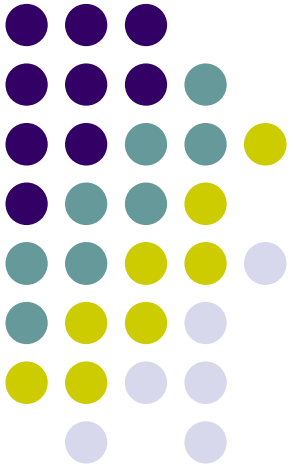
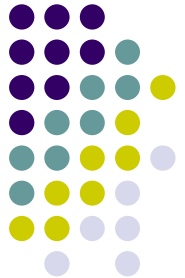
Java



Primeros pasos

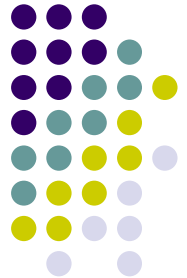
Lic. Claudio Zamoszczyk
claudio@honou.com.ar

Contenidos



- ¿Qué es Java?
- ¿Por qué Java? Características
- La maquina virtual
- Primeros Pasos: Hola Mundo!!!
- Sintaxis básica del lenguaje
- Programación Orientación a Objetos
- Trabajando con Objetos

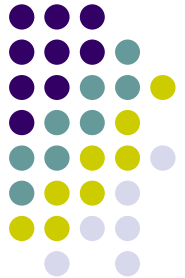
¿Qué es JAVA?



- **Java es un lenguaje de programación orientado a objetos desarrollado por James Gosling y sus compañeros de Sun Microsystems al inicio de la década de 1990. Su objetivo primario era crear un lenguaje para programar electrodomésticos interactivos.**

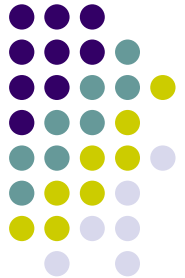
- **A diferencia de los lenguajes de programación convencionales, que generalmente están diseñados para ser compilados a código nativo, Java es compilado en un bytecode que es ejecutado por una máquina virtual (Código interpretado y precompilado).**

¿Por qué JAVA?



- Es independiente de la plataforma (Hardware - Software)
- Sintaxis similar a C/C++
- Es Orientado por Objetos
- Fácil de aprender
- Seguro
- Robusto
- No se necesita liberar memoria (Recolector de basura)
- Fuertemente tipado
- Útil para desarrollar cualquier tipo de aplicación (Web, Escritorio, etc.)
- Gran soporte por parte de las empresas (IBM, Oracle, Sun, etc.) y por la comunidad Open Source

¿Por qué JAVA? Plataformas



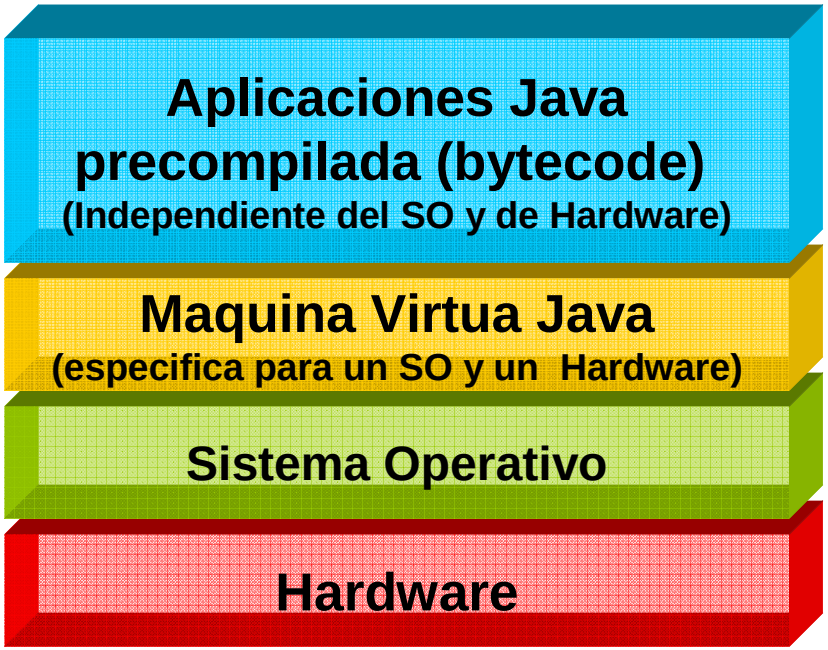
- **Java ME (Java Platform, Micro Edition) o J2ME — orientada a entornos de limitados recursos, como teléfonos móviles, PDAs (Personal Digital Assistant), etc.**

- **Java SE (Java Platform, Standard Edition) o J2SE — para entornos de gama media y estaciones de trabajo (PC de escritorio).**

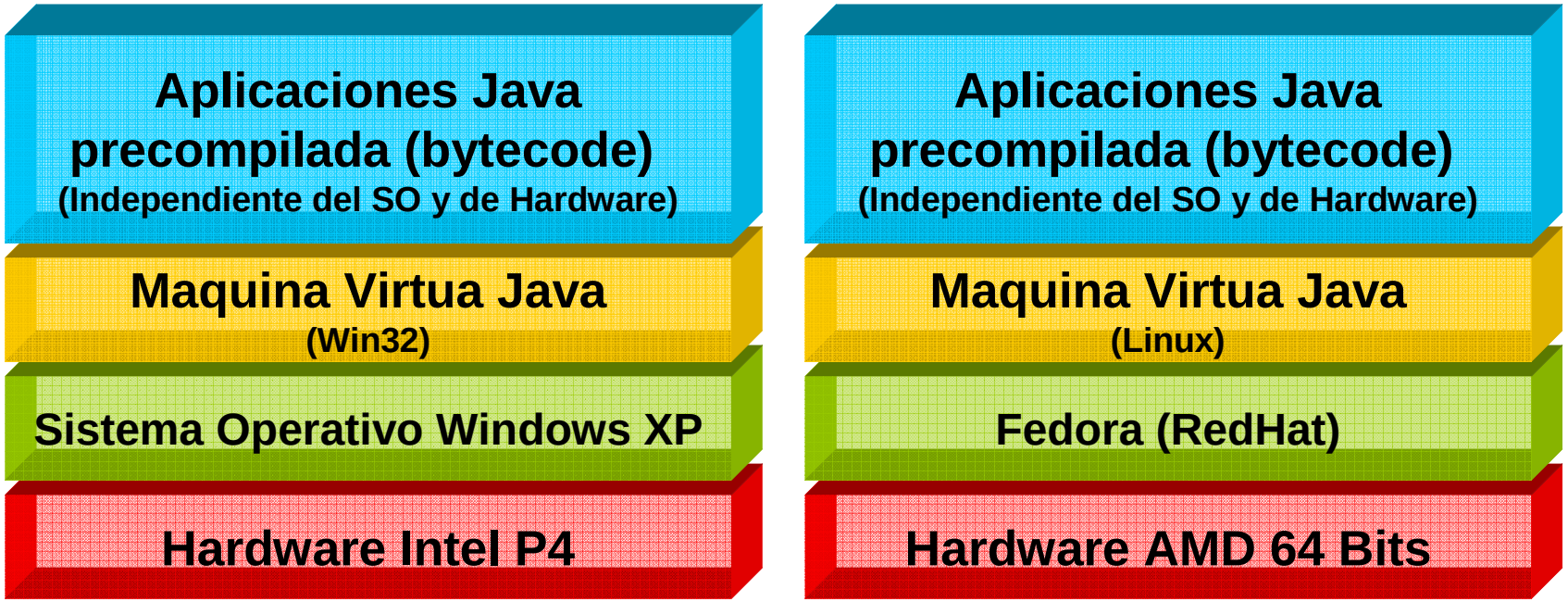
- **Java EE (Java Platform, Enterprise Edition) o J2EE — orientada a entornos distribuidos empresariales o de Internet.**

Maquina virtual

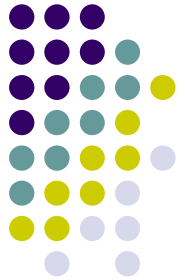
- **La JVM es un componente crucial de la plataforma Java. “Write once, run anywhere”**

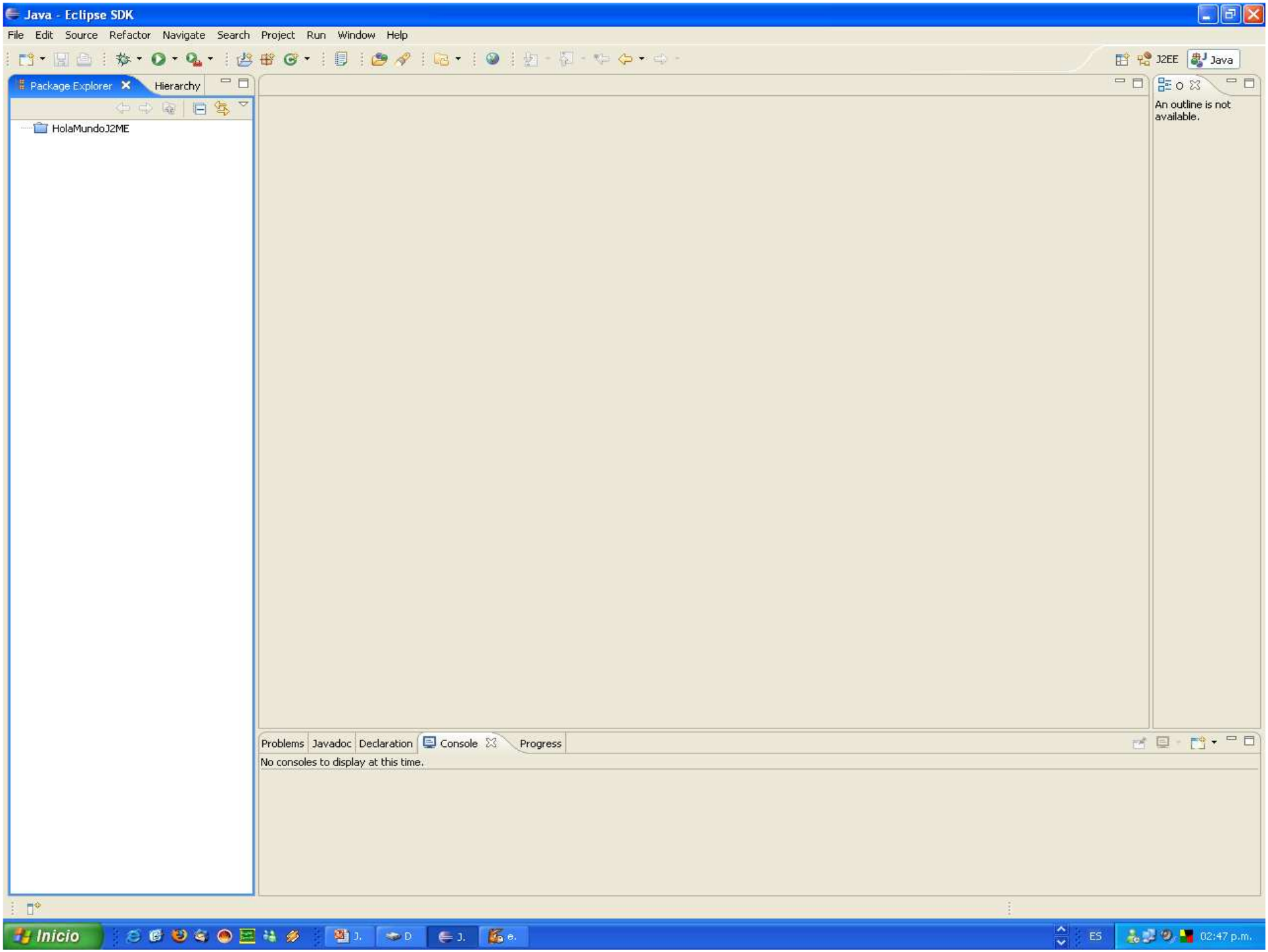


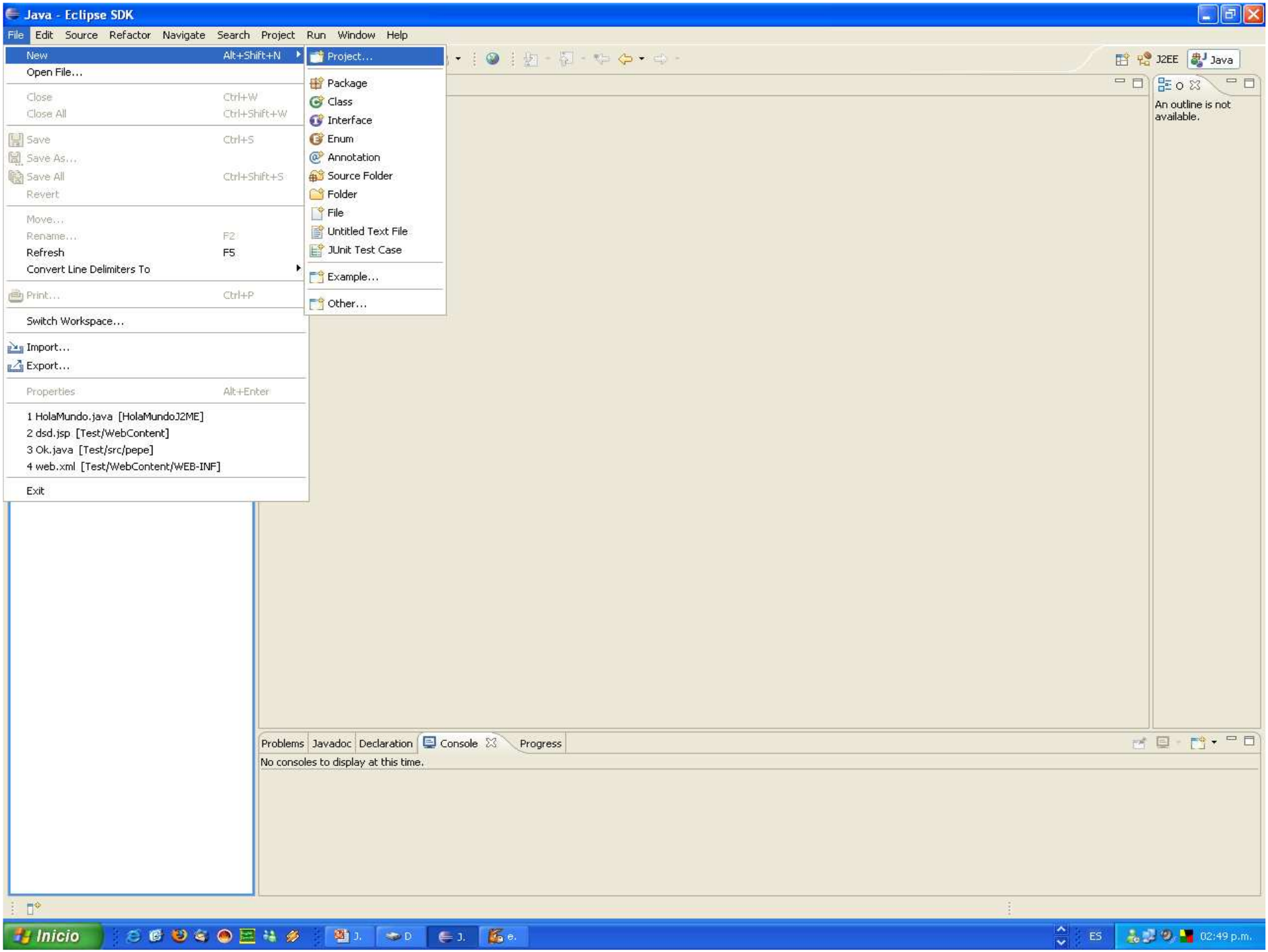
La maquina virtual

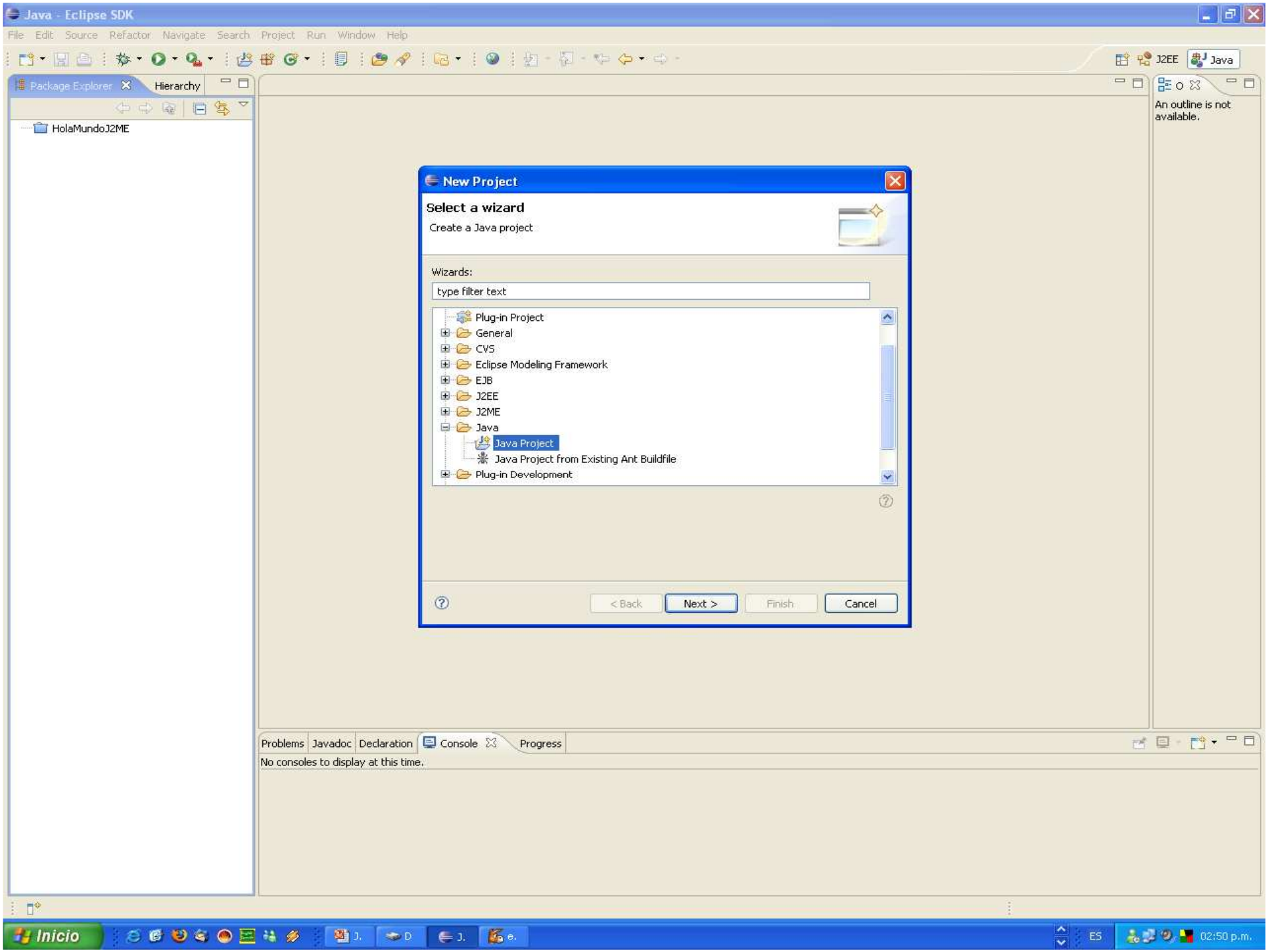


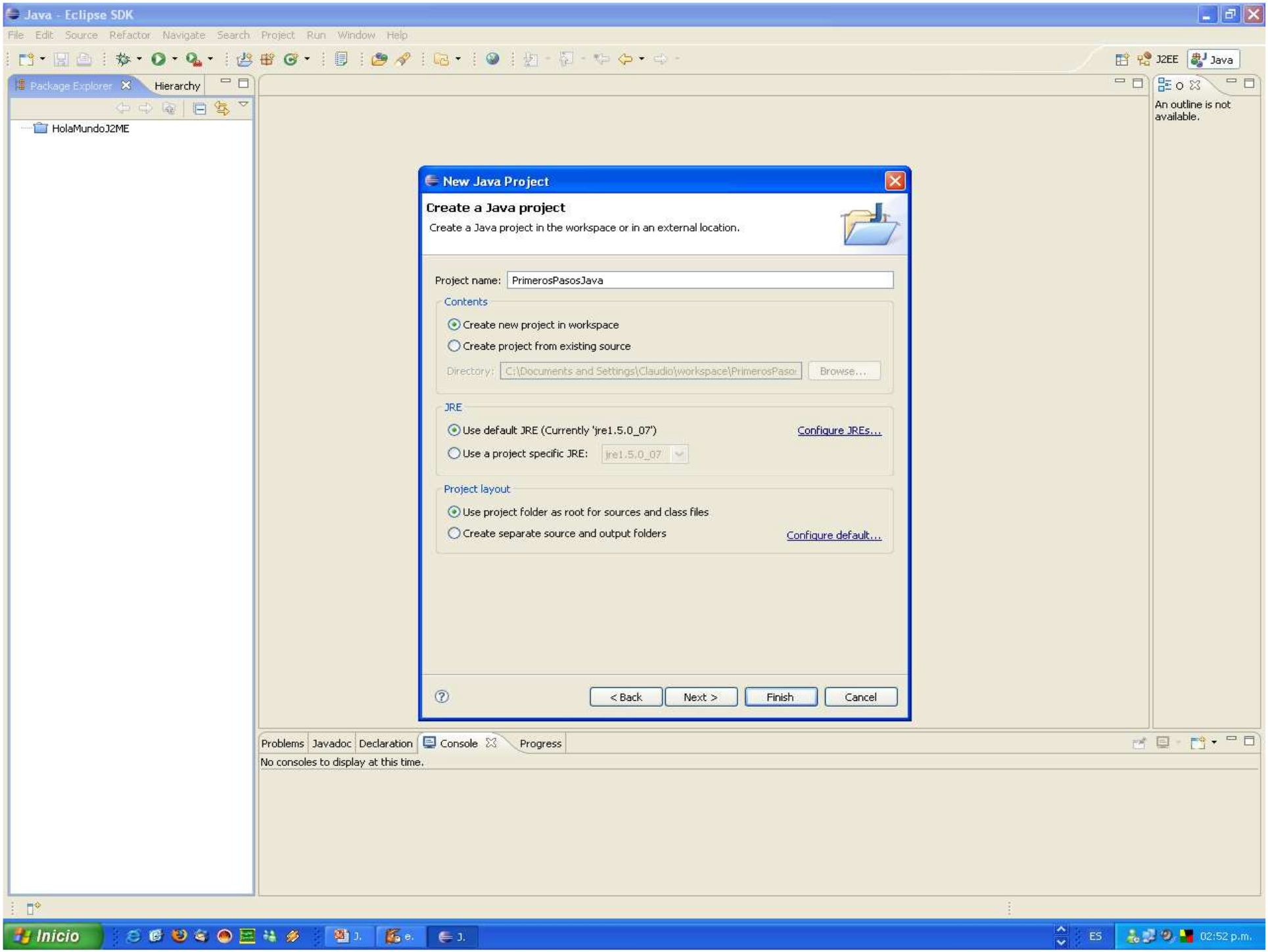
Primeros Pasos con Java y Eclipse: HolaMundo

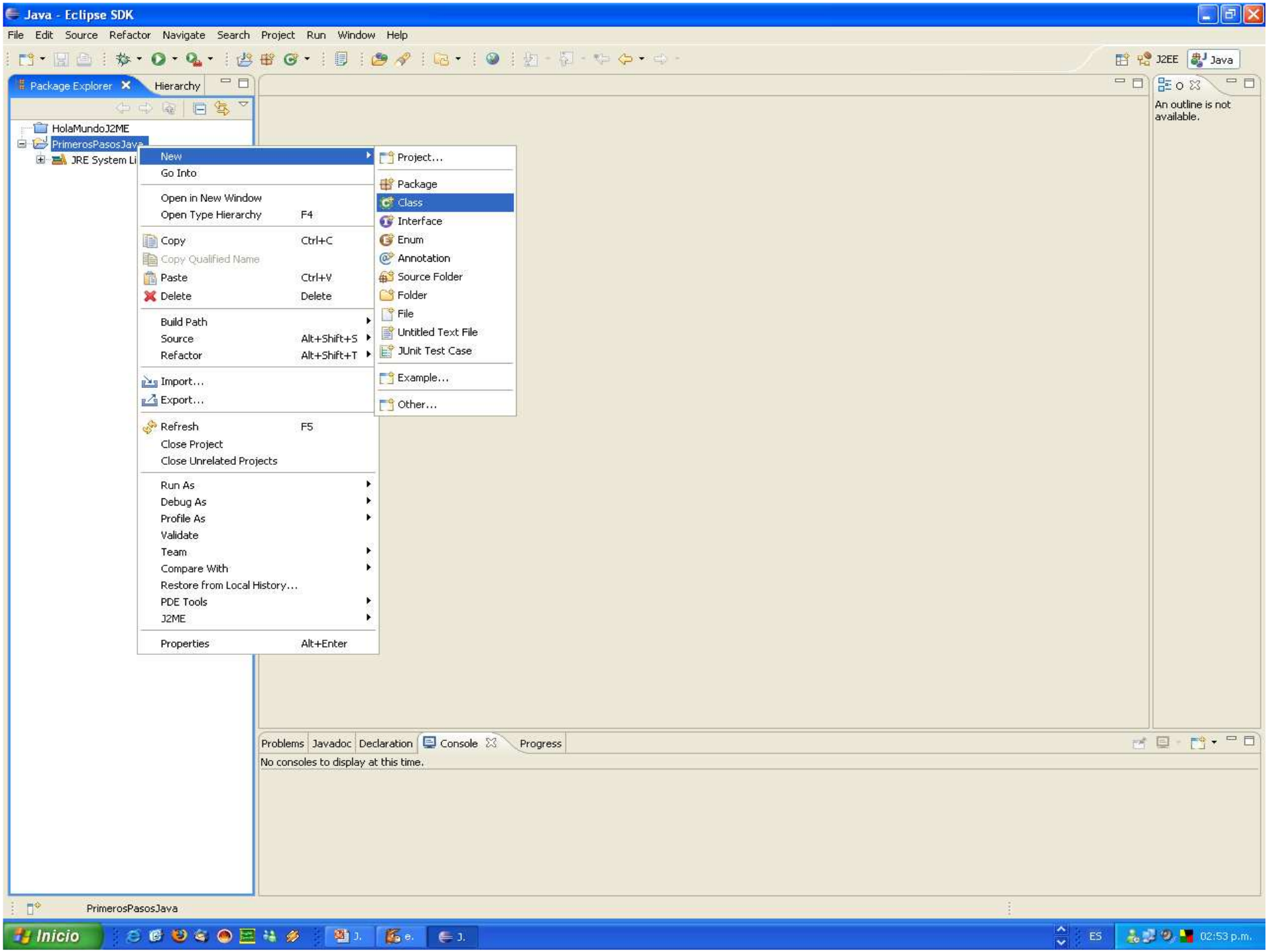


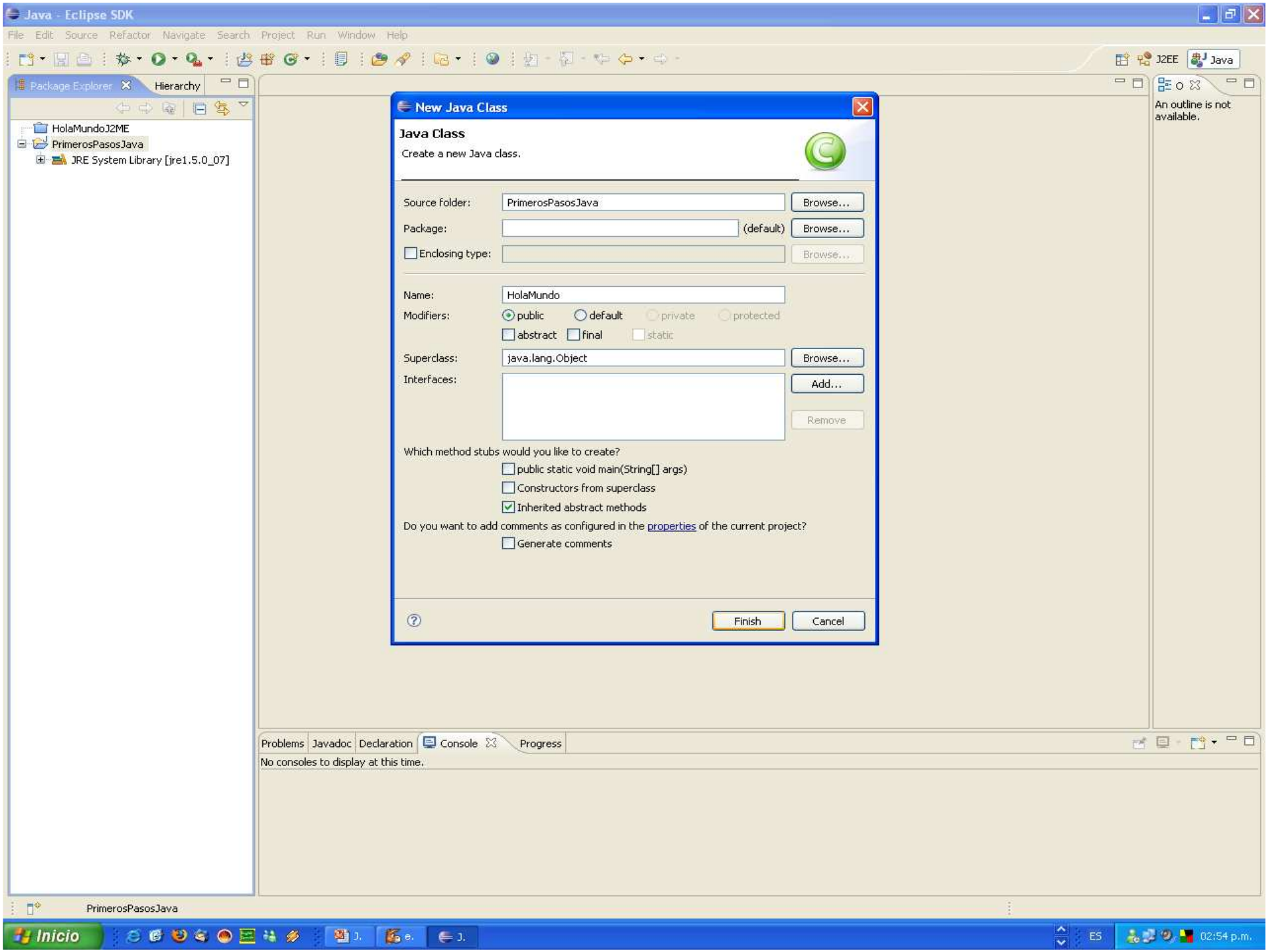

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
J2EE
Java
Package Explorer
Hierarchy
HolaMundoJ2ME
An outline is not available.
Problems Javadoc Declaration Console Progress
No consoles to display at this time.
Inicio
ES
02:47 p.m.


Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
New Alt+Shift+N
Open File...
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To
Print... Ctrl+P
Switch Workspace...
Import...
Export...
Properties Alt+Enter
1 HolaMundo.java [HolaMundoJ2ME]
2 dsd.jsp [Test/WebContent]
3 Ok.java [Test/src/pepe]
4 web.xml [Test/WebContent/WEB-INF]
Exit
Project...
Package
Class
Interface
Enum
Annotation
Source Folder
Folder
File
Untitled Text File
JUnit Test Case
Example...
Other...
J2EE
Java
An outline is not available.
Problems Javadoc Declaration Console Progress
No consoles to display at this time.
Inicio
ES
02:49 p.m.

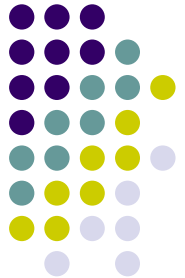

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
J2EE
Java
Package Explorer
Hierarchy
HolaMundoJ2ME
An outline is not available.
New Project
Select a wizard
Create a Java project
Wizards:
type filter text
Plug-in Project
General
CVS
Eclipse Modeling Framework
EJB
J2EE
J2ME
Java
Java Project
Java Project from Existing Ant Buildfile
Plug-in Development
< Back
Next >
Finish
Cancel
Problems
Javadoc
Declaration
Console
Progress
No consoles to display at this time.
Inicio
ES
02:50 p.m.


Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
J2EE
Java
Package Explorer
Hierarchy
HolaMundoJ2ME
An outline is not available.
New Java Project
Create a Java project
Create a Java project in the workspace or in an external location.
Project name: PrimerosPasosJava
Contents
Create new project in workspace
Create project from existing source
Directory: C:\Documents and Settings\Claudio\workspace\PrimerosPaso
Browse...
JRE
Use default JRE (Currently 'jre1.5.0_07')
Configure JREs...
Use a project specific JRE: jre1.5.0_07
Project layout
Use project folder as root for sources and class files
Create separate source and output folders
Configure default...
< Back
Next >
Finish
Cancel
Problems
Javadoc
Declaration
Console
Progress
No consoles to display at this time.
Inicio
ES
02:52 p.m.

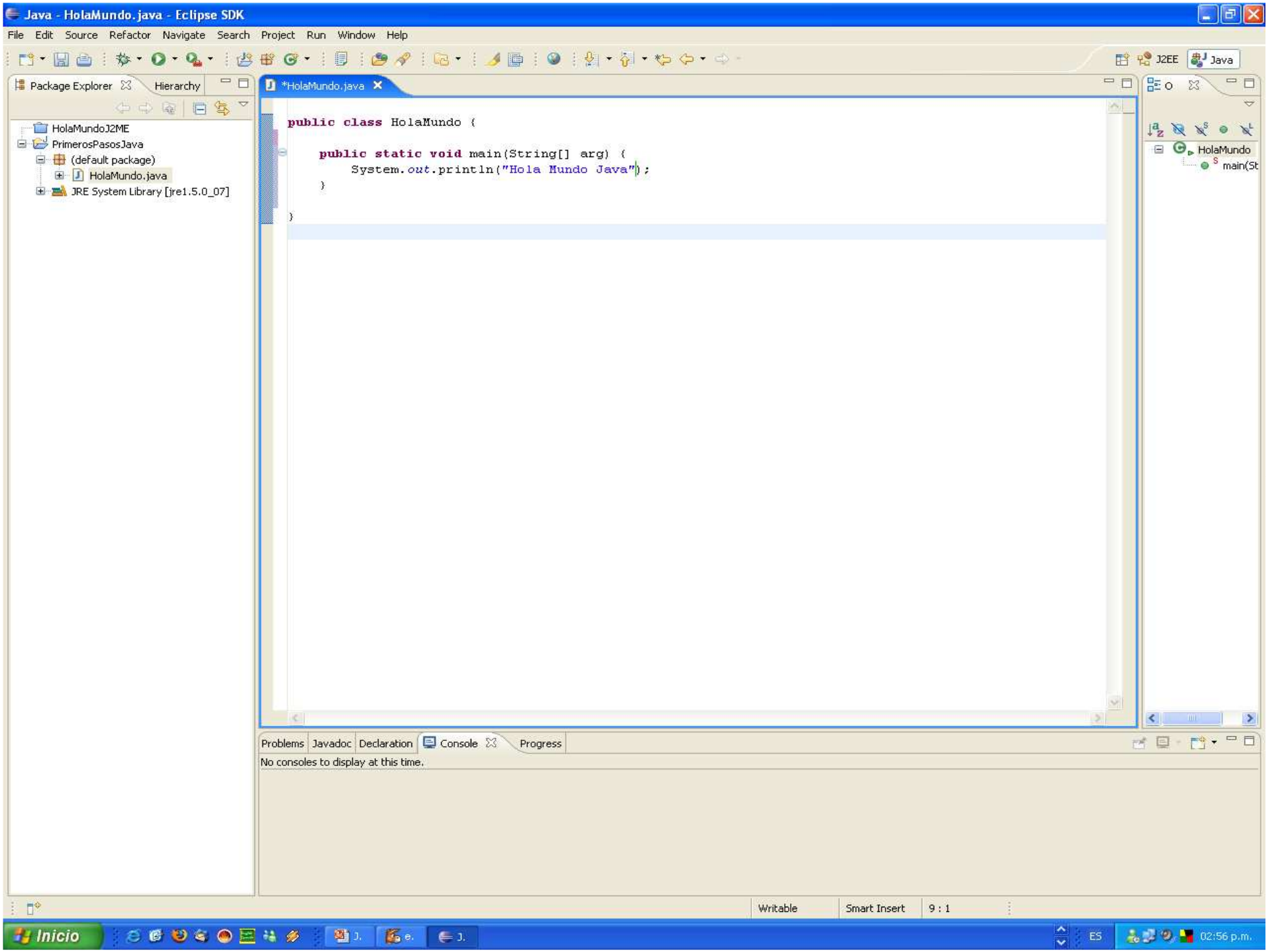

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
J2EE
Java
Package Explorer
Hierarchy
HolaMundoJ2ME
PrimerosPasosJava
JRE System Li
New
Go Into
Open in New Window
Open Type Hierarchy F4
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
Close Project
Close Unrelated Projects
Run As
Debug As
Profile As
Validate
Team
Compare With
Restore from Local History...
PDE Tools
J2ME
Properties Alt+Enter
Project...
Package
Class
Interface
Enum
Annotation
Source Folder
Folder
File
Untitled Text File
JUnit Test Case
Example...
Other...
An outline is not available.
Problems Javadoc Declaration Console Progress
No consoles to display at this time.
PrimerosPasosJava
Inicio
ES
02:53 p.m.

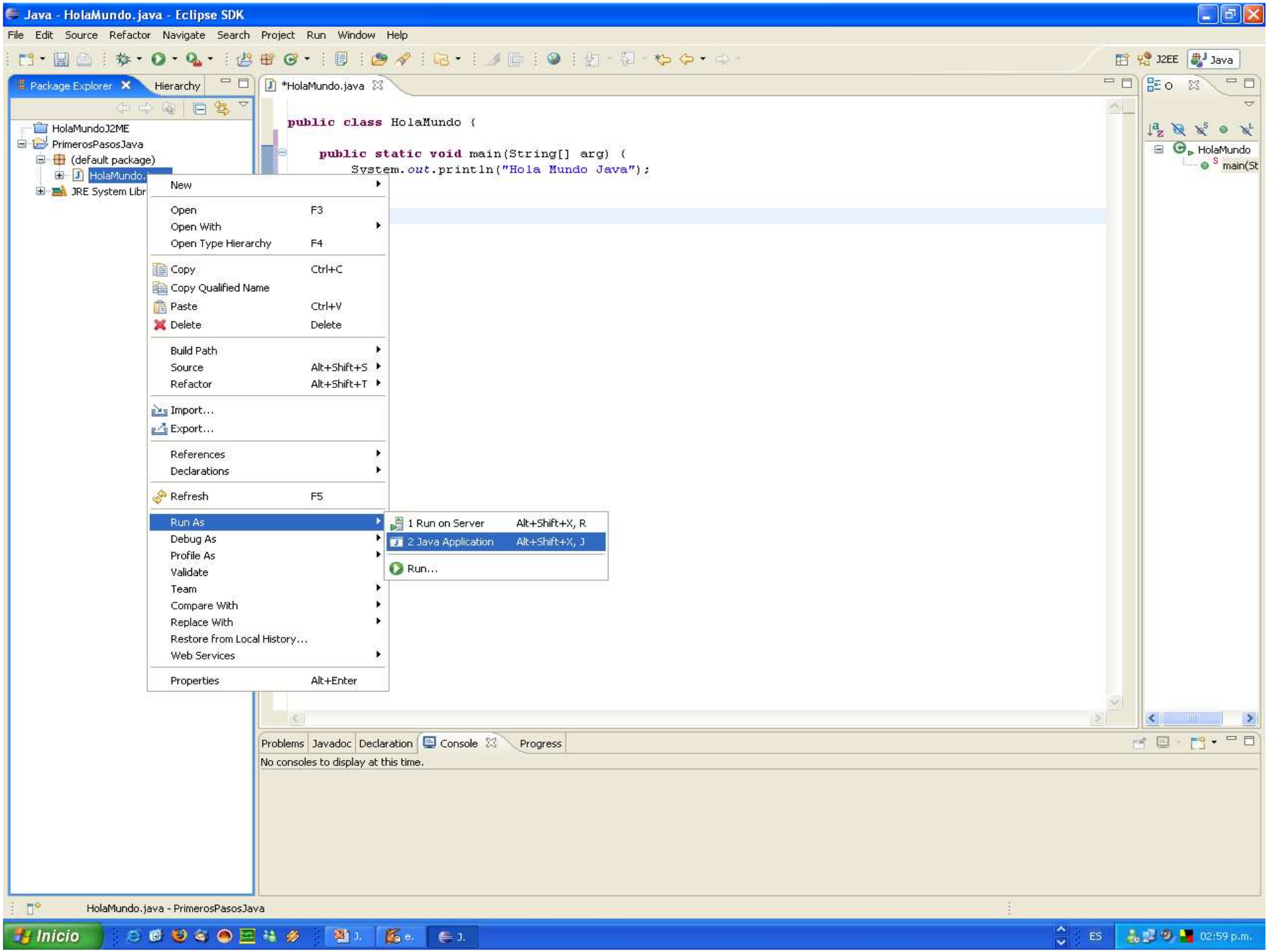

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
J2EE
Java
Package Explorer
Hierarchy
HolaMundoJ2ME
PrimerosPasosJava
JRE System Library [jre1.5.0_07]
An outline is not available.
New Java Class
Java Class
Create a new Java class.
Source folder:
PrimerosPasosJava
Browse...
Package:
(default)
Enclosing type:
Name:
HolaMundo
Modifiers:
public
default
private
protected
abstract
final
static
Superclass:
java.lang.Object
Interfaces:
Add...
Remove
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
Do you want to add comments as configured in the properties of the current project?
Generate comments
Finish
Cancel
Problems
Javadoc
Declaration
Console
Progress
No consoles to display at this time.
PrimerosPasosJava
Inicio
ES
02:54 p.m.

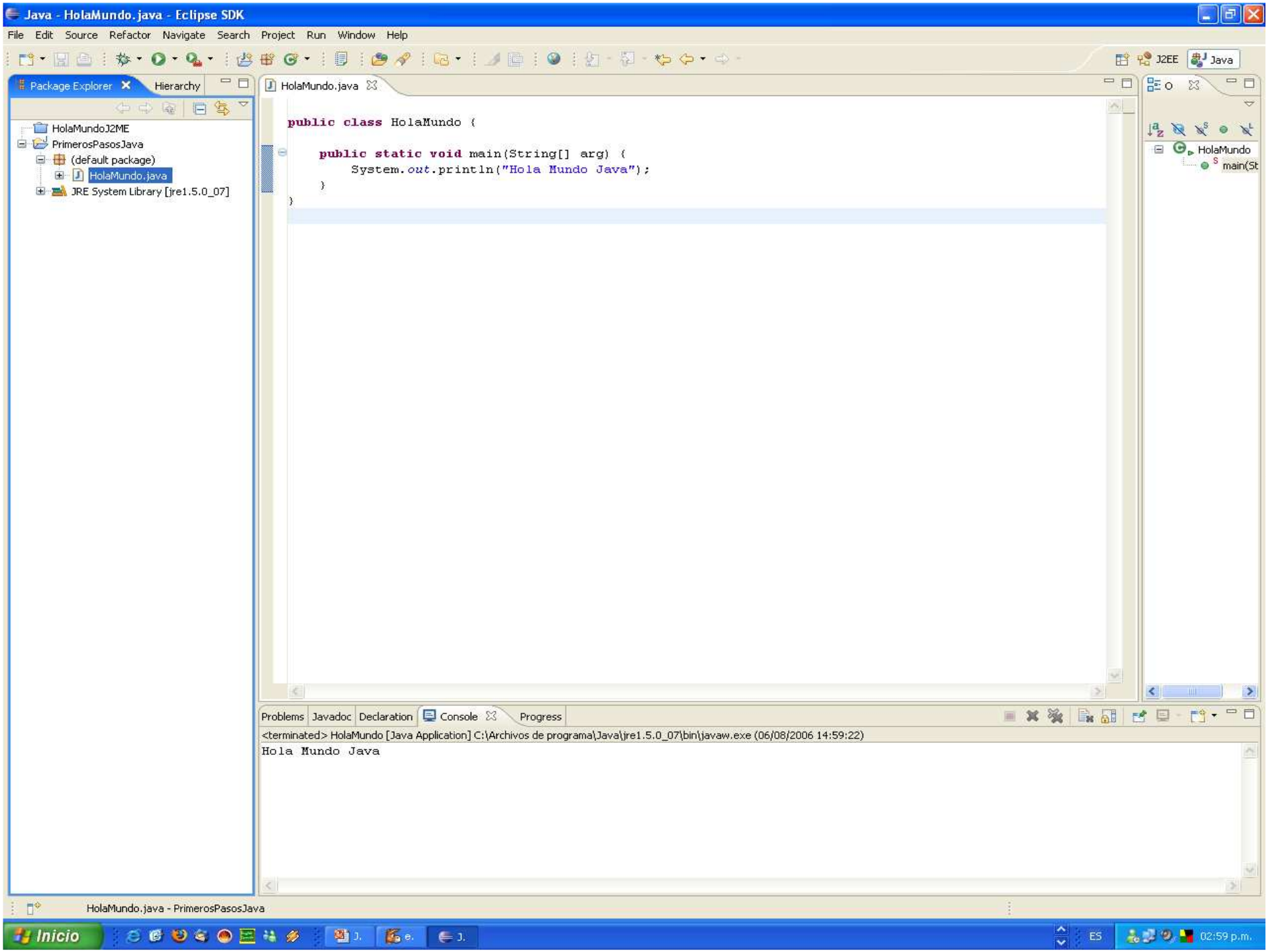
Primeros Pasos: HolaMundo



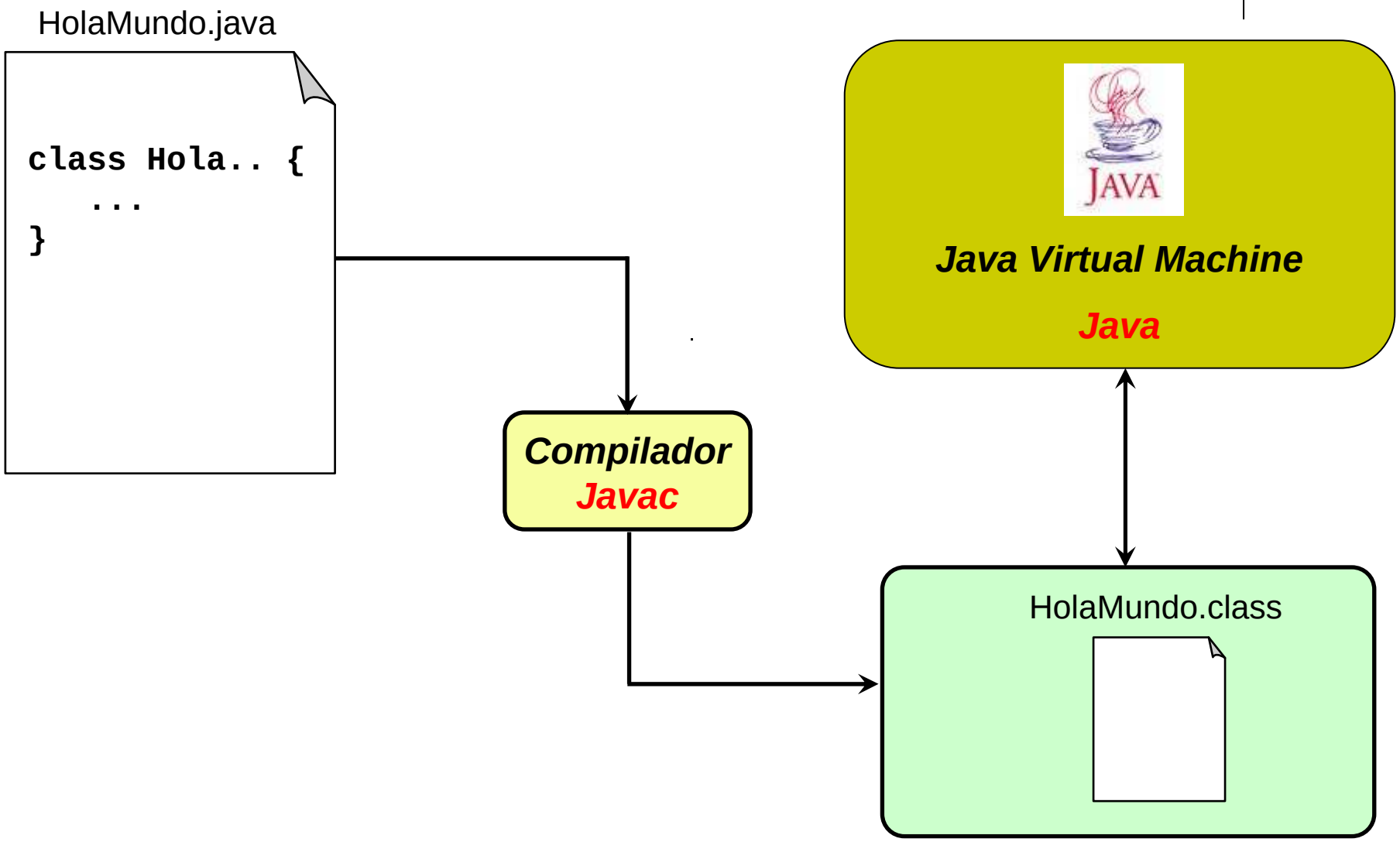
```
public class HolaMundo {

    public static void main(String[] arg) {
        System.out.println("Hola Mundo Java");
    }

}
```


Java - HolaMundo.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
J2EE
Java
Package Explorer
Hierarchy
HolaMundoJ2ME
PrimerosPasosJava
(default package)
HolaMundo.java
JRE System Library [jre1.5.0_07]
*HolaMundo.java
public class HolaMundo {
public static void main(String[] arg) {
System.out.println("Hola Mundo Java");
}
}
HolaMundo
main(St
Problems
Javadoc
Declaration
Console
Progress
No consoles to display at this time.
Writable
Smart Insert
9 : 1
Inicio
ES
02:56 p.m.


Java - HolaMundo.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
J2EE Java
Package Explorer Hierarchy
*HolaMundo.java
HolaMundoJ2ME
PrimerosPasosJava
(default package)
HolaMundo.
JRE System Libr
public class HolaMundo {
public static void main(String[] arg) {
System.out.println("Hola Mundo Java");
HolaMundo
main(St
New
Open F3
Open With
Open Type Hierarchy F4
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
References
Declarations
Refresh F5
Run As
Debug As
Profile As
Validate
Team
Compare With
Replace With
Restore from Local History...
Web Services
Properties Alt+Enter
1 Run on Server Alt+Shift+X, R
2 Java Application Alt+Shift+X, J
Run...
Problems Javadoc Declaration Console Progress
No consoles to display at this time.
HolaMundo.java - PrimerosPasosJava
Inicio
ES
02:59 p.m.

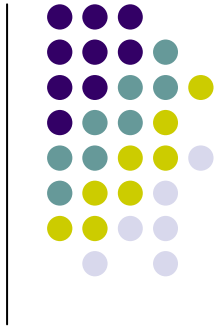

Java - HolaMundo.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
J2EE
Java
Package Explorer
Hierarchy
HolaMundoJ2ME
PrimerosPasosJava
(default package)
HolaMundo.java
JRE System Library [jre1.5.0_07]
HolaMundo.java
public class HolaMundo {
public static void main(String[] arg) {
System.out.println("Hola Mundo Java");
}
}
HolaMundo
main(St
Problems
Javadoc
Declaration
Console
Progress
<terminated> HolaMundo [Java Application] C:\Archivos de programa\Java\jre1.5.0_07\bin\javaw.exe (06/08/2006 14:59:22)
Hola Mundo Java
HolaMundo.java - PrimerosPasosJava
Inicio
ES
02:59 p.m.

Compilación



Sintaxis del lenguaje

Variables y tipos de datos



Tipos numéricos

Tipos enteros:

byte, enteros 8-bits
short, enteros 16-bits
int, enteros 32-bits
long, enteros 64-bits

Tipos en coma flotante:

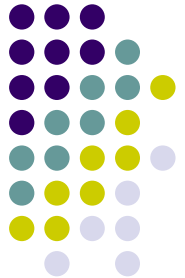
float, real en coma flotante, 32-bits
double, real en coma flotante, 64-bits

Tipos no numéricos

char, carácter 16-bits Unicode
boolean, tipo lógico, cuyos valores son true (cierto) y false (falso)
String , cadena de caracteres de 16bits Unicode

Sintaxis del lenguaje

Declaración de variables

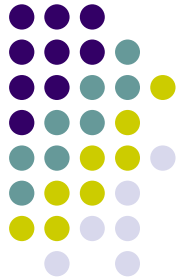


- Todas las variables y constantes deben ser declaradas antes de su uso
- Para declarar constantes, se utiliza la palabra reservada final.
- <tipo> <identificador> [=] <valor>
- Ej:

```
int numeroEntero = 34;
long altura = 0;
```

Sintaxis del lenguaje

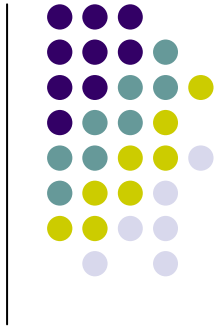
Operadores



- **Igualdad y desigualdad: == , !=**
- **Comparación: <, <=, >, >=**
- **Suma y resta: +, -**
- **Suma, multiplicación, división y módulo: +, -, *, /, %**
- **Incremento y decremento prefijos y posfijos: ++, --**
- **Lógica booleana: &&, ||**

Sintaxis del lenguaje

Asignaciones



Utilización de operadores:

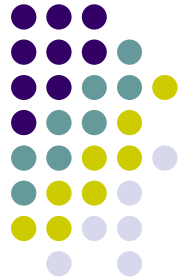
- **var1 = var2 + var3; // asignar a var1 el valor de var2 + var3**
- **var1 = var2 - ver3; // asignar a var1 el valor de var2 - var3**
- **var1 = var2 * ver3; // asignar a var1 el valor de var2 * var3**
- **var1 = var2 / ver3; // asignar a var1 el valor de var2 / var3**
- **var1 = var2 % ver3; // asignar a var1 el valor de var2 % var3**

Asignaciones de incremento decremento:

- **variable++; // añadir uno a variable**
- **variable--; // restar uno a varibale**
- **variable += exp; // añadir exp a variable**
- **variable -= exp; // restar exp a variable**

Sintaxis del lenguaje

if-else

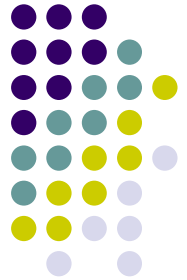


- if (condición) {
  instrucción;
  } else {
  instrucción;
  }

- if (velocidad > 100) {
  System.out.println("Multazo");
  } else {
  System.out.println("Buen conductor");
  }

Sintaxis del lenguaje

for



- for(inicio; prueba; actualización) {

  < cuerpo del bucle for >

  }

- for(int i=0; i<5;i++) {

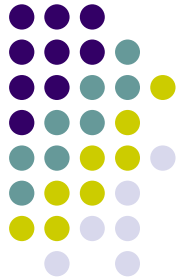
  cuenta += i;

  System.out.println("hola");

  }

Sintaxis del lenguaje

switch

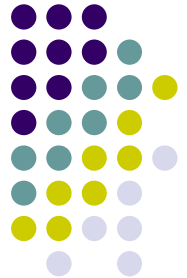


- **switch (expresion) {**
  case valor: instrucción; break;
  case valor: instrucción; break;
  ....
  default: instrucción; break;
  }

- **switch(nota) {**
  case 9:
  calificacion = ‘X’;
  break;
  case 7:
  calificacion = ‘N’;
  break;
  case 5:
  calificacion = ‘A’;
  break;
  default:
  calificacion = ‘S’; break;
  }

Sintaxis del lenguaje

while

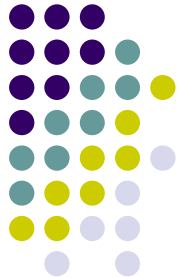


- <instrucciones para inicializar las condiciones>
  while (condiciones) {
  <instrucciones para llevar a cabo el bucle y cambiar condiciones >
  }

- int i=0;
  while(i<5) {
  System.out.println("hola");
  i++;
  }

Ver ejemplos

Ejercicios



Ejercicios:

1) Crear una clase Ejercicio1 que contenga solo el método main y que imprima los números solo los números pares hasta 100.

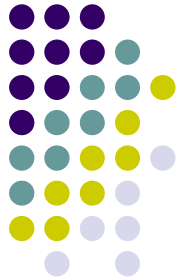
Pares del 0 al 100

0
2
4
6
..
..
100

2) Crear una clase Ejercicio2 que contenga solo el método main y que imprima la sumatoria de un numero con su anterior hasta que este llegue a 1024.

1
2
4
8
16
32
64
128
256
512
1024

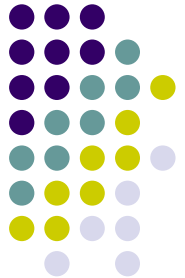
Ejercicios



3) Crear una clase de nombre Ejercicio3 que contenga sólo al método main y partiendo del String "Java es el mejor lenguaje de programación OO" declarada e inicializada como variable primitiva, mostrar por consola lo siguiente:

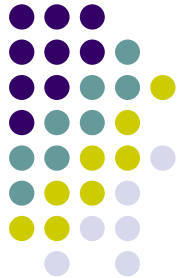
Su longitud
El carácter asociado al índice 7
La subcadena " lenguaje"
El índice que ocupa el carácter 'ó'
La String transformada en mayúsculas
Por último, comprobar si el primer carácter de la String es 'J' y mostrar por consola un mensaje que lo indique.

Programación Orientación a Objetos



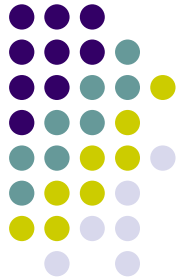
- **La Programación Orientada a Objetos (POO u OOP según siglas en inglés) es un paradigma de programación que define los programas en términos de "clases de objetos“.**
- **Los objetos que son entidades que combinan estado (es decir, atributos “variables”), comportamiento (esto es, métodos/funciones) e identidad (propiedad del objeto que lo diferencia del resto).**
- **La programación orientada a objetos expresa un programa como un conjunto de estos objetos (creados a partir de plantillas llamadas clases), que colaboran entre ellos para realizar tareas. Esto permite hacer los programas y módulos más fáciles de escribir, mantener y reutilizar.**

Programación Orientación a Objetos



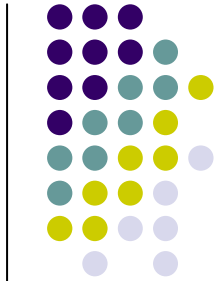
- **La programación estructurada anima al programador a pensar sobre todo en términos de procedimientos o funciones, y en segundo lugar en las estructuras de datos que esos procedimientos manejan.**
- **Los programadores que emplean lenguajes orientados a objetos definen objetos con datos y métodos y después envían mensajes a los objetos diciendo qué realicen esos métodos en sí mismos.**

Programación Orientación a Objetos: Características

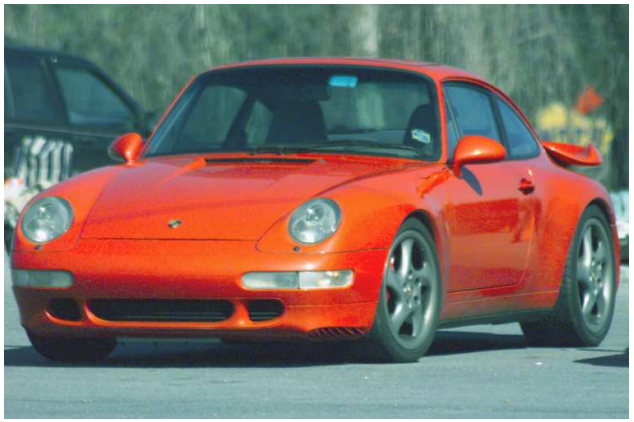


- **Abstracción:**
- **Encapsulamiento**
- **Herencia**
- **Polimorfismo**

Programación Orientación a Objetos: Abstracción



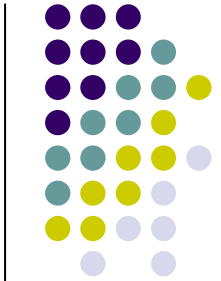
- **La abstracción es la propiedad que permite representar las características esenciales de un objeto, sin preocuparse de las restantes características (no esenciales)**



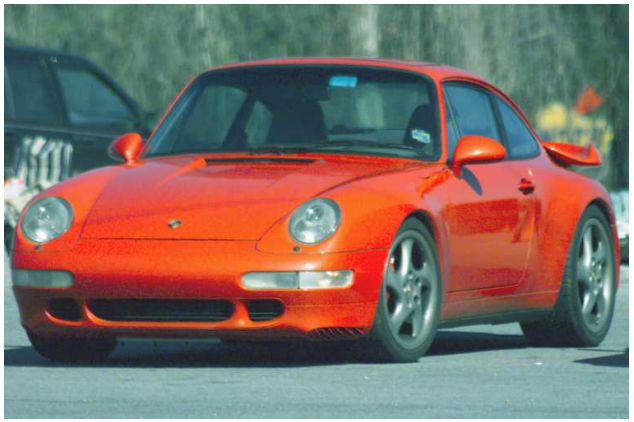
Auto.java

```
class Auto.. {
    motor..
    modelo..
    color..

}
```

Programación Orientación a Objetos: Encapsulamiento



- **Es el proceso de ocultar todos los secretos de un objeto que no contribuyen a sus características esenciales.**

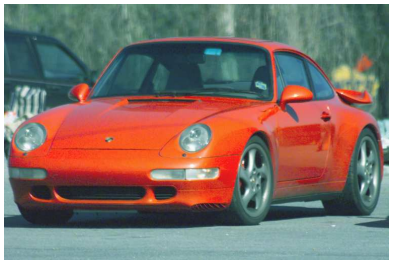


Auto.java

```
class Auto.. {
    motor..
    modelo..
    color..

}
```

Programación Orientación a Objetos: Herencia

- **La herencia define una relación entre clases, en donde una clase comparte la estructura o comportamiento definido en una o más clases (Padre – Hijo).**

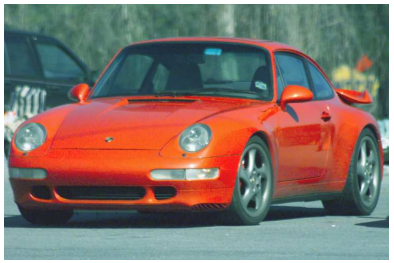


```
class Auto.. {
..
}
```

```
class AutoF1 extends
Auto {
}
```

Programación Orientación a Objetos: Polimorfismo

- **El polimorfismo permite referirse a objetos de clases diferentes mediante el mismo elemento de programa y realizar la misma operación de diferentes formas, según sea el objeto que se referencia en ese momento**.



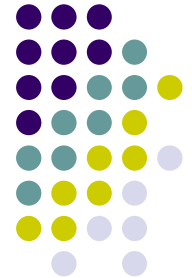
Auto.java

```
class Auto.. {
  arrancar..
}
```

AutoF1.java

```
class AutoF1..
{
  arrancar..
}
```

Programación Orientación a Objetos: Declaración de clases



- **public class nombreClase {**
  < declaración de atributos >
  < declaración de métodos >

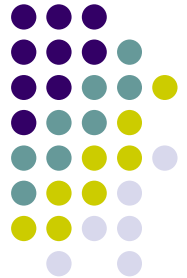
}

- **public class Auto {**

```
        private int velocidad;

        public void prenderMotor() { …..

        }

        public boolean elMotorEstaPrendido() { …..
```

}

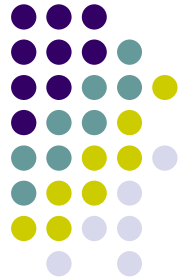
Programación Orientación a Objetos: Creación de Objetos



- **nombreClase nombreObjeto;**
- **nombreClase nombreObjeto = new nombreClase();**

- **Ejemplo:**
- **Auto fiat;**
- **Auto bmw = new Auto();**

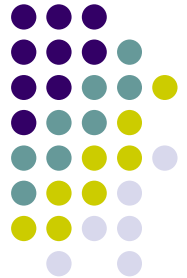
Programación Orientación a Objetos: definición de atributos



modificadores tipo nombreCampo;

- modificador puede ser:
  - **private**, accesible solo desde dentro de la propia clase
  - **protected**, accesible solo desde la propia clase, clases del mismo ***paquete*** o subclases
  - **public**,accesible por todos
- tipo es el tipo del atributo (tipo primitivo o clase del objeto)
- nombreCampo es el nombre del atributo
- Ej: private int edad = 0;

Programación Orientación a Objetos: definición de metodos



modificador tipo nombreMetodo(<lista de parametros formales>) {

< cuerpo del método; instrucciones >

return expresion; // solo métodos que devuelven resultado

}

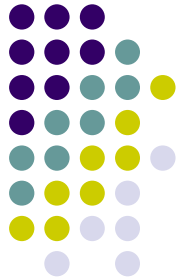
modificador significa lo mismo que para los atributos.

tipo es el tipo devuelto por el método, puede ser:

- **void** si el método no devuelve nada
- un tipo primitivo
- una clase

nombreMetodo es el nombre del método

Programación Orientación a Objetos: definición de metodos

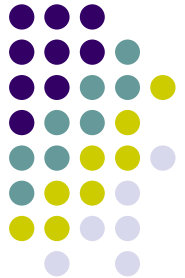


Ej:

```
Public class Auto …….
    private int capacidadDelTanque;

    public boolean cargarNafta(int cantidad, char tipo) {
            capacidadDelTanque = capacidadDelTanque + cantidad;
            …..
            return true;
    }
}
```

Ver ejemplo

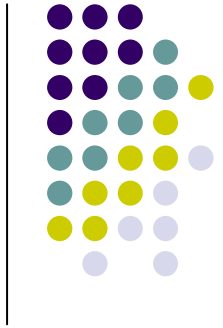
Programación Orientación a Objetos: llamando a metodos



- **Ejemplo:**

```
Auto bmw = new Auto();
boolean estado = bmw.cargarNafta(20,'S');
bmw.arrancar();
```

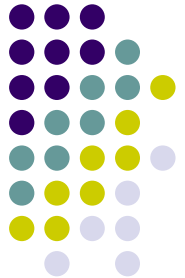
Ejercicios



Ejercicios:
4) Crear una clase Ejercici4 que contenga solo el método main y que permita ingresar por teclado 2 números y la operación deseada delegando la responsabilidad de calcular el resultado a una clase calculadora, luego crear la clase calculadora que contenga las operaciones de suma, resta, multiplicación y división.

Ingrese el primer numero:3
Ingrese el segundo numero:3
Operación (1: suma,2: resta,3: mult, 4:div):1
Resultado: 6

Links



www.java.sun.com

www.eclipse.org

www.javahispano.org

www.programacion.com

www.onjava.com

www.ibm.com

¿Preguntas?