Programación de Computadores: Introducción a Java - Tipos de Datos, Expresiones, Instrucciones de Control

Juan F. Pérez

Departamento MACC
Matemáticas Aplicadas y Ciencias de la Computación
Universidad del Rosario

juanferna.perez@urosario.edu.co

Segundo Semestre de 2017

Contenidos

Introducción

¿Qué es Java?

Respuestas...

¿Qué es Java?

Respuestas...

- Un lenguaje de programación...

¿Qué es Java?

Respuestas...

- Un lenguaje de programación...
- ... de alto nivel.

¿Qué es Java?

Respuestas...

- Un lenguaje de programación...
- ... de alto nivel.
- ... imperativo.

¿Qué es Java?

Respuestas...

- Un lenguaje de programación...
- ... de alto nivel.
- ... imperativo.
- ... orientado a objetos.

¿Qué es Java?

Respuestas...

- Un lenguaje de programación...
- ... de alto nivel.
- ... imperativo.
- ... orientado a objetos.
- ... interpretado.

Algunas características de Java

- Más complejo que Python para iniciar

Algunas características de Java

- Más complejo que Python para iniciar
- Requiere definir variables con más cuidado (evita problemas)

Algunas características de Java

- Más complejo que Python para iniciar
- Requiere definir variables con más cuidado (evita problemas)
- Bloques de código demarcados por corchetes $\{\}$ (indentación no obligatoria, pero reconmendada)

Algunas características de Java

- Más complejo que Python para iniciar
- Requiere definir variables con más cuidado (evita problemas)
- Bloques de código demarcados por corchetes $\{\}$ (indentación no obligatoria, pero reconmendada)
- Usado en muchas áreas (e.g., aplicaciones empresariales, web, android)

Algo de historia

- Desarrollado por James Gosling, Mike Sheridan y Patrick Naughton a partir de 1991

Algo de historia

- Desarrollado por James Gosling, Mike Sheridan y Patrick Naughton a partir de 1991
- Sun Microsystems lanzó su primera implementación pública en 1995.
- Promesa: "Write Once, Run Anywhere" (WORA).

Algo de historia

- Desarrollado por James Gosling, Mike Sheridan y Patrick Naughton a partir de 1991
- Sun Microsystems lanzó su primera implementación pública en 1995.
- Promesa: "Write Once, Run Anywhere" (WORA).
- `https://en.wikipedia.org/wiki/Java_(programming_language)`

Algo más de historia - Versiones

- JDK 1.0 (1996)

Algo más de historia - Versiones

- JDK 1.0 (1996)
- JDK 1.1 (1997)

Algo más de historia - Versiones

- JDK 1.0 (1996)
- JDK 1.1 (1997)
- J2SE 1.2 (1998)

Algo más de historia - Versiones

- JDK 1.0 (1996)
- JDK 1.1 (1997)
- J2SE 1.2 (1998)
- J2SE 1.3 (2000)

Algo más de historia - Versiones

- JDK 1.0 (1996)
- JDK 1.1 (1997)
- J2SE 1.2 (1998)
- J2SE 1.3 (2000)
- J2SE 1.4 (2002)

Algo más de historia - Versiones

- JDK 1.0 (1996)
- JDK 1.1 (1997)
- J2SE 1.2 (1998)
- J2SE 1.3 (2000)
- J2SE 1.4 (2002)
- J2SE 5.0 (2004)

Algo más de historia - Versiones

- JDK 1.0 (1996)
- JDK 1.1 (1997)
- J2SE 1.2 (1998)
- J2SE 1.3 (2000)
- J2SE 1.4 (2002)
- J2SE 5.0 (2004)
- Java SE 6 (2006)

Algo más de historia - Versiones

- JDK 1.0 (1996)
- JDK 1.1 (1997)
- J2SE 1.2 (1998)
- J2SE 1.3 (2000)
- J2SE 1.4 (2002)
- J2SE 5.0 (2004)
- Java SE 6 (2006)
- Java SE 7 (2011)

Algo más de historia - Versiones

- JDK 1.0 (1996)
- JDK 1.1 (1997)
- J2SE 1.2 (1998)
- J2SE 1.3 (2000)
- J2SE 1.4 (2002)
- J2SE 5.0 (2004)
- Java SE 6 (2006)
- Java SE 7 (2011)
- Java SE 8 (2014)

Orientado a Objetos e Interpretado

Orientado a Objetos vs. Procedimental

Programación procedimental

- Lo que hemos hecho hasta el momento
- Procedimientos que ejecutan instrucciones sobre información

Orientado a Objetos vs. Procedimental

Programación procedimental

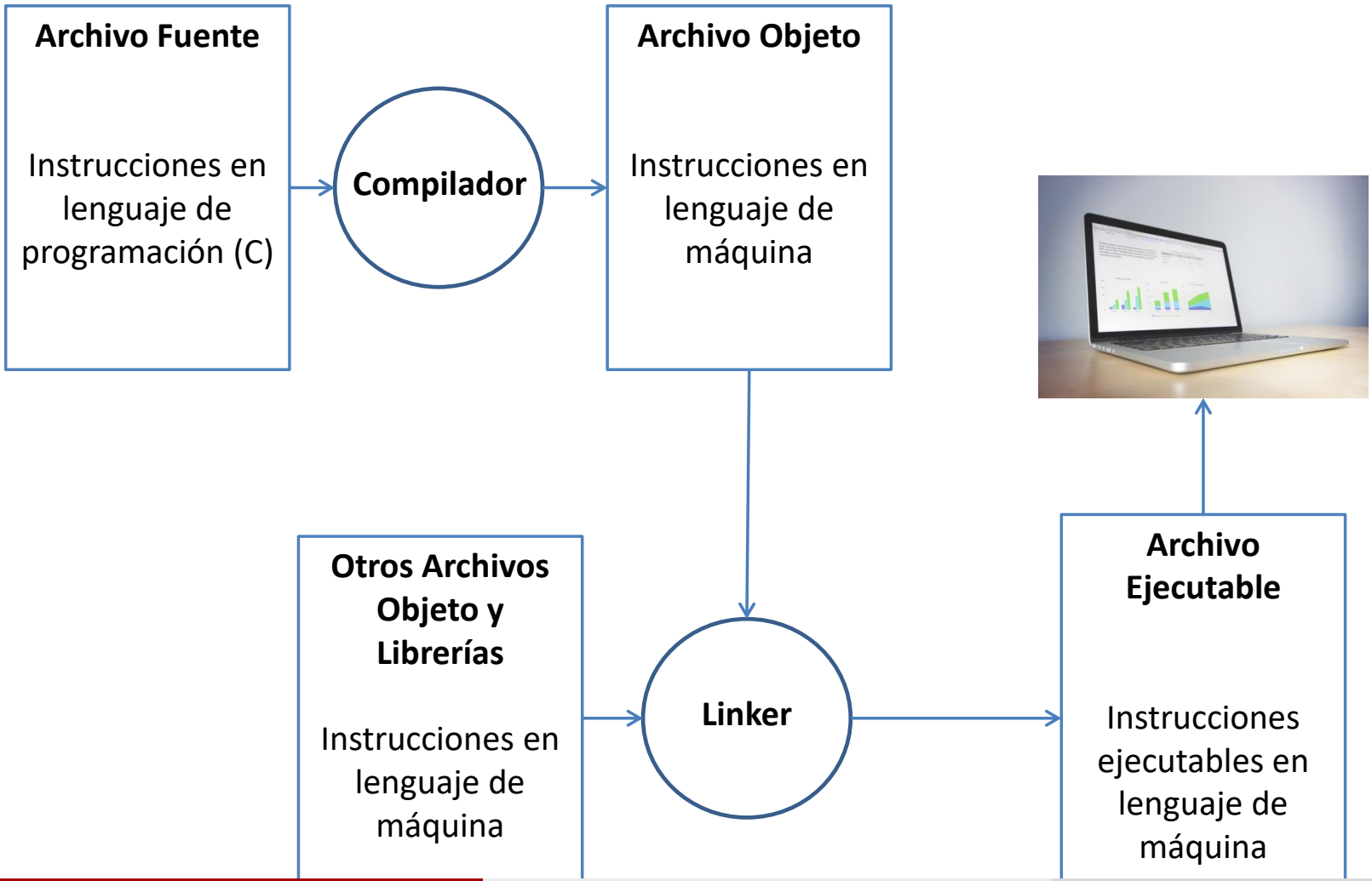
- Lo que hemos hecho hasta el momento
- Procedimientos que ejecutan instrucciones sobre información

Programación orientada a Objetos

- Objetos de diferentes **clases** interactúan a través de mensajes
- Objeto definido por su estado y comportamiento

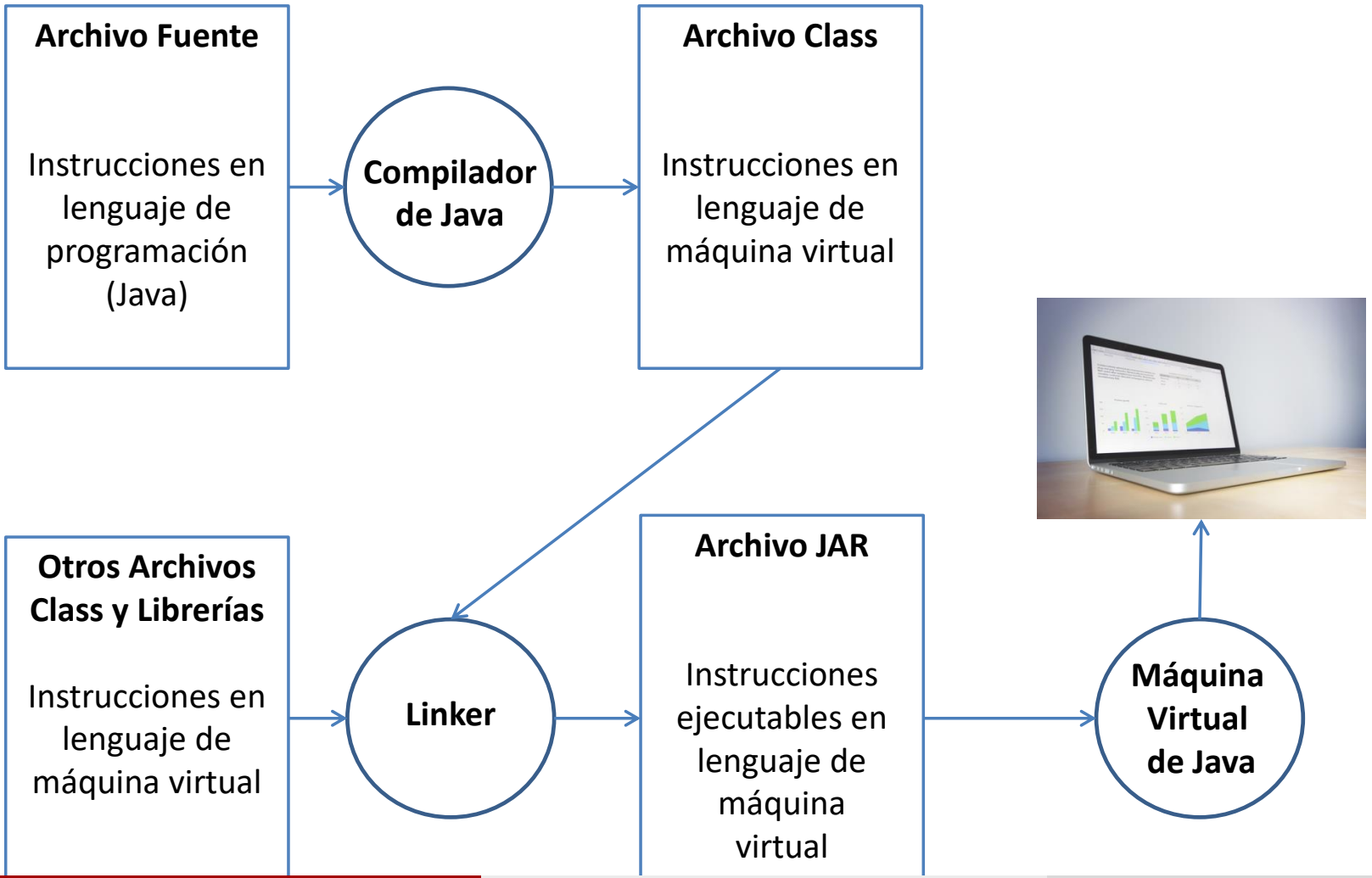
Interpretado vs. Compilado

Lenguaje Compilado



Interpretado vs. Compilado

Lenguaje Interpretado



Un primer código

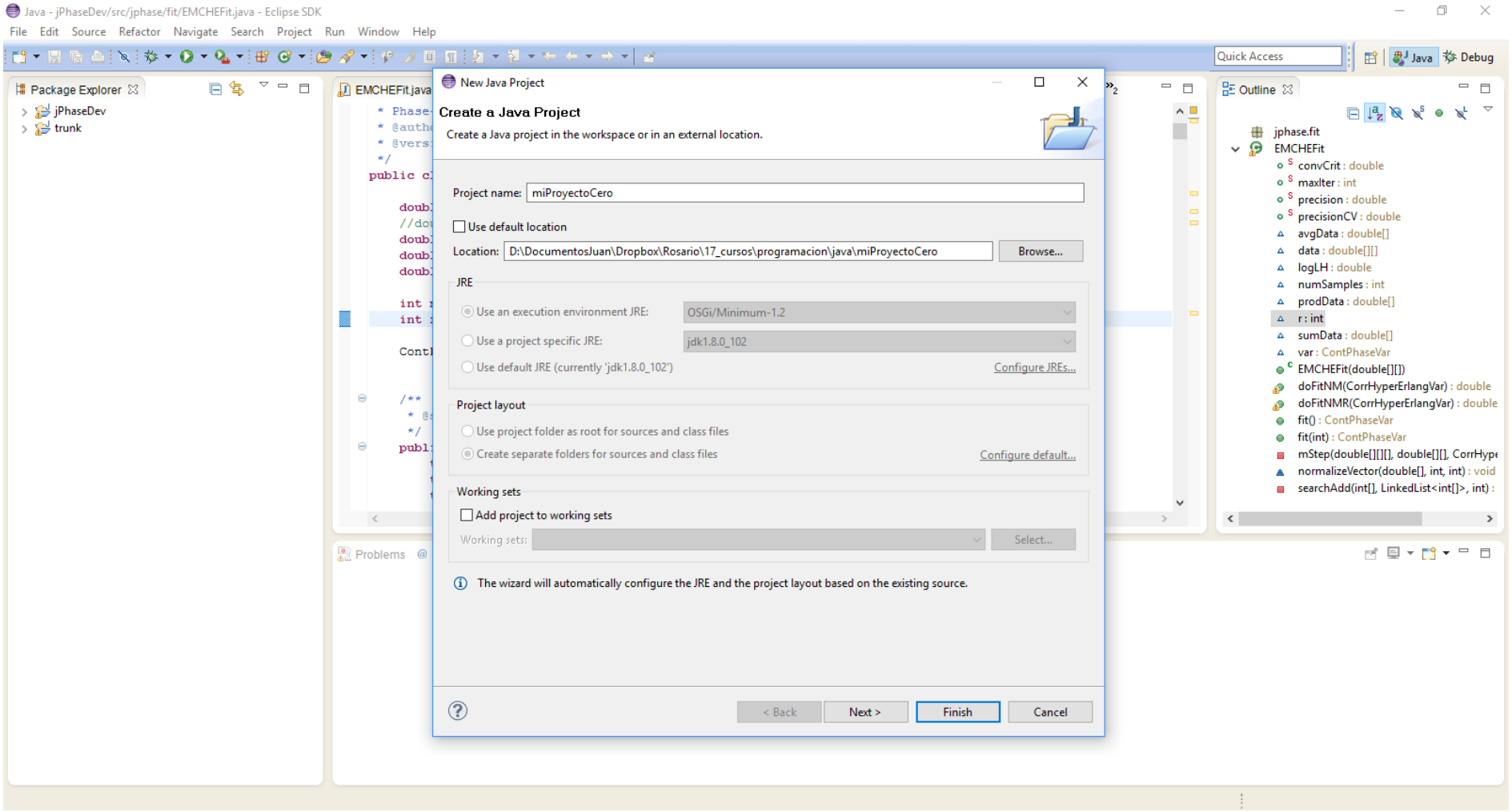
El lenguaje Java

- Algunas instrucciones primitivas
- Algo de gramática
- Algunos símbolos/palabras reservados
- Algunas instrucciones no serán claras inicialmente

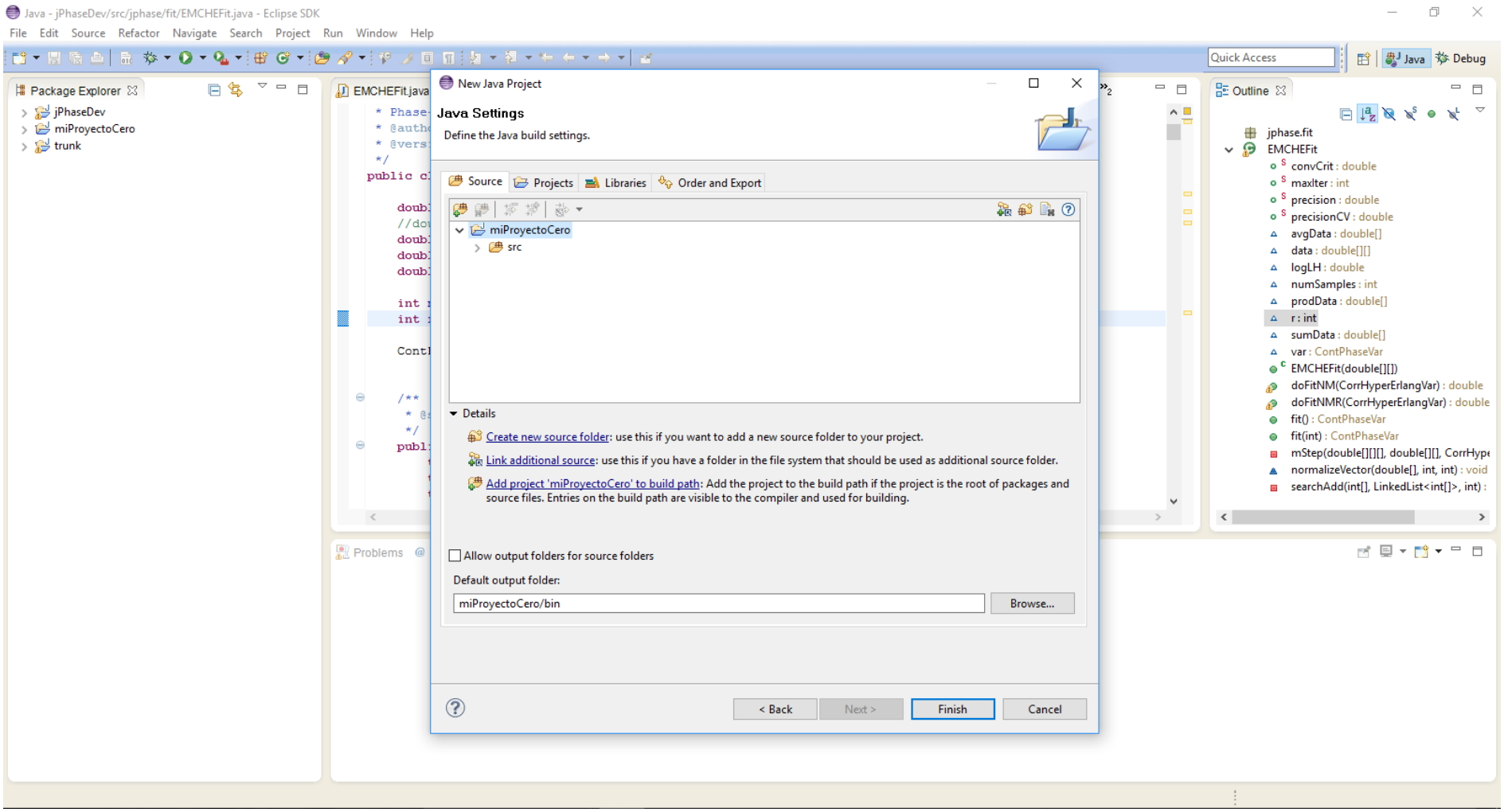
"Hola mundo" en Java

- Java Development Kit (JDK) versión 8.0
- Editor: Eclipse
- Nuevo proyecto en Java

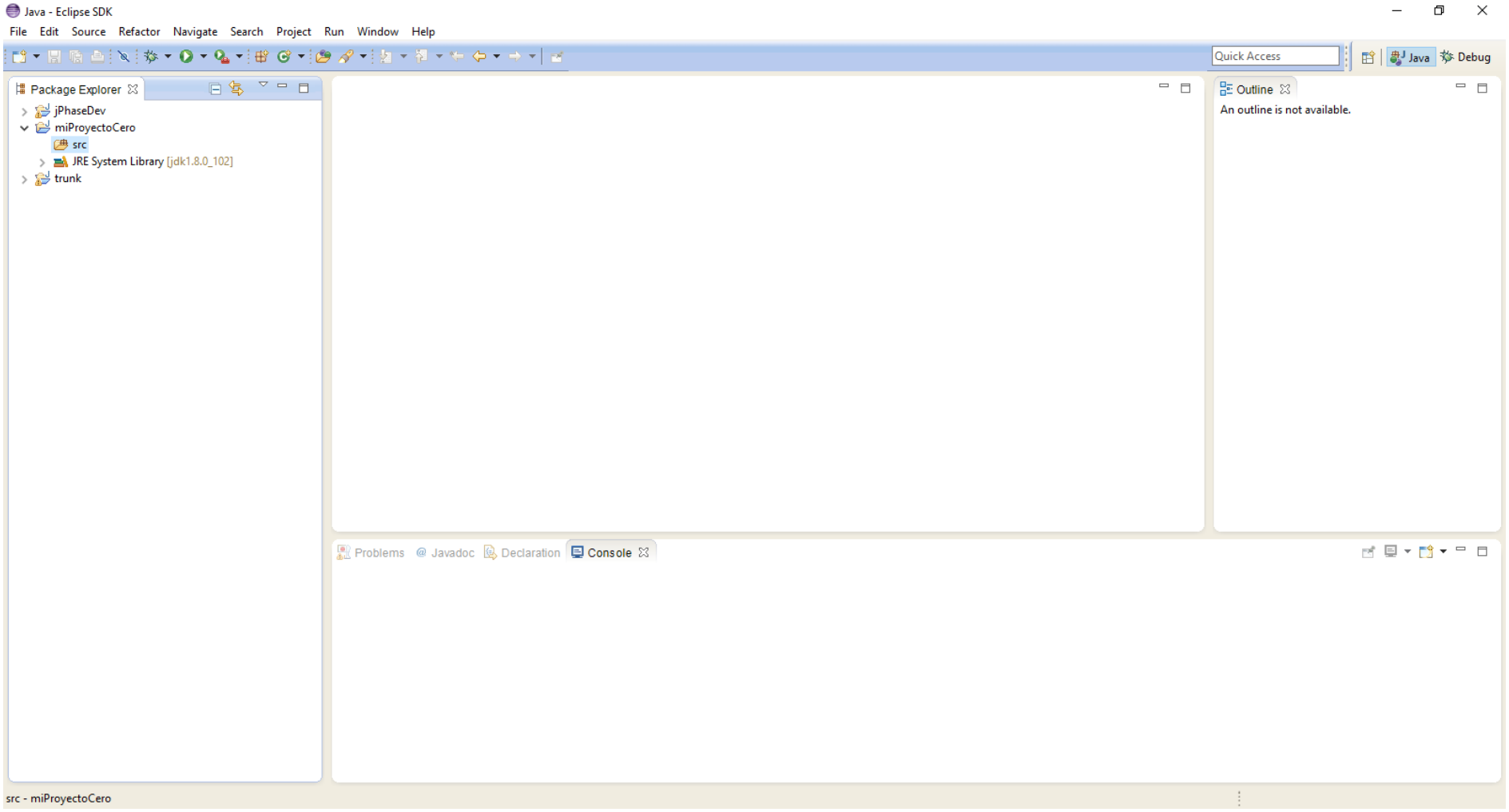
Creamos un nuevo proyecto en Java



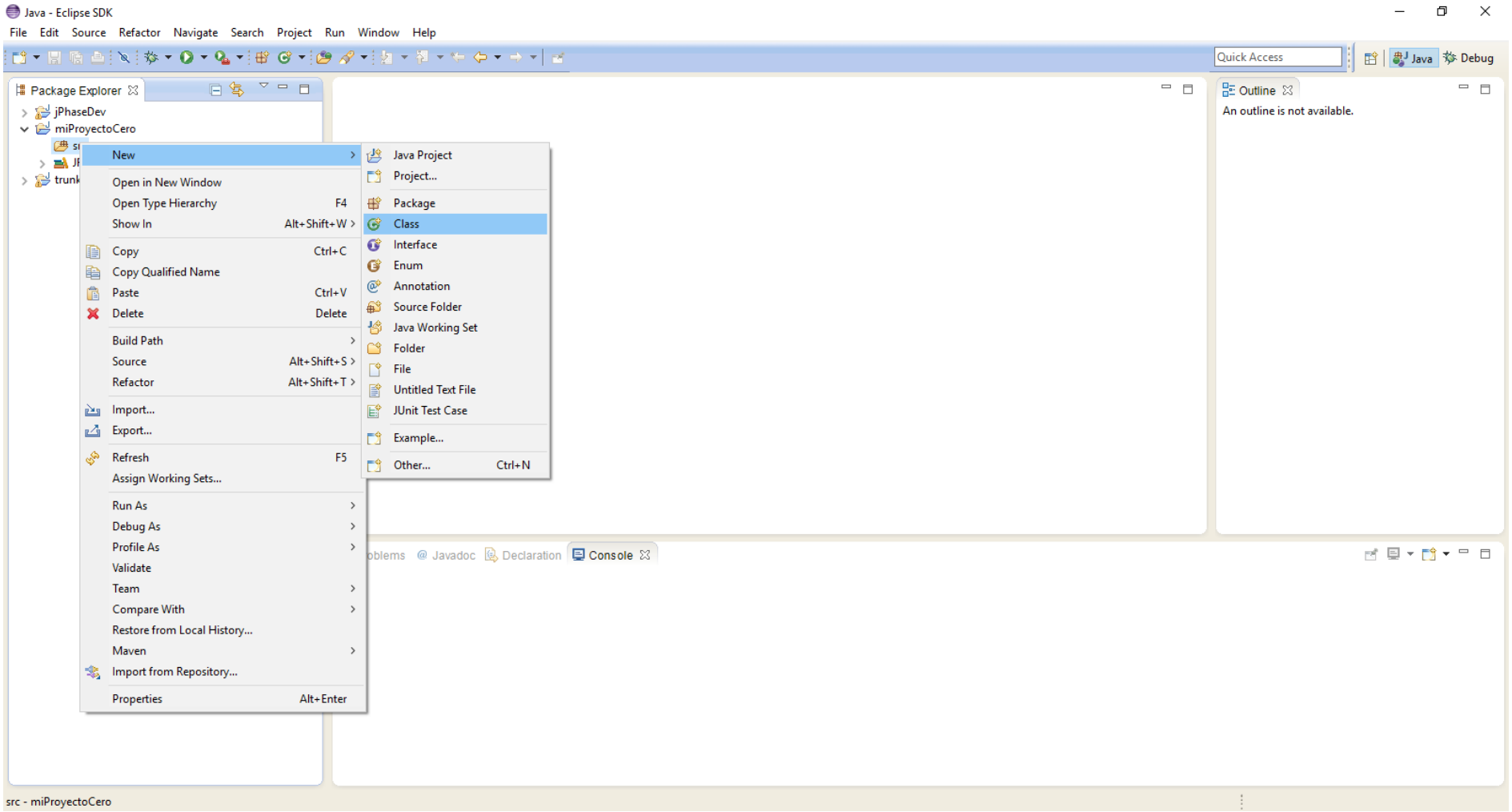
Creamos un nuevo proyecto en Java



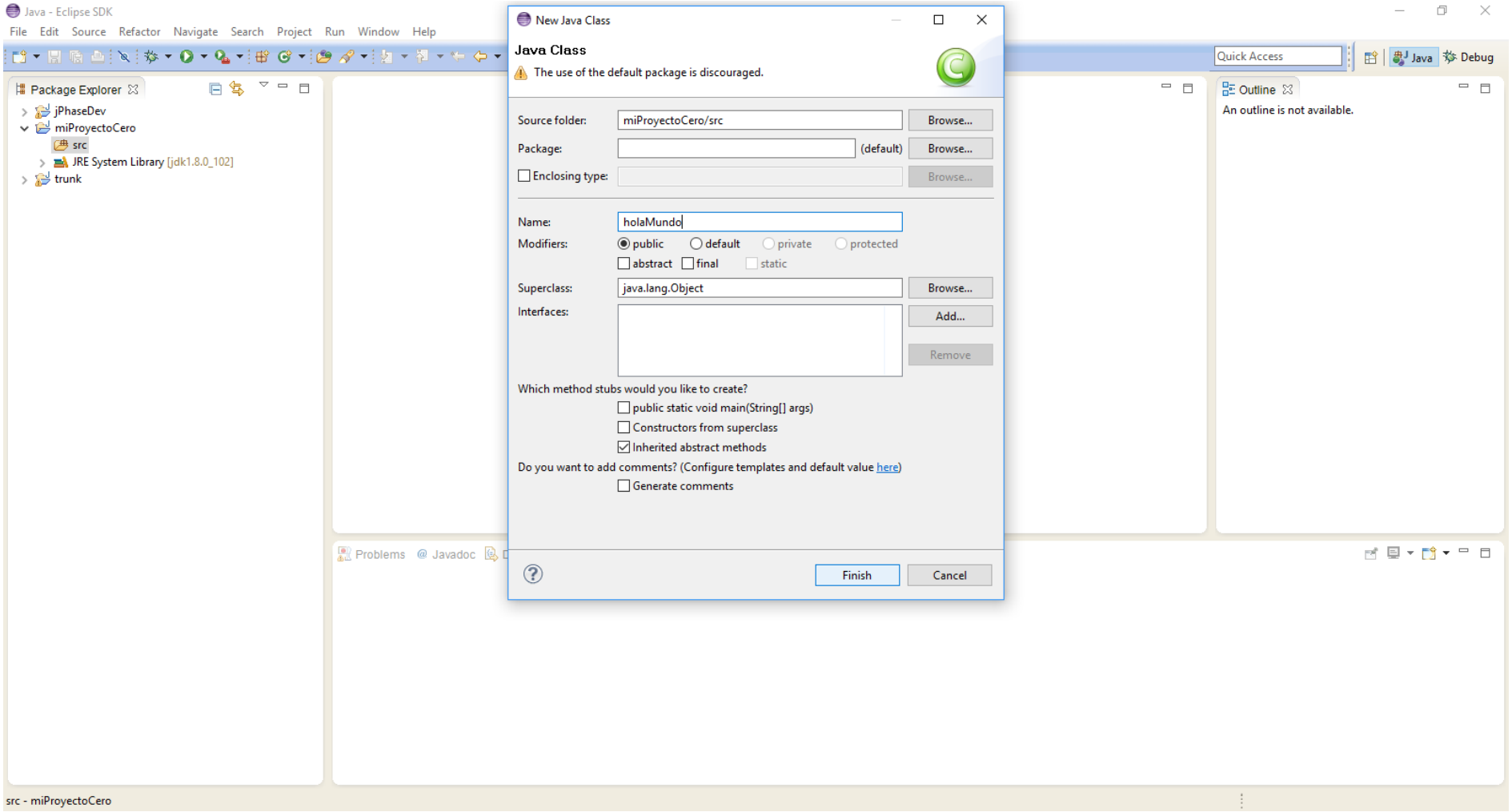
Creamos un nuevo proyecto en Java



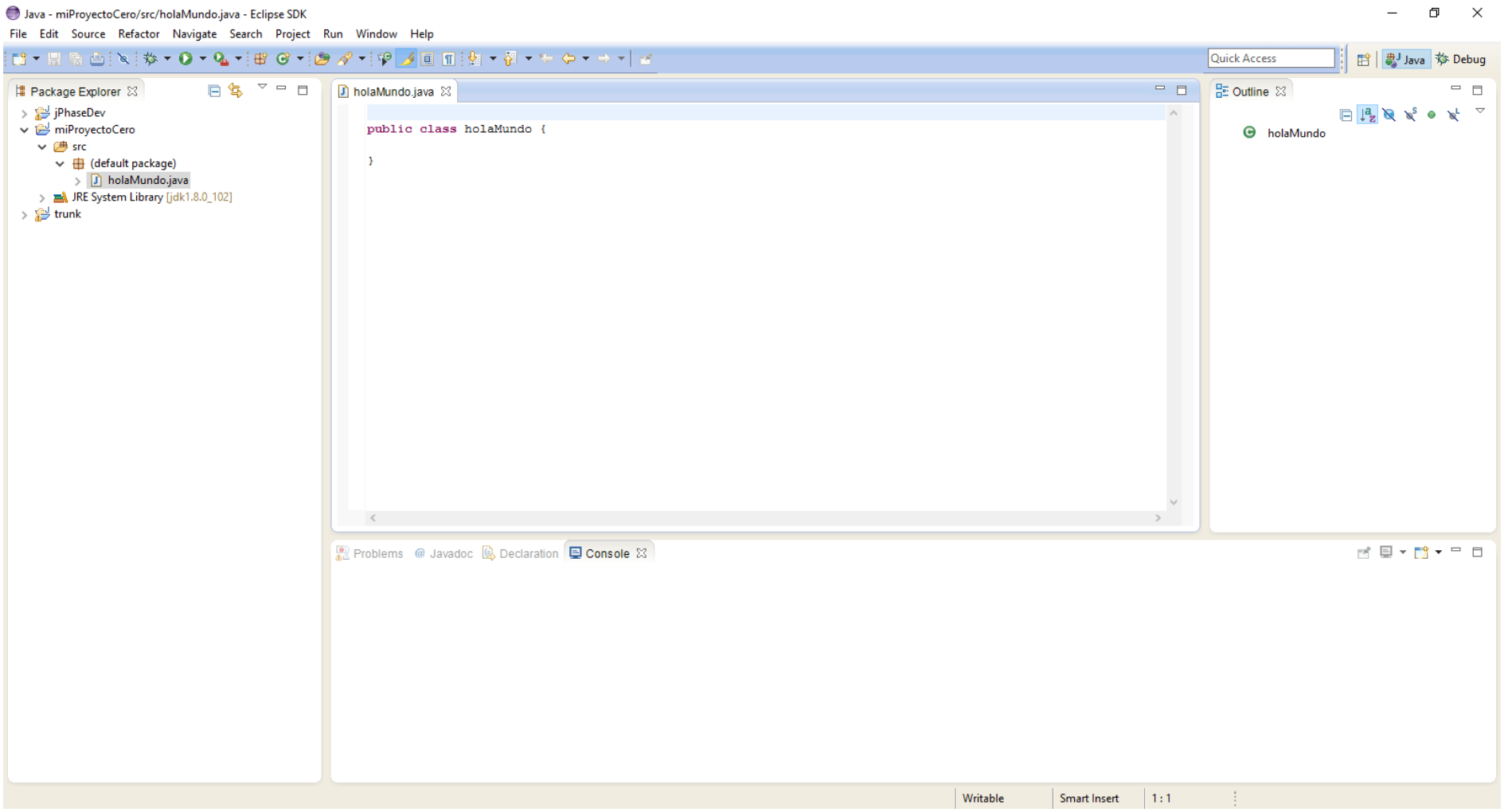
Creamos una nueva clase



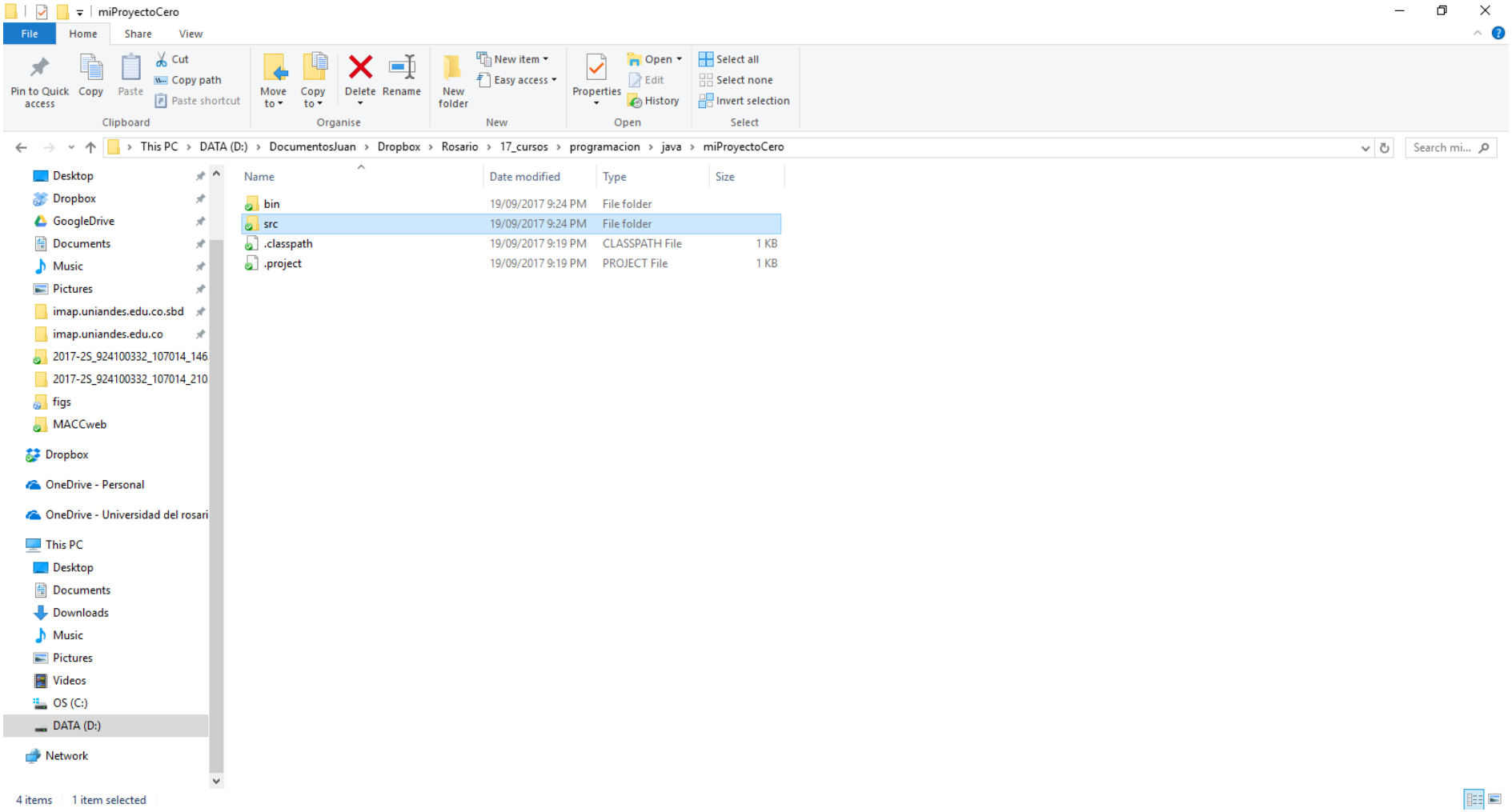
Creamos una nueva clase



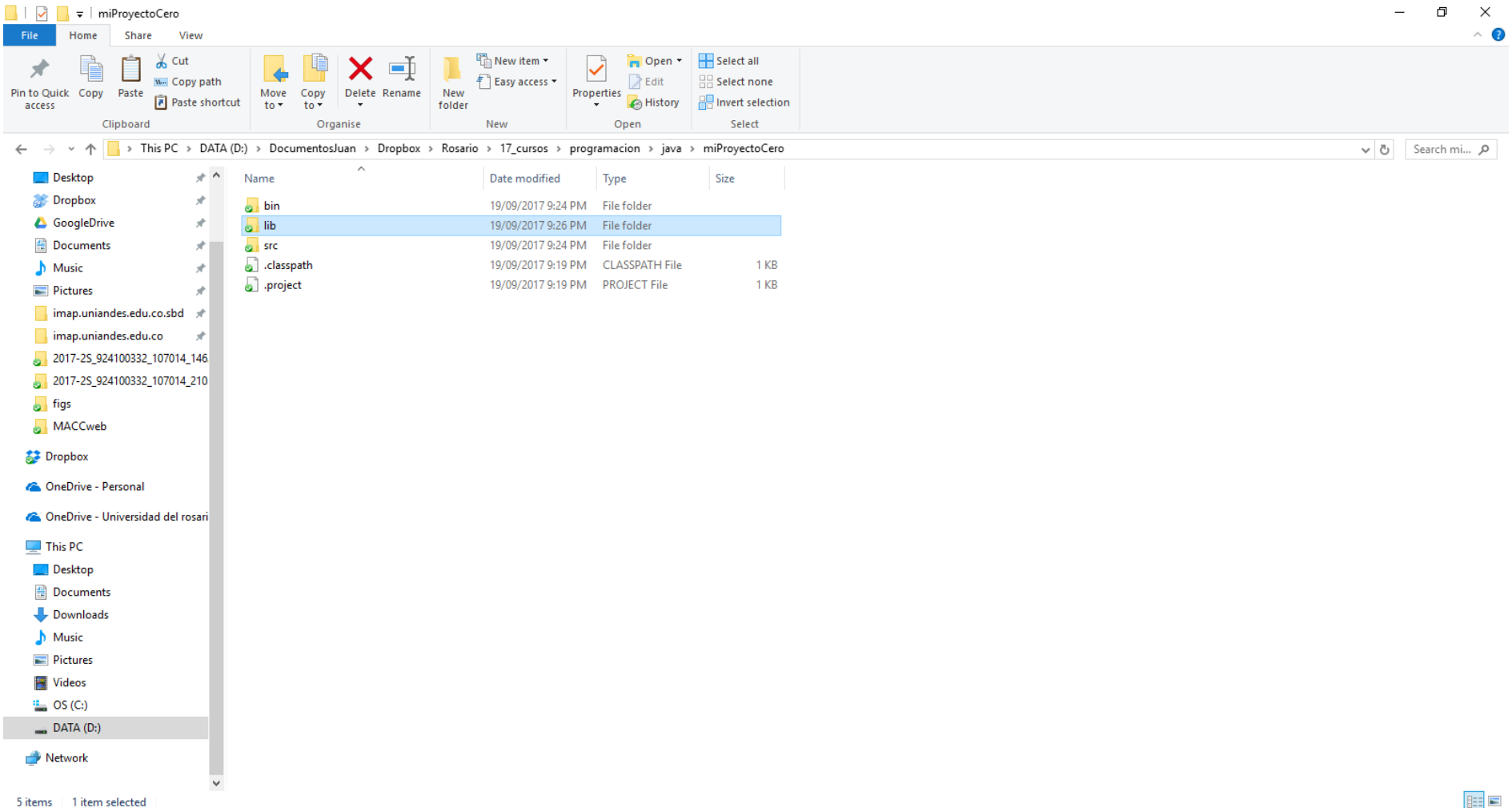
Creamos una nueva clase



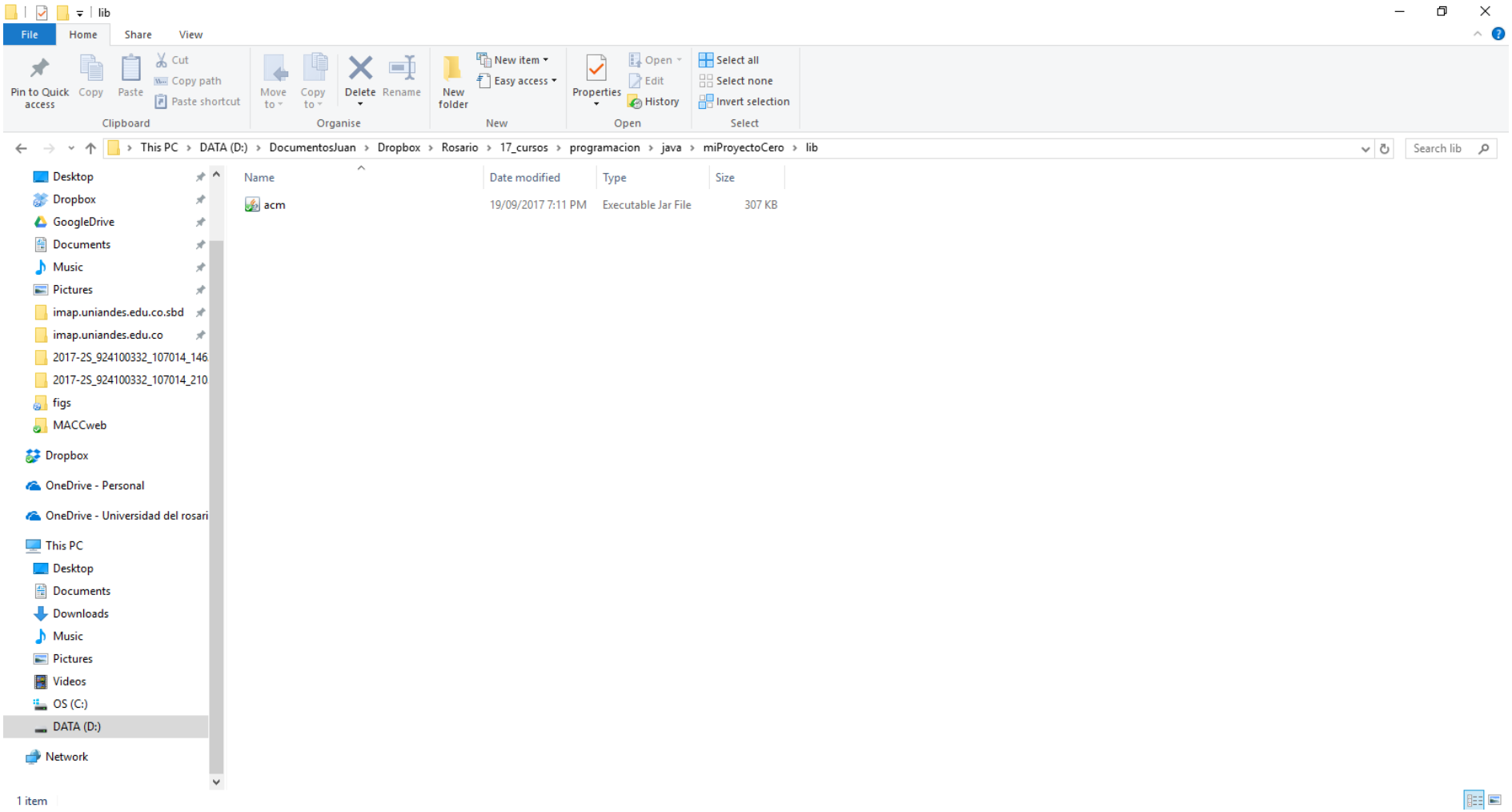
Agregamos y enlazamos librería `acm.jar`



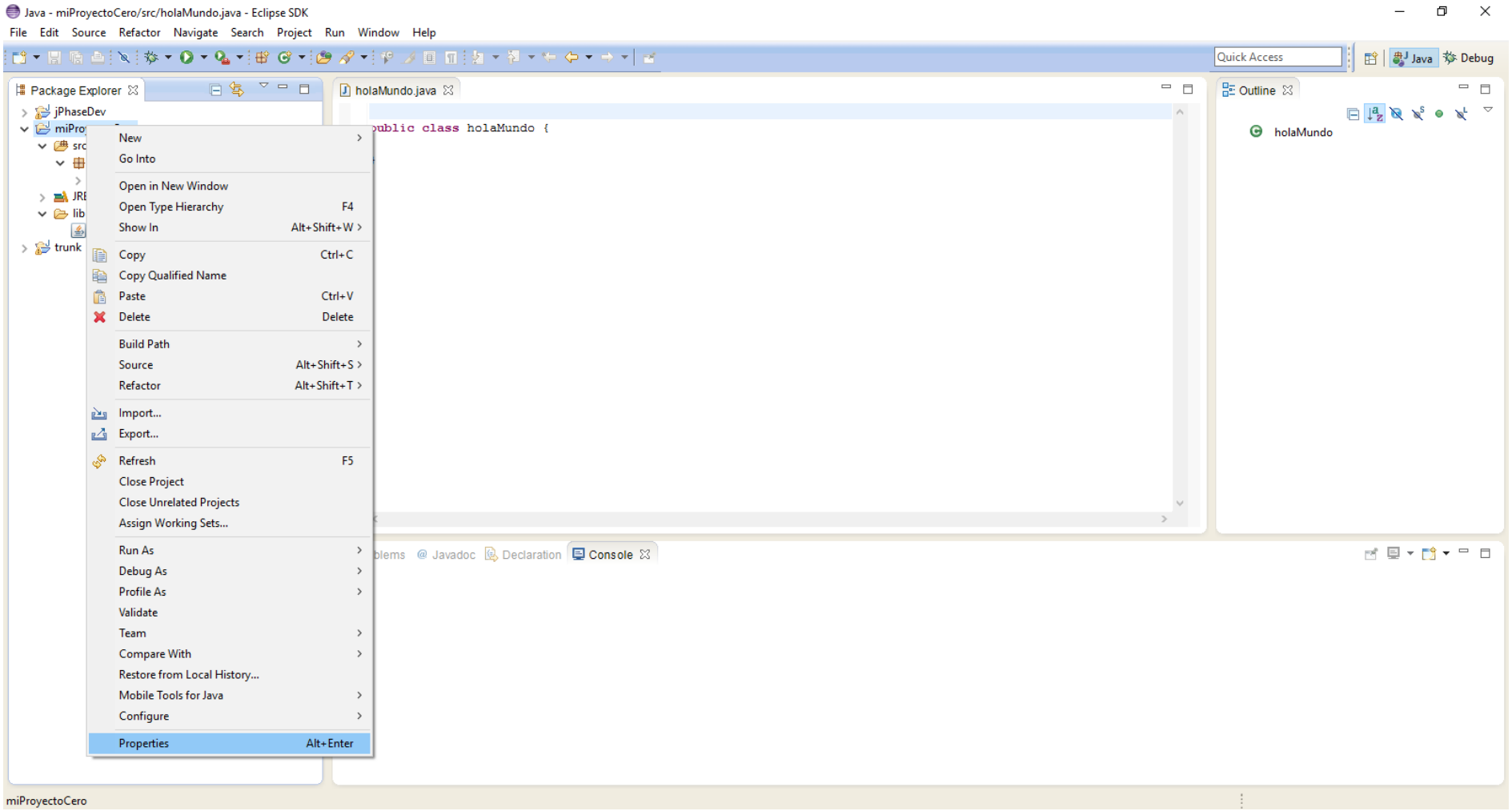
Agregamos y enlazamos librería `acm.jar`



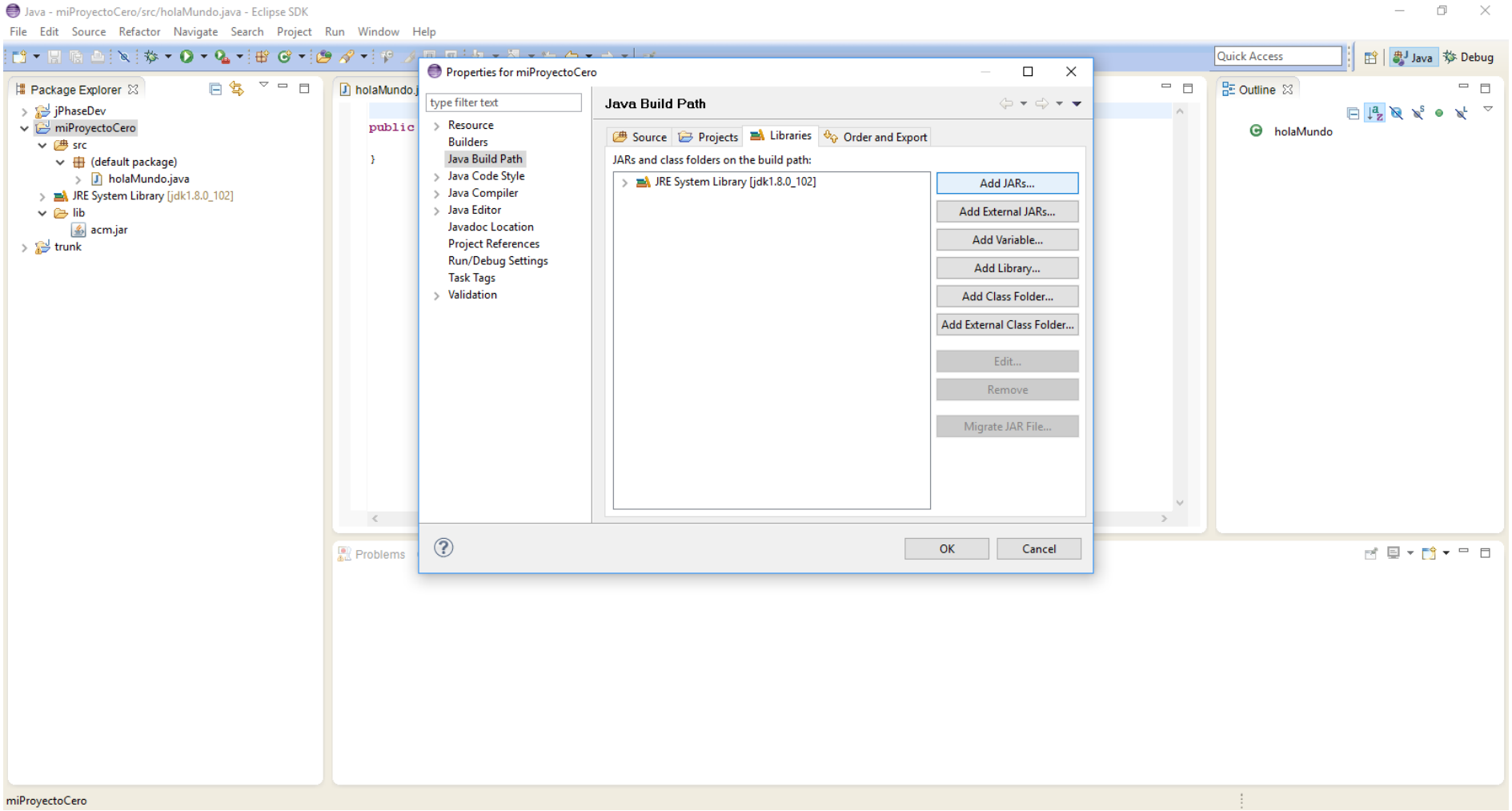
Agregamos y enlazamos librería `acm.jar`



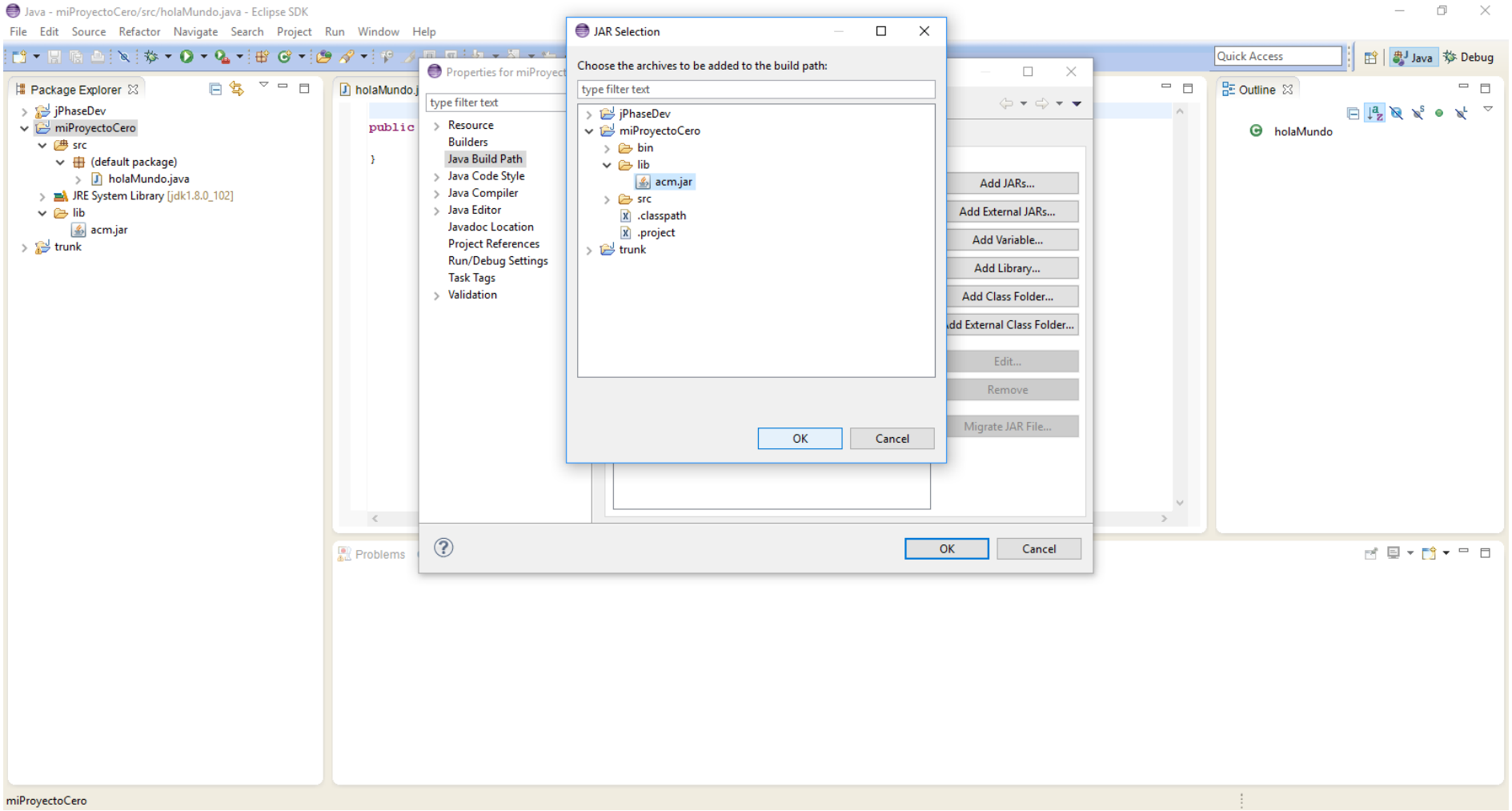
Agregamos y enlazamos librería `acm.jar`



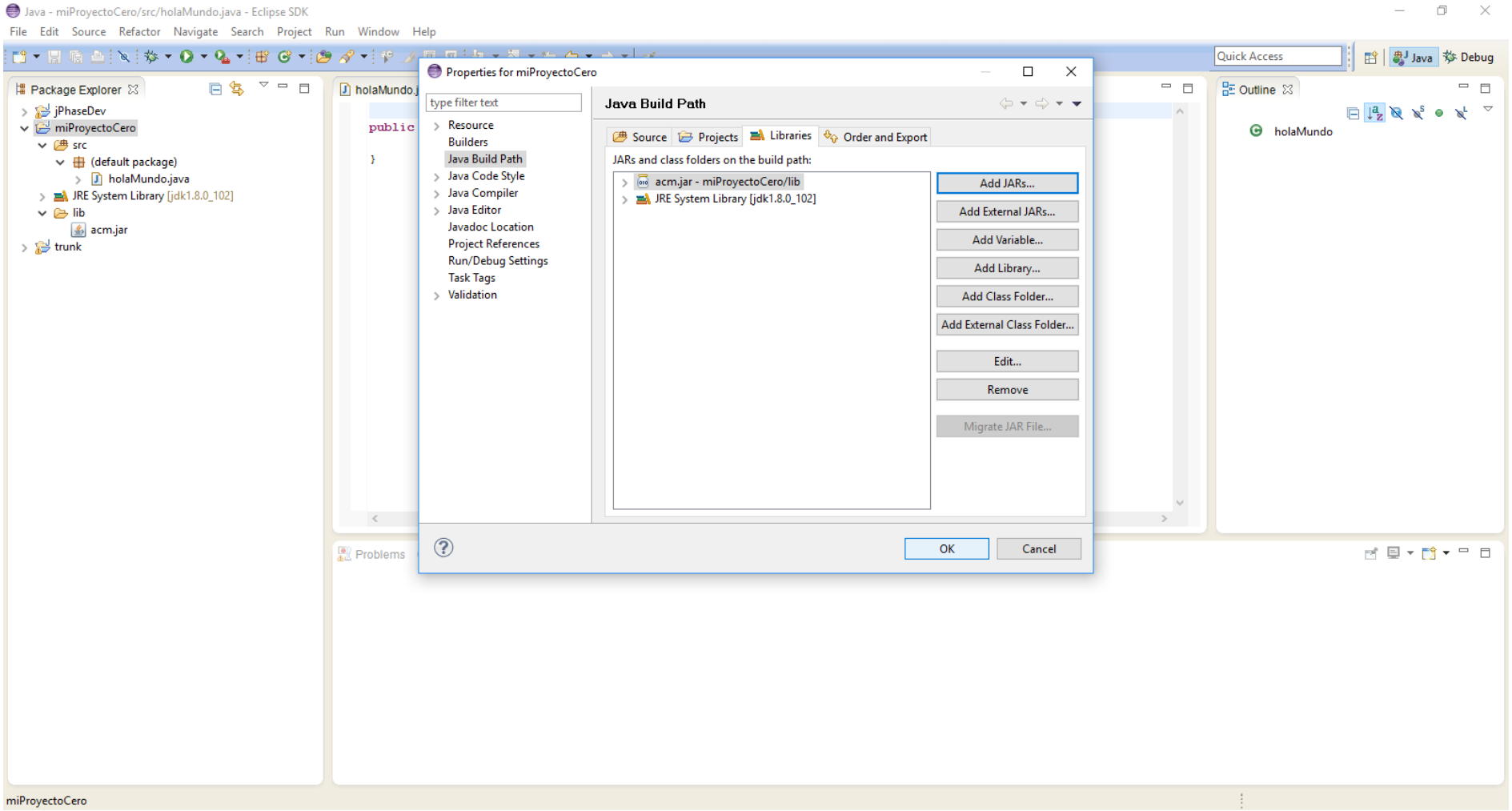
Agregamos y enlazamos librería `acm.jar`



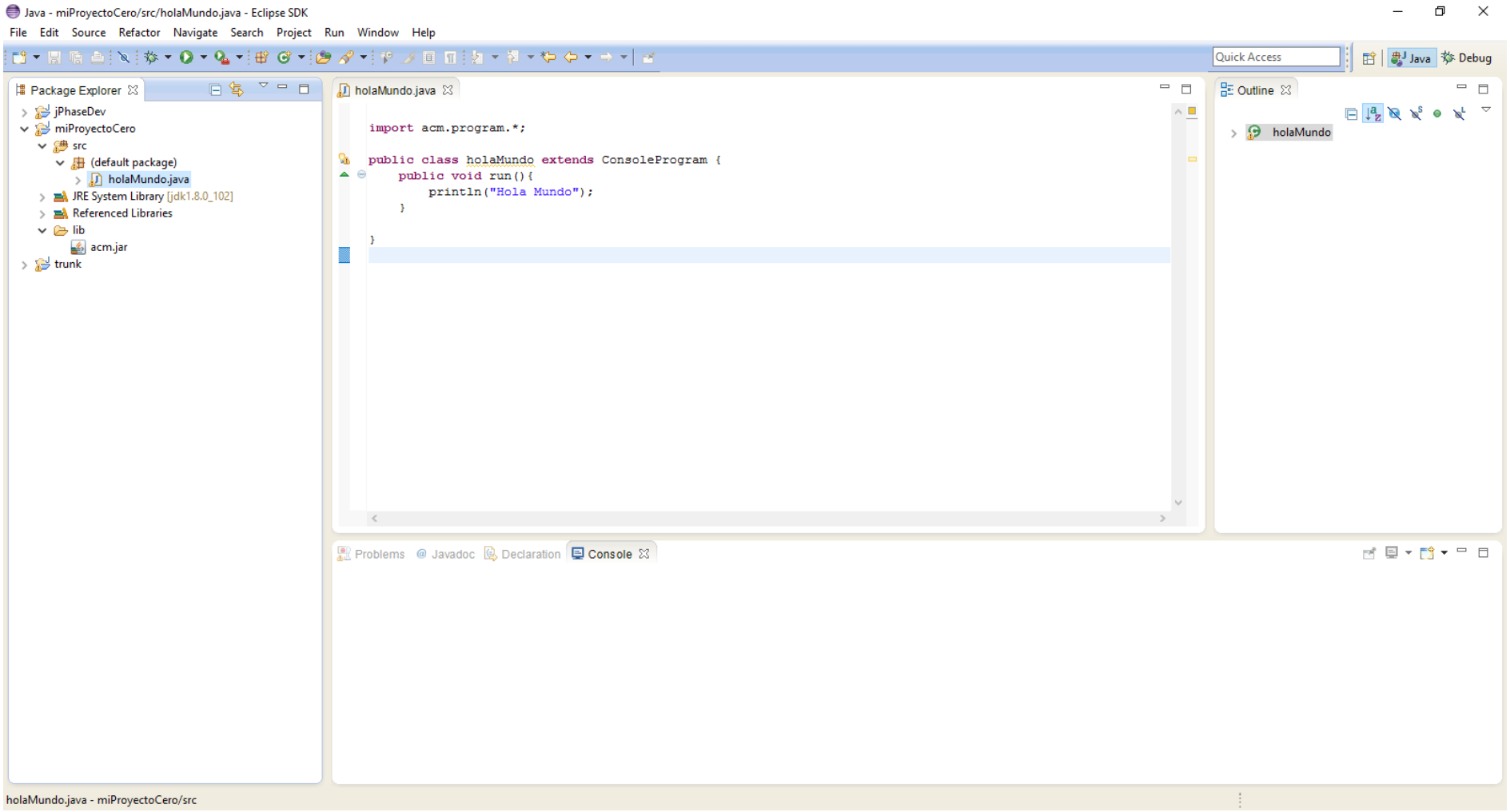
Agregamos y enlazamos librería `acm.jar`



Agregamos y enlazamos librería `acm.jar`



Agregamos y enlazamos librería `acm.jar`

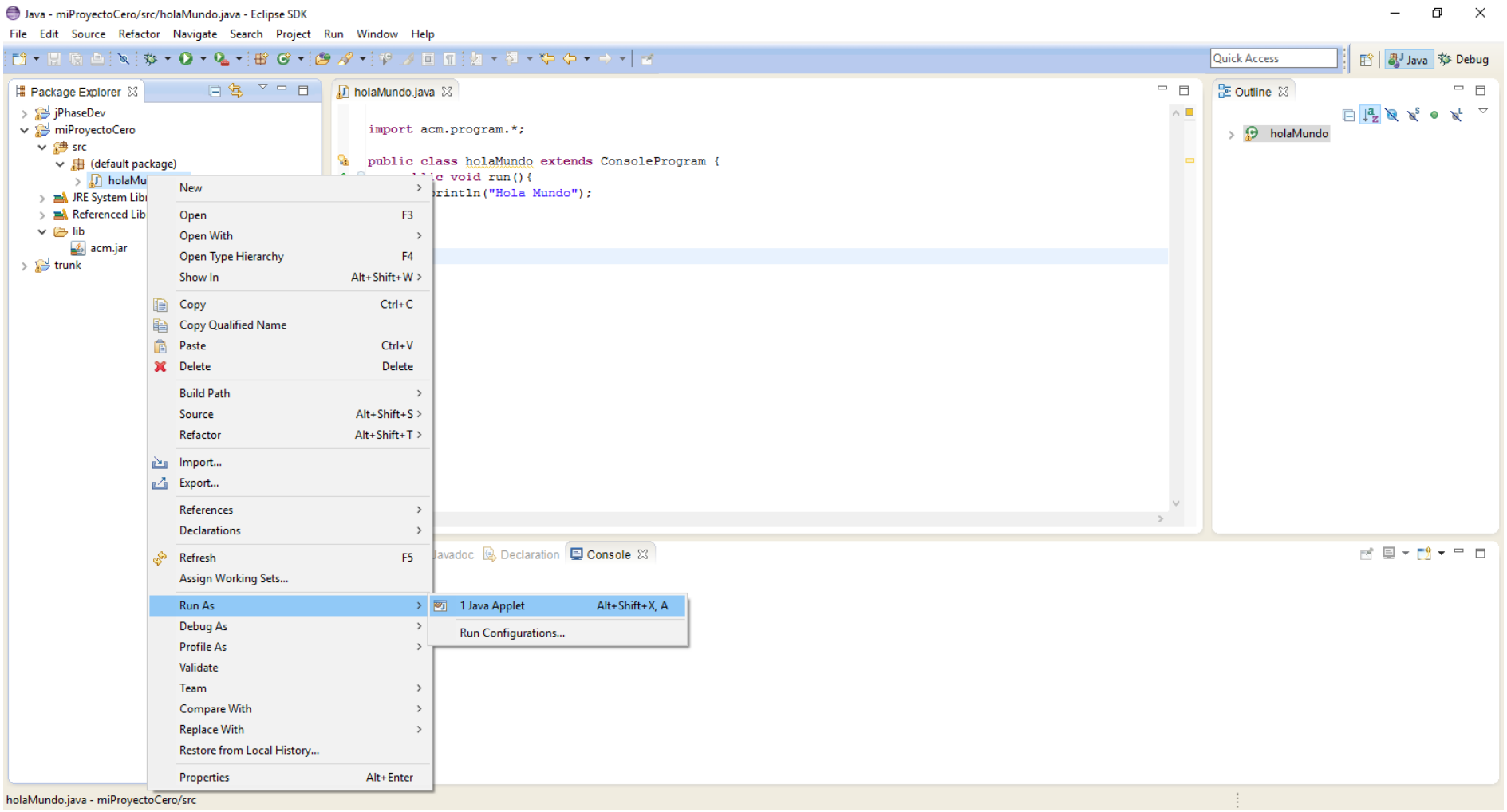


Escribimos nuestro código

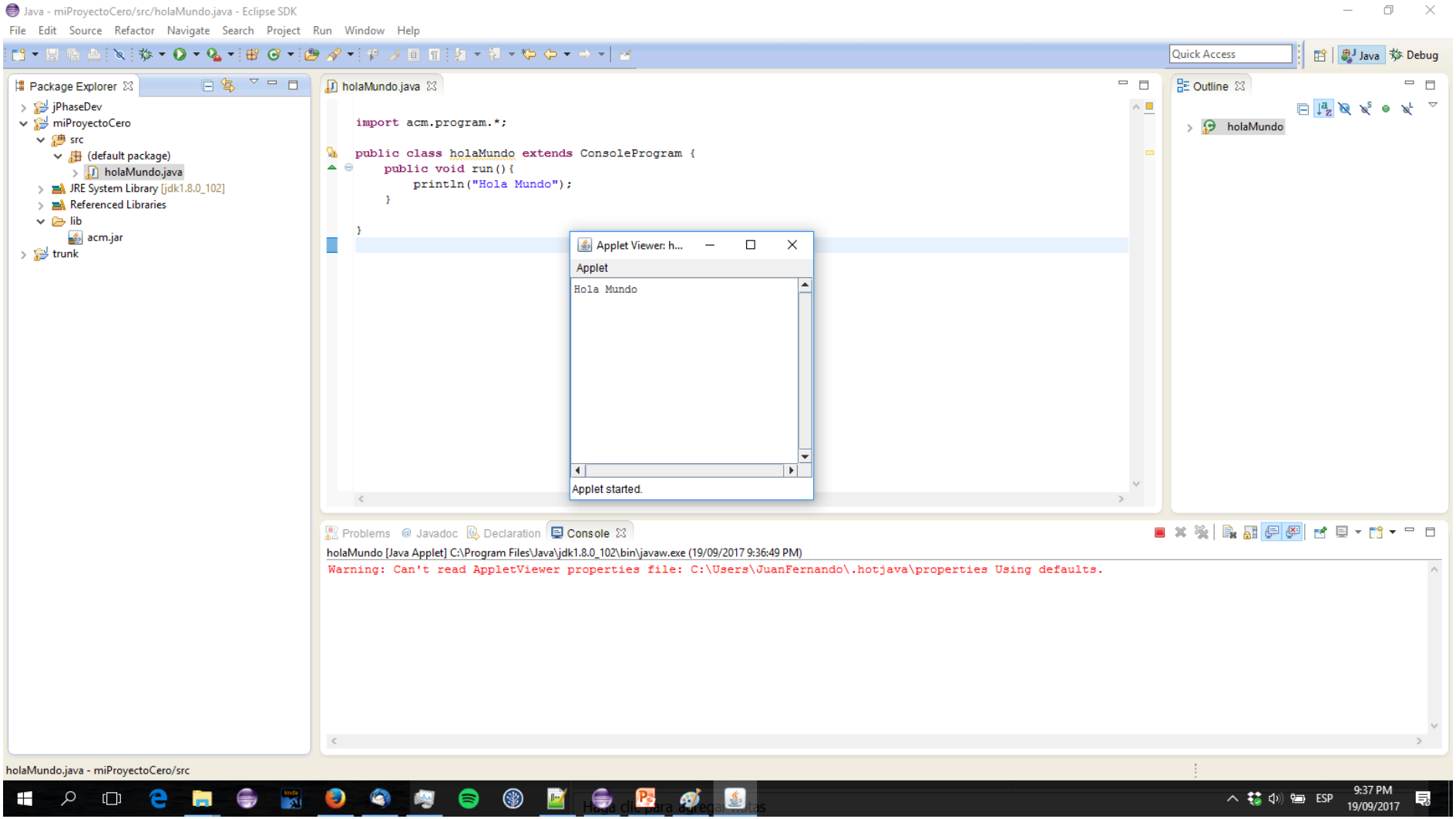
```
import acm.program.*;

public class holaMundo extends ConsoleProgram {
 public void run(){
        println("Hola Mundo");
 }
}
```

Ejecutamos nuestras clase como Applet



Creamos un nuevo proyecto en Java



Algunos ejemplos más

Suma de enteros

```
import acm.program.*;

public class sumaDosNumeros extends ConsoleProgram{
 public void run(){
  int n1 = 2695;
  int n2 = 1042;
  int suma = n1 + n2;
  println("La suma de "+n1+" + "+n2+" es igual a "+suma);
 }
}
```

Leer de usuario dos enteros y sumarlos

```
import acm.program.*;

public class recibeSumaDosNumeros extends ConsoleProgram{
 public void run(){
  println("Este programa suma dos números");
  int n1 = readInt("Ingrese el primer número ");
  int n2 = readInt("Ingrese el segundo número ");
  int suma = n1 + n2;
  println("La suma de "+n1+" + "+n2+" es igual a "+suma);
 }
}
```

Leer de usuario dos flotantes y promediarlos

```
import acm.program.*;

public class recibePromedioDosNumeros extends ConsoleProgram
    {
 public void run(){
  println("Este programa suma dos números");
  double n1 = readDouble("Ingrese el primer número ");
  double n2 = readDouble("Ingrese el segundo número ");
  double promedio = (n1 + n2)/2;
  println("El promedio de "+n1+" y "+n2+" es igual a "+
    promedio);
 }
}
```

Tipos de datos primitivos

Enteros

Tipo	Dominio
`byte`	Entero de 8 bits entre -128 (-2^7) y 127 ($2^7 - 1$)
`short`	Entero de 16 bits entre -32768 (-2^{15}) y 32767 ($2^{15} - 1$)
`int`	Entero de 32 bits entre -2,147,483,648 (-2^{31}) a 2,147,483,647 ($2^{31} - 1$)
`long`	Entero de 64 bits entre -9,223,372,036,854,775,808 (-2^{63}) a 9,223,372,036,854,775,807 ($2^{63} - 1$)

Operaciones con Enteros

Operadores Aritméticos:

- $+, -, *, /, \%$

Operadores Relacionales:

- $==$ igual
- $!=$ diferente
- $<$ menor que
- $<=$ menor o igual que
- $>$ mayor que
- $>=$ mayor o igual que

Punto flotante

Tipo	Dominio
`float`	Número de punto flotante de 32 bits entre $\pm 1{,}4 \times 10^{-45}$ y $3{,}4028235 \times 10^{38}$
`double`	Número de punto flotante de 64 bits entre $\pm 4{,}49 \times 10^{-322}$ y $1{,}7976931348623157 \times 10^{308}$

Los mismos operadores aritméticos (excepto %) y relacionales

Caracteres y lógicos

Tipo	Dominio
`char`	Caracter de 16 bits en Unicode
`boolean`	Número de punto flotante de 64 bits entre $\pm 4{,}49 \times 10^{-322}$ y $1{,}7976931348623157 \times 10^{308}$

Operadores:

- `char`: operadores relacionales
- `boolean`:
  - `&&`: y
  - `||`: o
  - `!`: no

Incrementos

Incrementos

Expresiones idénticas:

```
x = x + 1;
x += 1;
x ++;
```

Expresiones idénticas:

```
x = x - 1;
x -= 1;
x --;
```

Expresiones idénticas:

```
x = x + 10;
x += 10;
```

Variables y Constantes

Variables y Constantes

```
package miProyecto;
import acm.program.*;

public class areaCirculo extends ConsoleProgram{
 public void run(){
  final double PI = 3.14159265358979323846;
  double radio = readDouble("Ingrese el radio del círculo");
  double area = PI*radio*radio;
  println("El área del cículo de radio "+radio+" es "+area);
 }
}
```

Variables y Constantes

```
package miProyecto ;
import acm. program .*;

public class AreaTriangulo extends ConsoleProgram {
 public void run (){
  double base;
  double altura;
  base = readDouble ("␣Ingrese␣la␣base␣de␣triángulo␣");
  altura = readDouble ("␣Ingrese␣la␣altura␣de␣triángulo␣");
  double area = base * altura / 2 ;
  println ("El␣␣área␣␣del␣␣␣triángulo␣de␣base␣"+ base +"␣y␣
    altura␣"+ altura +"␣es␣"+ area );
 }
}
```

Variables y Constantes

Declarar (definir) una variable:

```
double base;
```

Inicializar una variable:

```
base = 100;
```

Declarar e inicializar una variable:

```
double base = 100;
```

Instrucciones de Control - Condicionales

Condicionales: **if**

```
if(expresión1){
 instrucción;
 instrucción;
 instrucción;
}else if(expresión2){
 instrucción;
 instrucción;
 instrucción;
}else{
 instrucción;
 instrucción;
}
```

Condicionales: **if**

```
package miProyecto;
import acm.program.ConsoleProgram;

public class NumeroNegativo  extends ConsoleProgram {
 public void run (){
 double numero = readDouble("Ingrese un número");
 if(numero < 0)
  println("El número es negativo");
 println("----------------");
 }
}
```

Condicionales: **if**

```
package miProyecto;
import acm.program.ConsoleProgram;

public class NumeroPositivoNegativo  extends ConsoleProgram
    {
 public void run (){
  double numero = readDouble("Ingrese un número");
  if(numero < 0){
   println("El número es negativo");
  }else if(numero >0){
   println("El número es positivo");
  }else{
   println("El número es cero");
  }
 }
}
```

Condicionales: ?:

```
(expresión) ? instrucción1 : instrucción2;
```

Condicionales: ?:

```
package miProyecto;
import acm.program.ConsoleProgram;

public class Maximo  extends ConsoleProgram {
 public void run (){
  double x = readDouble("Ingrese el primer número ");
  double y = readDouble("Ingrese el segundo número ");
  println("Versión 1:");
  double max;
  if(x > y){
   max = x;
  }else{
   max = y;
  }
  println("El máximo entre los dos números es "+max);
  println("-------------------------------------");
  println("Versión 2:");
  double max2 = x > y ? x : y;
  println("El máximo entre los dos números es "+max2);
 }
```

Condicionales: ?:

```
package miProyecto;
import acm.program.ConsoleProgram;

public class NumeroPar  extends ConsoleProgram {
 public void run (){
  double x = readInt("Ingrese un entero ");
  boolean par = x % 2==0 ? true : false;
  println("El número "+ (par?"":"no ")+"es par");
 }
}
```

Instrucciones de Control - Switch

Instrucción **switch**

```
switch (expresión){
 case c1:
  instrucción;
  instrucción;
  break;
 case c2:
  instrucción;
        instrucción;
  break;
 case default:
  instrucción;
  instrucción;
  break;
}
```

Instrucción switch

```
package miProyecto;
import acm.program.ConsoleProgram;

public class NumeroPar2  extends ConsoleProgram {
 public void run (){
  int numero = readInt("Ingrese un entero ");
  int residuo = numero % 2;
  switch (residuo){
   case 0:
    println("El número es par");
    break;
   case 1:
    println("El número es impar");
    break;
  }
 }
}
```

Instrucción switch

```
package miProyecto;
import acm.program.ConsoleProgram;

public class OtroNumero  extends ConsoleProgram {
 public void run (){
  int numero = readInt("Ingrese un entero ");
  switch (numero){
   case 0:
    println("El número es 0");
    break;
   case 10:
    println("El número es 10");
    break;
   default:
    println("El número no es 0 ni 100");
    break;
  }
 }
}
```

Instrucciones de Control - Ciclos

Ciclo **while**

```
while(condición){
  instruccion1;
  instruccion2;
  instruccion3;
}
```

Ciclo **while**

```
package miProyecto;
import acm.program.ConsoleProgram;

public class Countdown extends ConsoleProgram {
 public void run (){
  int i = 10;
  while(i > 0){
   println("El número actual es "+ i);
   i = i - 1;
  }
  }
}
```

Ciclo **for**

```
for( inicialización; condición; paso){
  instrucción1;
  instrucción2;
}
```

Ciclo **for**

```
package miProyecto;
import acm.program.ConsoleProgram;

public class Countdown2 extends ConsoleProgram {
 public void run (){
  for(int i = 10; i > 0; i--){
   println("El número actual es "+ i);
  }
 }
}
```

Ciclo **for**

```
package miProyecto;
import acm.program.ConsoleProgram;

public class CountBy10 extends ConsoleProgram {
 public void run (){
  for(int i = 0; i <=100; i+=10){
   println("El número actual es "+ i);
  }
 }
}
```

Métodos

Métodos

Equivalentes a las funciones en Python

Modularizar código

Encabezado:
`visibilidad tipo` **`nombre`**`(parámetros)`

Retornar valores:
`return`

Métodos propios (1 de 2)

```
package miProyecto;
import acm.program.ConsoleProgram;

public class NumeroPar3  extends ConsoleProgram {
 public void run (){
  int numero = readInt("Ingrese un entero ");
  if(esPar(numero)){
   println("El número es par");
  }else{
   println("El número es impar");
  }
 }
```

Métodos propios (2 de 2)

```
  private boolean esPar(int num){
   if(num % 2 == 0){
    return true;
   }else{
    return false;
   }
  }
}
```

Métodos incluidos y propios (1 de 2)

```
package miProyecto;
import acm.program.ConsoleProgram;

public class Distancia  extends ConsoleProgram {
 public void run (){
  double x1 = readDouble("Ingrese la coordenada x del primer
    número ");
  double y1 = readDouble("Ingrese la coordenada y del primer
    número ");
  double x2 = readDouble("Ingrese la coordenada x del
    segundo número ");
  double y2 = readDouble("Ingrese la coordenada y del
    segundo número ");

  double dist = calcularDistancia(x1,y1,x2,y2);
  println("La distancia euclidiana entre estos dos puntos es
    "+dist);
 }
```

Métodos incluidos y propios (2 de 2)

```
  private double calcularDistancia(double x1, double y1,
     double x2, double y2){
   double dx = Math.abs(x1-x2);
   double dy = Math.abs(y1-y2);
   double distancia = Math.sqrt(Math.pow(dx, 2) + Math.pow(dy
     , 2) );
   return distancia;
  }
}
```