	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	3/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Guía práctica de estudio 01: Entorno y lenguaje de programación

---




---


***Elaborado por:***

M.C. M. Angélica Nakayama C.

Ing. Jorge A. Solano Gálvez

***Autorizado por:***

M.C. Alejandro Velázquez Mena

	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	4/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Guía práctica de estudio 01: Entorno y lenguaje de programación

### Objetivo:

Identificar y probar el entorno de ejecución y el lenguaje de programación orientado a objetos a utilizar durante el curso.

### Actividades:

- Probar los conceptos básicos del entorno y el lenguaje.
- Revisar la instalación y configuración del entorno de ejecución.
- Realizar un programa en el lenguaje de programación a utilizar usando el entorno de ejecución, utilizando la sintaxis básica (notación, palabras reservadas, comentarios, etc.)


### Introducción

Para una mejor comprensión de la **programación orientada a objetos**, se deben practicar los conceptos aprendidos en la teoría implementándolos en algún lenguaje de programación.

Lo primero que se debe hacer para comenzar a programar en un lenguaje es conocer sus **fundamentos** y el **entorno de ejecución**, así como también las **herramientas** útiles con las que se cuenta para optimizar el desarrollo de programas.

Una vez que se han comprendido las bases del lenguaje, entorno y herramientas, se puede proceder a crear un programa sencillo, realizando todos los pasos necesarios desde la codificación en el lenguaje hasta la ejecución del mismo en el entorno.

**Nota:** Elegir un lenguaje para aprender programación orientada a objetos es un tema de discusión entre programadores profesionales, ya que no existe un criterio unánime respecto a qué lenguaje es el ideal para aprender todos los conceptos necesarios. En estas

	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	5/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			


guías se tomará como caso de estudio el lenguaje de programación Java, sin embargo, queda a criterio del profesor el uso de éste u otro lenguaje orientado a objetos.

## El entorno de ejecución

**La tecnología Java** es un lenguaje de programación y una plataforma comercializada por primera vez en 1995 por Sun Microsystems.

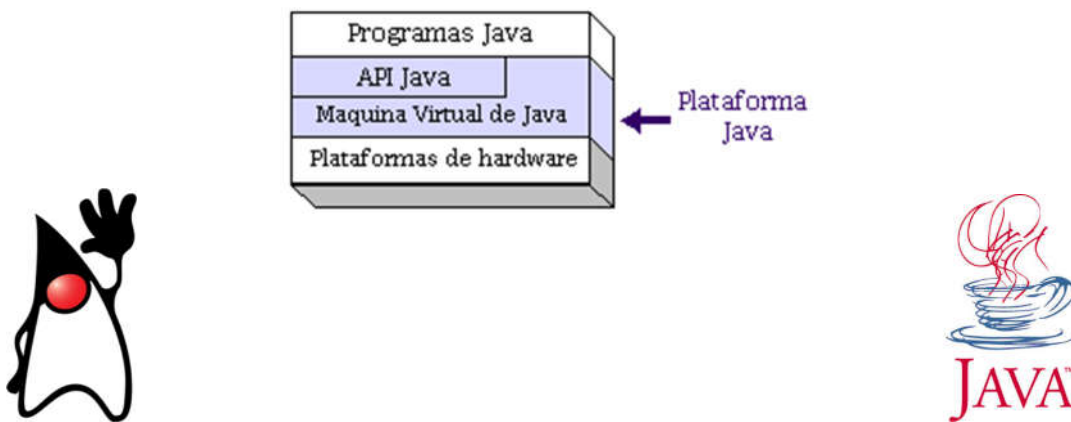
**El lenguaje de programación Java** fue originalmente desarrollado por James Gosling de Sun Microsystems (compañía que fue adquirida por Oracle) y publicado en 1995 como un componente fundamental de la plataforma Java de Sun Microsystems. Su sintaxis deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. El lenguaje de programación Java es un lenguaje de alto nivel, orientado a objetos. El lenguaje es inusual porque los programas Java son tanto compilados como interpretados.

**La plataforma Java** es una plataforma de software que se ejecuta sobre la base de varias plataformas de hardware. Está compuesto por la JVM (**Java Virtual Machine**) y la Interfaz de Programación de Aplicaciones Java o API (un amplio conjunto de componentes de software listos para usar, que facilitan el desarrollo y despliegue de aplicaciones).


	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	6/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Además de la API, toda implementación completa de la plataforma Java incluye:

- Herramientas de desarrollo para compilar, ejecutar, supervisar, depurar y documentar aplicaciones.
- Mecanismos estándar para desplegar aplicaciones para los usuarios.
- Kits de herramientas de interfaz de usuario que permiten crear interfaces gráficas de usuario (GUIs) sofisticadas.
- Bibliotecas de integración que permiten que los programas accedan a bases de datos y manipulen objetos remotos.



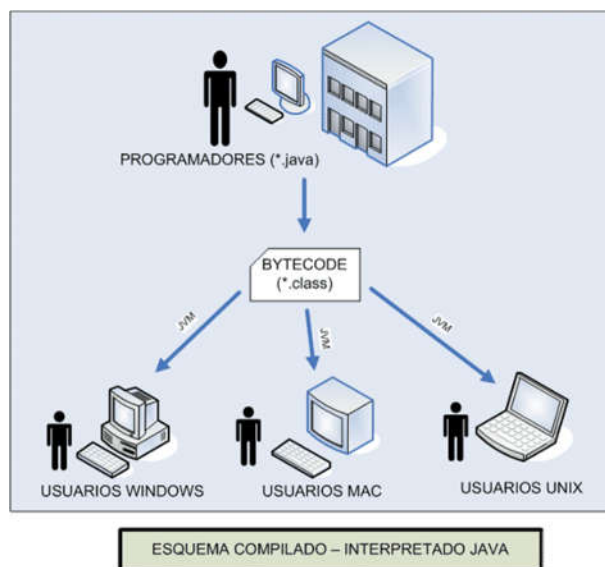


	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	8/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Todo programa en Java está organizado en clases, éstas se codifican en archivos de texto con extensión **.java**. Cada archivo de código fuente **.java** puede contener una o varias clases, aunque lo normal es que haya un archivo por clase.

Cuando se compila un archivo **.java** se genera uno o varios archivos **.class** de código binario (*bytecodes*) que son independientes de la arquitectura. Esta independencia supone que los *bytecodes* no pueden ser ejecutados directamente por ningún sistema operativo.


Durante la fase de ejecución es cuando los archivos **.class** se someten a un proceso de interpretación, consistente en traducir los *bytecodes* a código ejecutable por el sistema operativo. Esta operación es realizada por la **JVM**.

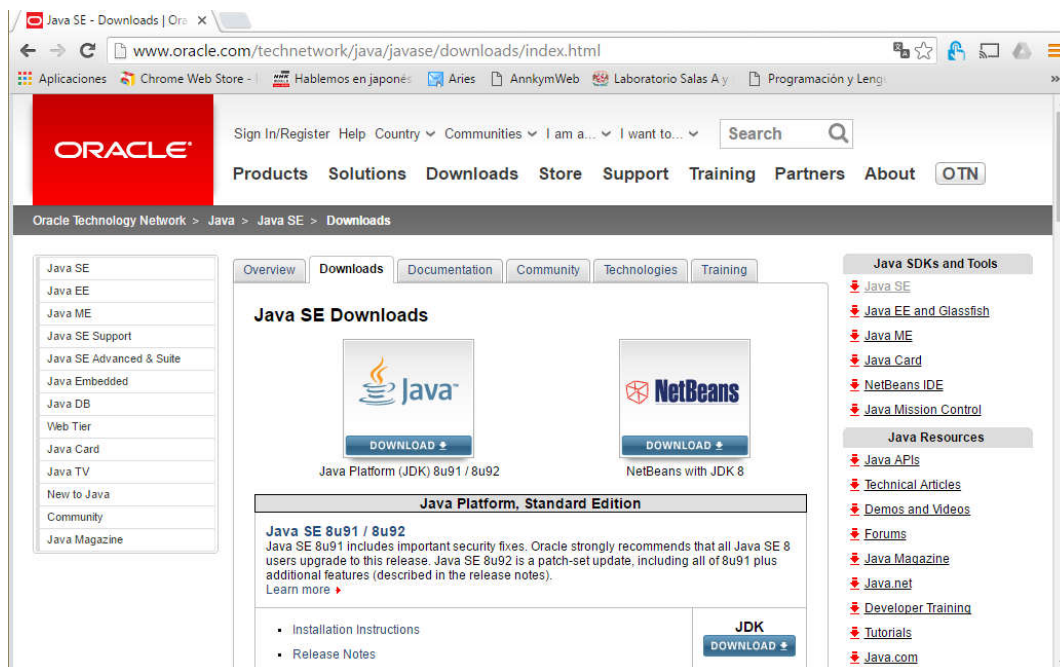


## Herramientas de desarrollo (JDK)

El **Java Development Kit (JDK)** proporciona el conjunto de herramientas básico para el desarrollo de aplicaciones con Java estándar. Se puede obtener de manera gratuita en internet, descargándola desde el sitio de Oracle.

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>


	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	9/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

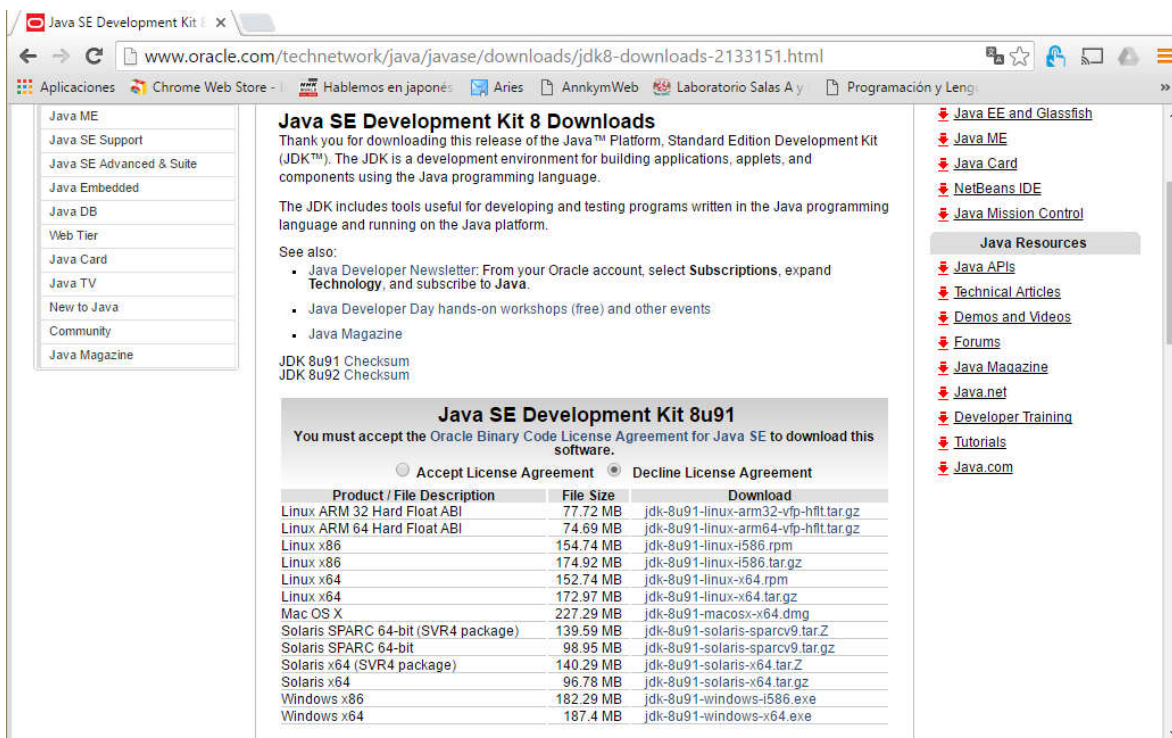


En este sitio se pueden encontrar varios recursos relacionados con java, así como otras versiones y herramientas útiles.

Para este curso se utilizará la versión 8 de Java SE (Standard Edition).



	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	10/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			



**Java SE Development Kit 8 Downloads**

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications, applets, and components using the Java programming language.

The JDK includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform.

See also:

- Java Developer Newsletter: From your Oracle account, select **Subscriptions**, expand **Technology**, and subscribe to **Java**.
- Java Developer Day hands-on workshops (free) and other events
- Java Magazine

JDK 8u91 Checksum  
JDK 8u92 Checksum

**Java SE Development Kit 8u91**

You must accept the **Oracle Binary Code License Agreement for Java SE** to download this software.

☐ Accept License Agreement ☒ Decline License Agreement


Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.72 MB	jdk-8u91-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.69 MB	jdk-8u91-linux-arm64-vfp-hflt.tar.gz
Linux x86	154.74 MB	jdk-8u91-linux-i586.rpm
Linux x86	174.92 MB	jdk-8u91-linux-i586.tar.gz
Linux x64	152.74 MB	jdk-8u91-linux-x64.rpm
Linux x64	172.97 MB	jdk-8u91-linux-x64.tar.gz
Mac OS X	227.29 MB	jdk-8u91-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.59 MB	jdk-8u91-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	98.95 MB	jdk-8u91-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140.29 MB	jdk-8u91-solaris-x64.tar.Z
Solaris x64	96.78 MB	jdk-8u91-solaris-x64.tar.gz
Windows x86	182.29 MB	jdk-8u91-windows-i586.exe
Windows x64	187.4 MB	jdk-8u91-windows-x64.exe

Para obtener la descarga correcta se debe seleccionar el sistema operativo en donde se instalará y aceptar la licencia.

Las instrucciones de instalación en distintos sistemas operativos se encuentran disponibles en el sitio:

[http://docs.oracle.com/javase/8/docs/technotes/guides/install/install\\_overview.html](http://docs.oracle.com/javase/8/docs/technotes/guides/install/install_overview.html)



	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	11/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Programa en JAVA

Aunque aún no se tiene conocimiento del lenguaje, se puede crear un sencillo **programa en Java** que imprima un saludo. Este programa tiene como fin conocer el procedimiento general que se debe seguir para crear, compilar y ejecutar programas Java.

### Codificación


Utilizando cualquier **editor de texto** disponible en el sistema (Block de notas, notepad++, gedit, vi, etc.) se captura el siguiente código (teniendo en cuenta que Java es *case sensitive*, es decir, sensible a mayúsculas y minúsculas):

```
public class HolaMundo {
    public static void main(String[] args) {
        System.out.println("Hola Mundo");
    }
}
```

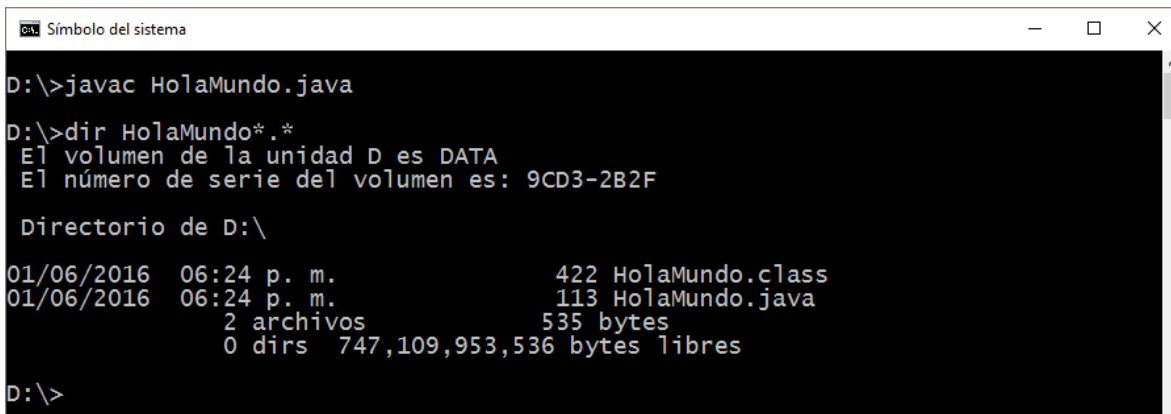
Después se guardará este programa en un archivo de texto llamado **HolaMundo.java** (el nombre del archivo debe ser el mismo que el de la clase)

### Compilación

La compilación de un archivo de código fuente **.java** se realiza a través del comando **javac** del **JDK**. Si se ha instalado y configurado correctamente el **JDK**, entonces **javac** podrá ser invocado desde el directorio en el que se encuentre el archivo *HolaMundo.java* creado.

	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	12/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Tras ejecutar este comando, se generará un archivo **HolaMundo.class**.



```

D:\>javac HolaMundo.java

D:\>dir HolaMundo*.*
El volumen de la unidad D es DATA
El número de serie del volumen es: 9CD3-2B2F

Directorio de D:\

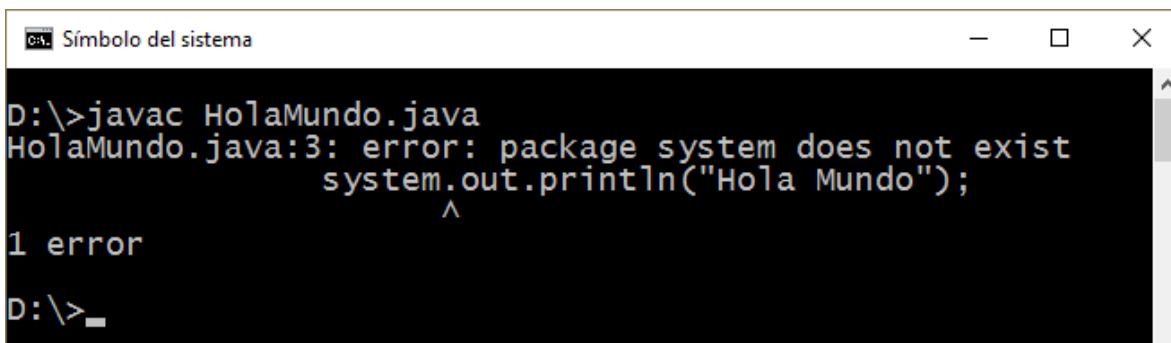
01/06/2016  06:24 p. m.          422 HolaMundo.class
01/06/2016  06:24 p. m.          113 HolaMundo.java
                2 archivos          535 bytes
                0 dirs  747,109,953,536 bytes libres

D:\>

```

En caso de que existieran errores sintácticos en el código fuente, el compilador informará de ello en la ventana de comandos y el archivo **.class** no se generaría.

Por ejemplo, si en el código fuente escribimos **system** en vez de **System**, al intentar compilar se tendría la salida:




```

D:\>javac HolaMundo.java
HolaMundo.java:3: error: package system does not exist
    system.out.println("Hola Mundo");
    ^
1 error

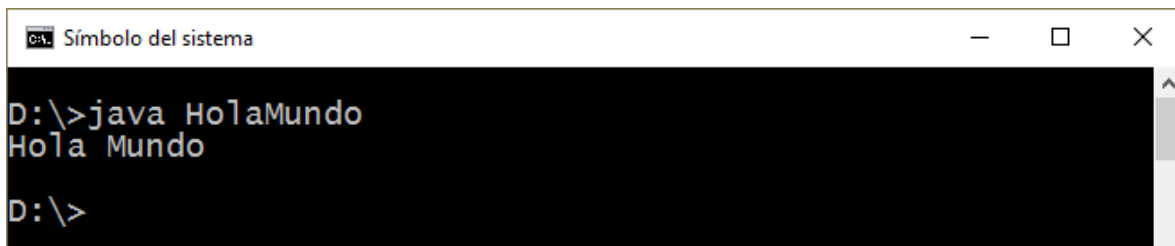
D:\>_

```

	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	13/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

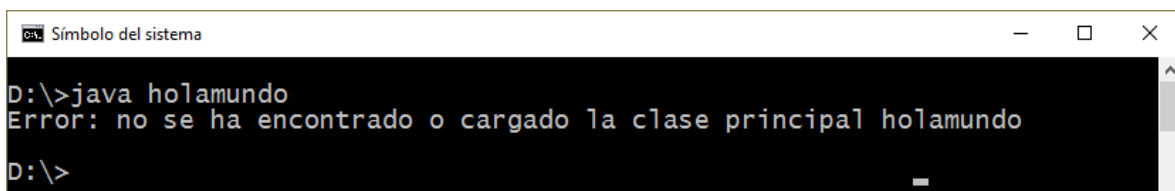
## Ejecución

Para ejecutar el programa (una vez compilado correctamente), se utiliza el comando **java** seguido del nombre de la clase que contiene el método *main()*. En este caso será **HolaMundo** ya que es la única clase existente.



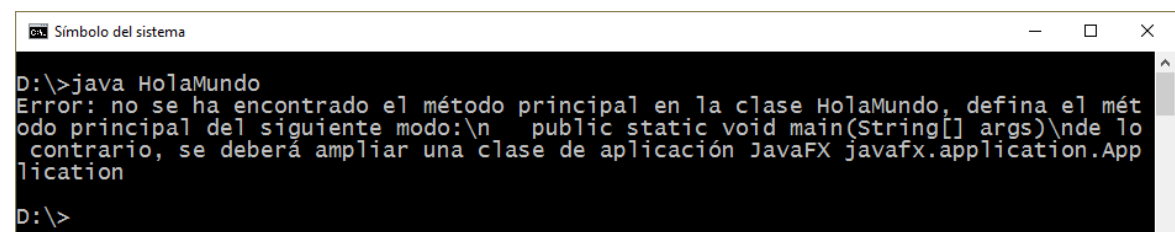
```
D:\>java HolaMundo
Hola Mundo
D:\>
```

La llamada al comando **java** insta a la máquina virtual a buscar en la clase indicada un método llamado *main()* y procede a su ejecución. En caso que **java** no encuentre la clase ya sea porque el **JDK** esté mal configurado o porque el nombre de la clase sea incorrecto, se producirá un error como el siguiente:




```
D:\>java holamundo
Error: no se ha encontrado o cargado la clase principal holamundo
D:\>
```

Si el problema no es la configuración ni el nombre de la clase, si no el formato del método *main()* no es correcto, el programa compilará correctamente pero se producirá un error al ejecutar el programa con java.



```
D:\>java HolaMundo
Error: no se ha encontrado el método principal en la clase HolaMundo, defina el método principal del siguiente modo:\n public static void main(String[] args)\nde lo contrario, se deberá ampliar una clase de aplicación JavaFX javafx.application.Application
D:\>
```

Este procedimiento para compilar y ejecutar la clase *HolaMundo* es el mismo para **cualquier programa en Java**.


	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	14/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

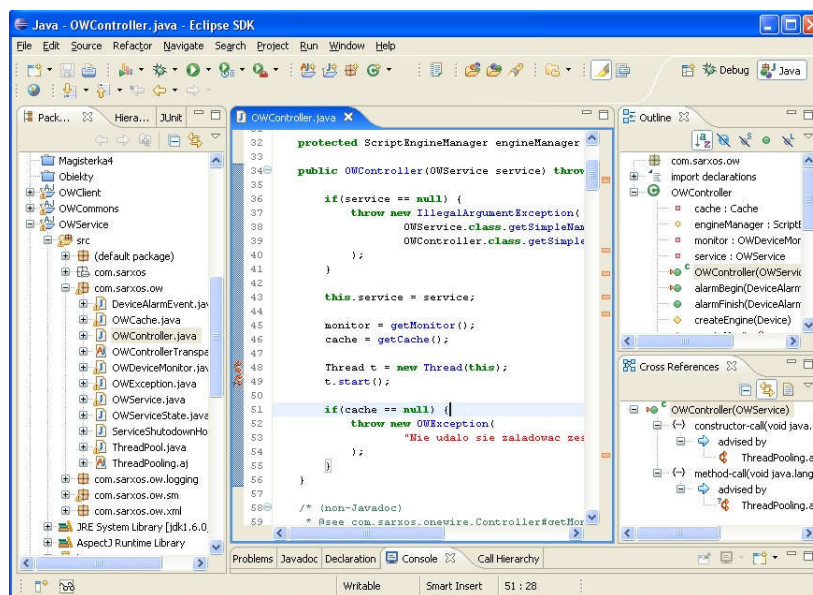
## Entorno de desarrollo integrado (IDE)

Para desarrollar un producto de software solo es necesario un editor de texto plano para capturar el código fuente y el compilador o el intérprete (según sea el caso) para transformar el lenguaje de alto nivel a lenguaje máquina. Sin embargo, también se puede hacer uso de una aplicación que contenga todas las herramientas en una interfaz, a este tipo de aplicaciones se les conoce como entorno de desarrollo integrado.

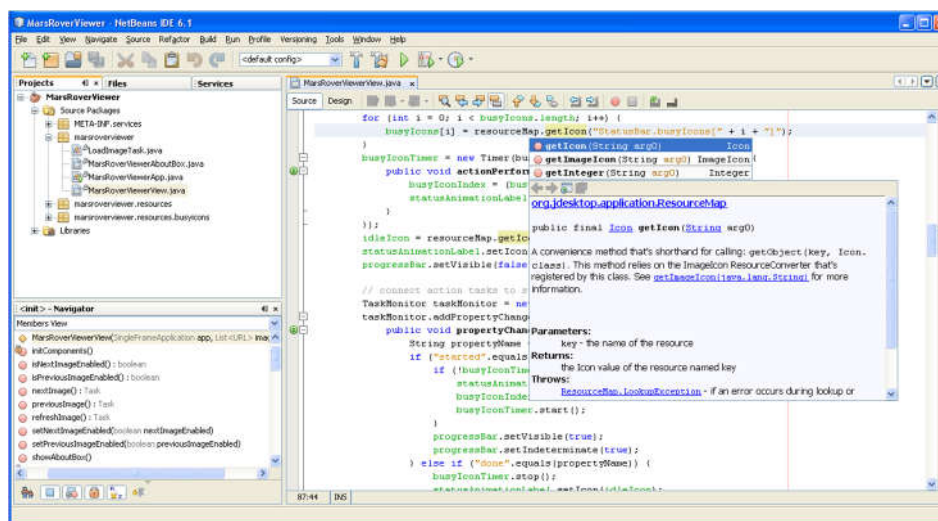
Un **entorno de desarrollo integrado** o **IDE (Integrated Development Environment)** es una aplicación que facilita el desarrollo de aplicaciones en algún lenguaje de programación. De manera general, un **IDE** es una interfaz gráfica de usuario diseñada para ayudar a los desarrolladores a construir aplicaciones de software proporcionando todas las herramientas necesarias para la codificación, compilación, depuración y ejecución.


Los **IDE** para Java utilizan internamente las herramientas básicas del **JDK** en la realización de estas operaciones, sin embargo, el programador no tendrá que hacer uso de la consola para ejecutar estos comandos, dado que el entorno ofrece una forma alternativa de utilización, basada en menús y barras de herramientas.

	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	15/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			




La escritura de código también resulta una tarea sencilla con un **IDE** ya que éstos suelen contar con un editor de código que resalta las palabras reservadas del lenguaje para distinguirlas del resto del código. Algunos incluso permiten autoescritura de instrucciones utilizando la técnica *Intellisense*, que consiste en mostrar la lista completa de métodos de un objeto según se escribe la referencia al mismo.



	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	16/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

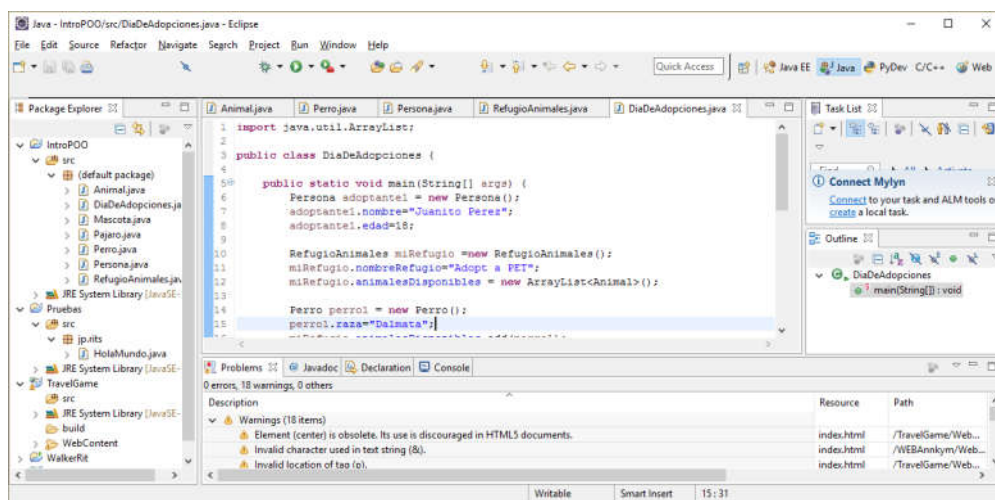
En el mercado existen diversos tipos de **IDE**, cada uno con características propias, empero, una constante es que permiten manejar las etapas para generar un programa dependiendo del tipo de lenguaje utilizado. La mecánica de utilización de estos programas es muy similar, todos ellos se basan en el concepto de proyecto como conjunto de clases que forman una aplicación.



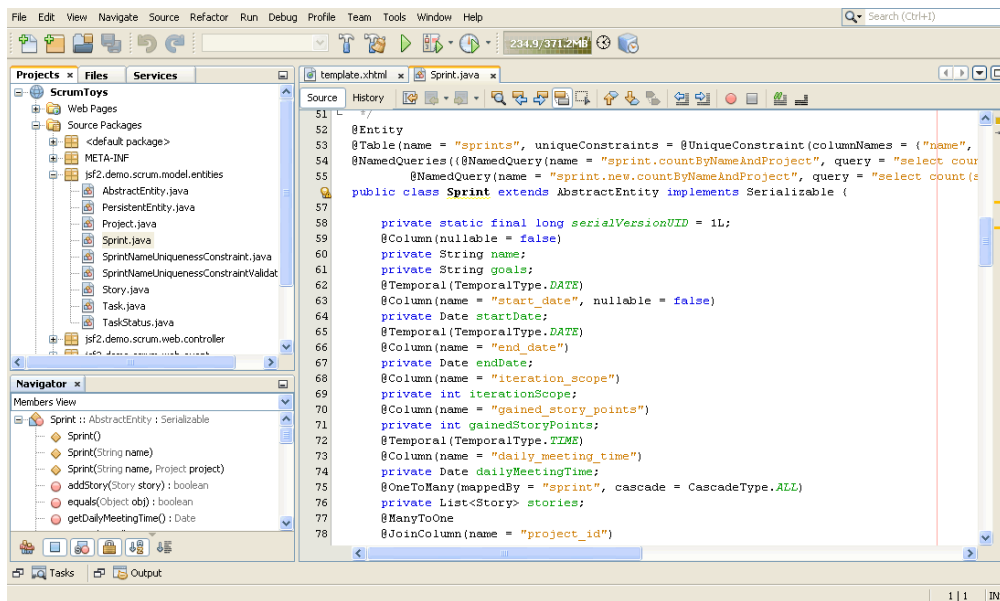
	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	17/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			


Algunos de los IDE más populares para Java son:

## Eclipse

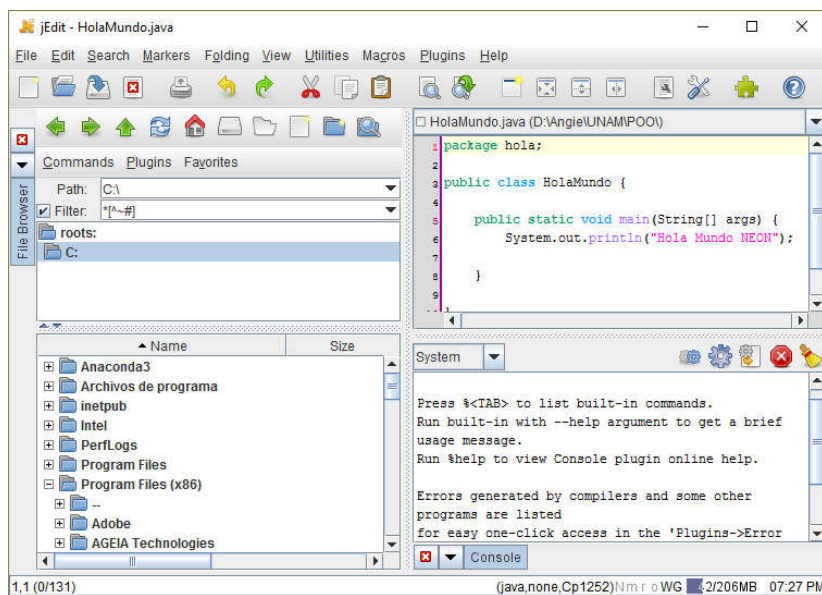


## NetBeans

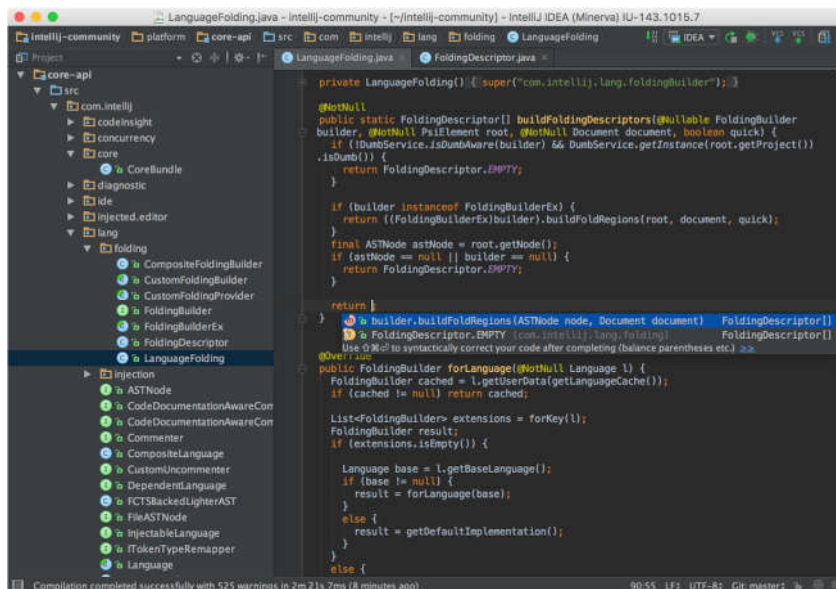



	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	18/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## jEdit

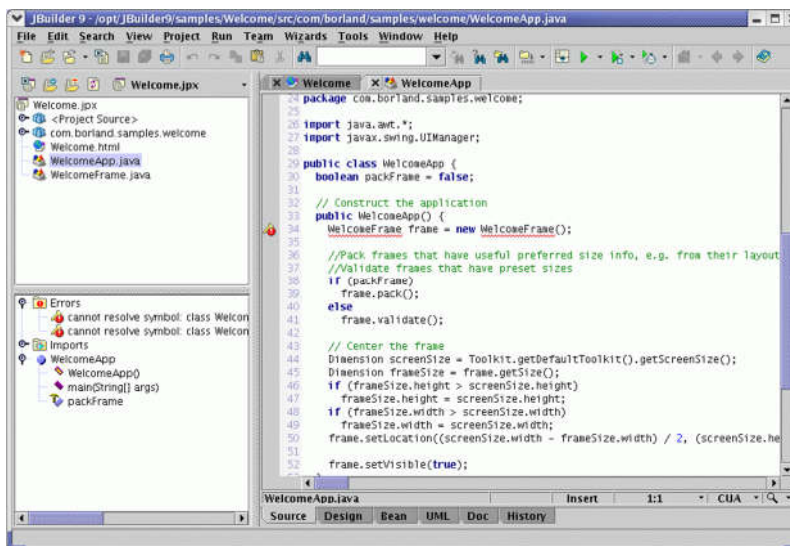


## IntelliJ IDEA

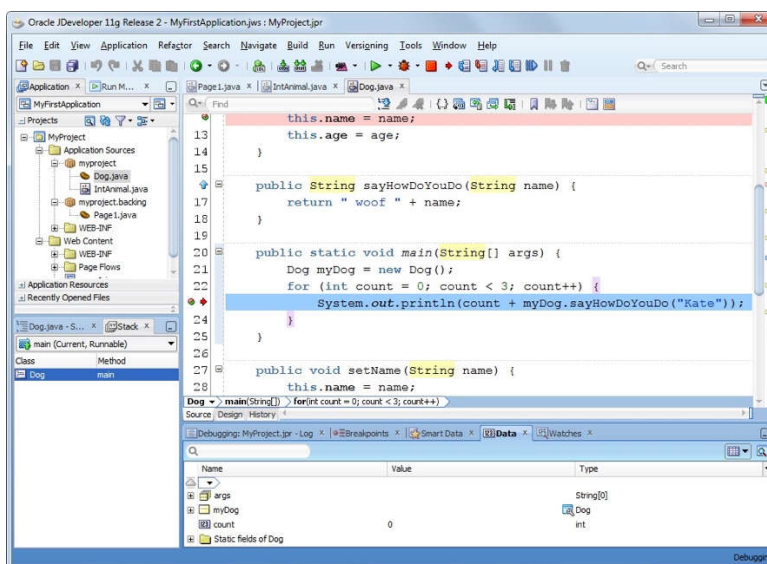


	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	19/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			


## JBUILDER



## JDeveloper



Queda a juicio del programador elegir si utiliza un IDE o no y en caso de hacerlo, también decidir cuál de acuerdo a sus necesidades y gustos.

	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	20/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Estructura general de un programa Java

Una de las principales características de Java es que es un lenguaje totalmente orientado a objetos. Como tal, todo programa Java debe estar escrito en una o varias **clases**, dentro de las cuales se podrá hacer uso del amplio conjunto de paquetes y clases prediseñadas.

Un programa en Java consta de una **clase** principal (que contiene el método **main**) y algunas clases de usuario (las específicas de la aplicación que se está desarrollando) que son utilizadas por el programa o clase principal.

La clase principal debe ser declarada con el modificador de acceso **public** y la palabra reservada **class**, seguida del nombre de la clase iniciando con mayúscula.


Un archivo fuente (\*.java) puede contener más de una clase, pero sólo una puede ser **public**. El nombre del archivo fuente debe **coincidir** con el de la clase **public** (con la extensión \*.java).

Ejemplo:

```
public class MiClase {
    public static void main(String[] args) {
        System.out.println("Esta es mi clase");
    }
}
```

El nombre del archivo tiene que ser **exactamente el mismo** que el de la clase, en este caso debe ser **MiClase.java**. Es importante que coincidan mayúsculas y minúsculas ya que *MiClase.java* y *miclase.java* serían clases diferentes para Java.

Normalmente, una aplicación está constituida por varios archivos \*.class. Cada clase realiza funciones particulares, permitiendo construir las aplicaciones con gran modularidad e independencia entre clases. La aplicación se ejecuta por medio del nombre de la clase que contiene el método **main()** (sin la extensión \*.class), invocando a la máquina virtual. Para el ejemplo:

	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	21/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			



El método **main**

En la clase principal debe existir una **función o método** estático llamado **main** cuyo formato debe ser:


```
public static void main(String[] args)
```

El método **main** debe cumplir las siguientes características:

- Debe ser un método público (**public**).
- Debe ser un método estático (**static**).
- No puede devolver ningún resultado (tipo de devolución **void**).
- Debe declarar un arreglo de cadenas de caracteres en la lista de parámetros o un número variable de argumentos).

El método **main** es el punto de arranque de un programa Java, cuando se invoca al comando **java** desde la línea de comandos, la **JVM** busca en la clase indicada un método estático llamado **main** con el formato indicado.

Dentro del código de **main** pueden crearse objetos de otras clases e invocar sus métodos. En general, se puede incluir cualquier tipo de lógica que respete la sintaxis y estructura de java.

	<b>Manual de prácticas del Laboratorio de Modelos de programación orientada a objetos</b>	Código:	MADO-21
		Versión:	02
		Página	22/166
		Sección ISO	8.3
		Fecha de emisión	14 de junio de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Bibliografía

*Martín, Antonio*

***Programador Certificado Java 2.***

*Segunda Edición.*

*México*

*Alfaomega Grupo Editor, 2008*

*Sierra Katy, Bates Bert*

***SCJP Sun Certified Programmer for Java 6 Study Guide***

*Mc Graw Hill*

*Dean John, Dean Raymond.*

***Introducción a la programación con Java***

*Primera Edición.*

*México*

*Mc Graw Hill, 2009*