlo.

6. Funciones

6.1. Función main()

La función main es el punto inicial de una aplicación en C++ y es de uso obligatorio, entre sus llaves van las sentencias de programación y desde ella se pueden invocar variables globales y otras funciones declaradas por el usuario. Ésta debe retornar un valor, no pueden existir dos o más funciones main en el mismo programa y es importante recordar que todas las líneas de código dentro de la función deben terminar en punto y coma (;). Dentro de la función main no se puede definir funciones creadas por el usuario.

6.2. Funciones definidas por el usuario

Las funciones creadas por el usuario deben estar definidas por fuera de la función main, pueden llevar el nombre que el usuario desee teniendo en cuenta que no se permite dejar espacios y no debe empezar por caracteres numéricos. En el paréntesis se definen los parámetros o variables que necesita la función para ser procesados y retornar un valor. Cuando se invoque la función se deben pasar los valores de dichos parámetros para su correcto funcionamiento.

A continuación se presentará el código modificado del primer ejemplo y se incluirá una función definida por el usuario, lo anterior para lograr una mejor comprensión:

```
1 #include <iostream>
2
3 using namespace std;
4 //Declaración de la función
5 int suma(int numero_1, int numero_2);
6 //Definición de la función main
7 int main()
8 {
9     int a, b;
10    cout << "Ingresa el primer numero"<<endl;
11    cin >> a;
12    cout << "Ingresa el segundo numero"<<endl;
13    cin >> b;
14    //Llamado a la función suma
15    cout <<"La suma de los numeros es: "<< suma(a,b) <<endl;
16    system("pause");
17    return EXIT_SUCCESS;
18 }
19 //Definición de la función con sus parámetros
20 int suma(int numero_1, int numero_2)
21 {
22     return numero_1 + numero_2;
23 }
```



Abra el ejercicio en Dev C++ y realice los cambios expuestos en el código anterior.

```
4 //Declaración de la función
5 int suma(int numero_1, int numero_2);
```

Como se observa en la línea 5, el primer paso fue declarar la función de tipo entero como si fuera una variable global, aquí mismo deben ir definidos sus parámetros. Definirla en esta sección permitirá ser invocada en cualquier parte del programa.

```
19 //Definición de la función con sus parámetros
20 int suma(int numero_1, int numero_2)
21 {
22     return numero_1 + numero_2;
23 }
```

Entre las líneas 20 y 23 se define la función con sus parámetros. Estos valores se reciben para ser procesados y se retorna el valor calculado de tipo entero.

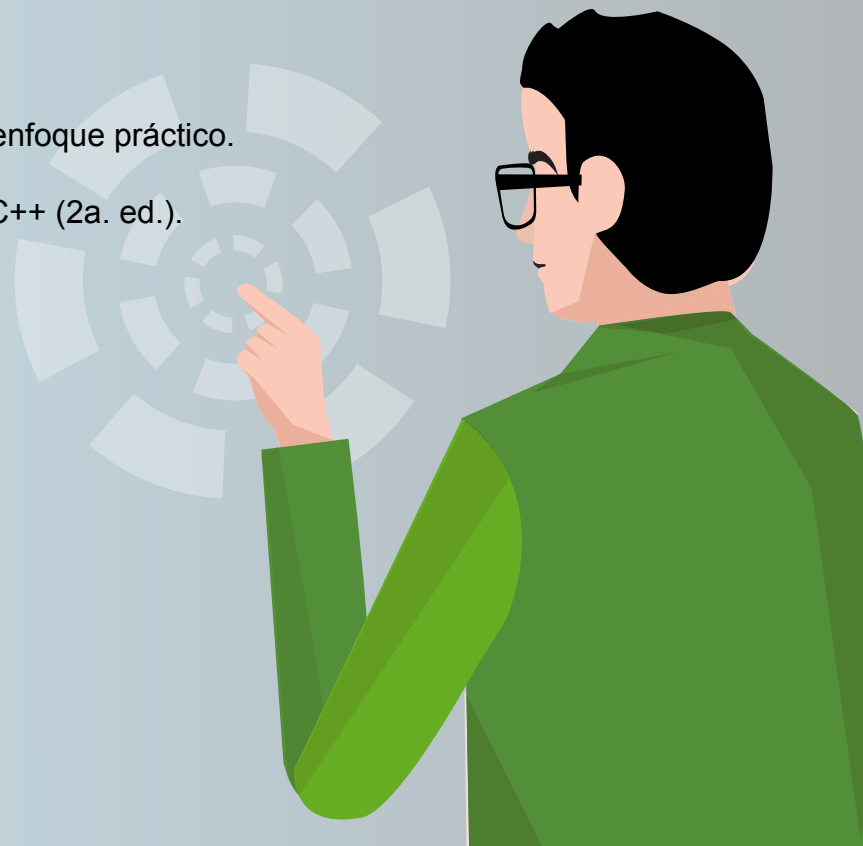
```
14 //Llamado a la función suma
15 cout <<"La suma de los numeros es: "<< suma(a,b) <<endl;
```

Finalmente, en la línea quince se invoca la función suma enviando la referencia de sus parámetros. Éstos deben ser del mismo tipo definido en la función y enviados en el mismo orden.



7. Material de apoyo

- Programación en C++: un enfoque práctico.
- Enciclopedia del lenguaje C++ (2a. ed.).





ABCD

8. Glosario

Código fuente: es un texto escrito generalmente por una persona a través de un editor que se utiliza como base para generar otro código con un compilador o intérprete para ser ejecutado por un computador.

Código objeto: es el código resultante de la compilación del código fuente, por lo general está codificado en código de máquina y distribuido en varios archivos resultantes de la compilación de cada archivo de código fuente.

Declaración: un estatuto que anuncia la existencia de una variable, función o clase, pero no la define.

Declaración global: las declaraciones globales son definiciones de variables o constantes que serán utilizadas por cualquiera de todas las funciones definidas en el programa.

Directivas: son definidas para que el compilador realice una acción antes de compilar el programa (revisar si la sintaxis es correcta y generar un código ejecutable por el computador), como la inclusión de funciones de alguna biblioteca conocida.

Directiva #include: instrucción al procesador que le indica la inclusión de un archivo de cabecera.

Función main: función que se llama en primer lugar cuando se ejecuta un programa en C++.

Sintaxis: reglas que definen cómo se forman las instrucciones de un lenguaje de programación específico.

Variable: posición de almacenamiento que puede contener diferentes valores.





9. Referencias bibliográficas

Ceballos, S. F. J. (2009). Enciclopedia del lenguaje C++ (2a. ed.). Madrid, ES: RA-MA Editorial.

Joyanes, L. Sánchez, L. (2006). Programación en C++: un enfoque práctico. España: McGraw-Hill.

Joyanes, L. Zahonero, I. (2007). Estructura de Datos en C++. España: McGraw-Hill.

10. Control del documento

	Nombre	Cargo	Versión	Fecha
Autor	Jorge Eliecer Andrade Cruz	Experto temático	1	Junio de 2017



Créditos

Equipo de Adecuación Gráfica
Centro de Comercio y servicios
SENA Regional Tolima
Línea de Producción

Director Regional

Félix Ramón Triana Gaitán

Subdirector de Centro

Álvaro Fredy Bermúdez Salazar

Coordinadora de formación profesional

Gloria Ines Urueña Montes

Experto temático

Jorge Eliecer Andrade Cruz

Senior equipo de adecuación

Claudia Rocio Varón Buitrago

Asesor pedagógico

Ricardo Palacio

Guionistas

Genny Carolina Mora Rojas
Jesús Bernardo Novoa Ortiz

Diseño y diagramación

Diana Katherine Osorio Useche
Pedro Nel Cabrera Vanegas
Ismael Enrique Cocomá Aldana

Programadores

Davison Gaitán Escobar
Héctor Horacio Morales García
Iván Darío Rivera Guzmán



Creatives commons

Atribución, no comercial, compartir igual.

Este material puede ser distribuido, copiado y exhibido por terceros si se muestra en los créditos. No se puede obtener ningún beneficio comercial y las obras derivadas tienen que estar bajo los mismos términos de licencia que el trabajo comercial.

