Programas Básicos

- Conceptos Básicos
  - Clases
    - Todo programa en Java esta formado por una o más clases.
    - Los ficheros son ficheros texto que contienen una o varias clases.
    - El nombre del fichero fuente debe coincidir con la clase que contiene con extensión `.java`
    - La clase es una plantilla donde se describen los datos y métodos que van a tener cada una de sus instancias.
    - No se permiten métodos o variables globales.
    - Existen métodos y variables estáticos comunes a todos los objetos de una clase.
    - Declaración de una clase:
      - sintaxis: `[<tipo_de_acceso>] class <nombre_de_clase>`
      - ejemplos: `public class ejemplo2_1{...}`

        `class Segmento {...}`

Programas Básicos

- Conceptos Básicos
  - Objetos
    - Un objeto es un elemento cuyo tipo corresponde a una determinada clase
    - Todo objeto debe ser declarado para indicar a que tipo de clase pertenece
      - sintaxis: <nombre_de_clase> <nombre_del_objeto>;
      - ejemplo: `Segmento seg0;`
    - Todo objeto debe ser instanciado para reservarle memoria y poder utilizar sus variables y métodos no estáticos.
      - sintaxis: <nombre_del_objeto> = new <constructor_de_clase>
      - ejemplo: `seg0 = new Segmento();`
    - Se puede declarar e instanciar un objeto a la vez
      - sintaxis:<nombre_de_clase> <nombre_del_objeto> = new <constructor_de_clase>
      - ejemplo: `Segmento seg0 = new Segmento();`

Programas Básicos

- Conceptos Básicos
  - Metodo main()
    - Primer metodo que invoca el interprete.
    - En un fichero sólo puede haber una clase que contenga la función main y será esta la que de nombre al fichero.
    - Su declaración es:

```
public static void main(String args[]){...}
```

Programas Básicos

- Conceptos Básicos
  - Ejemplo2_1:

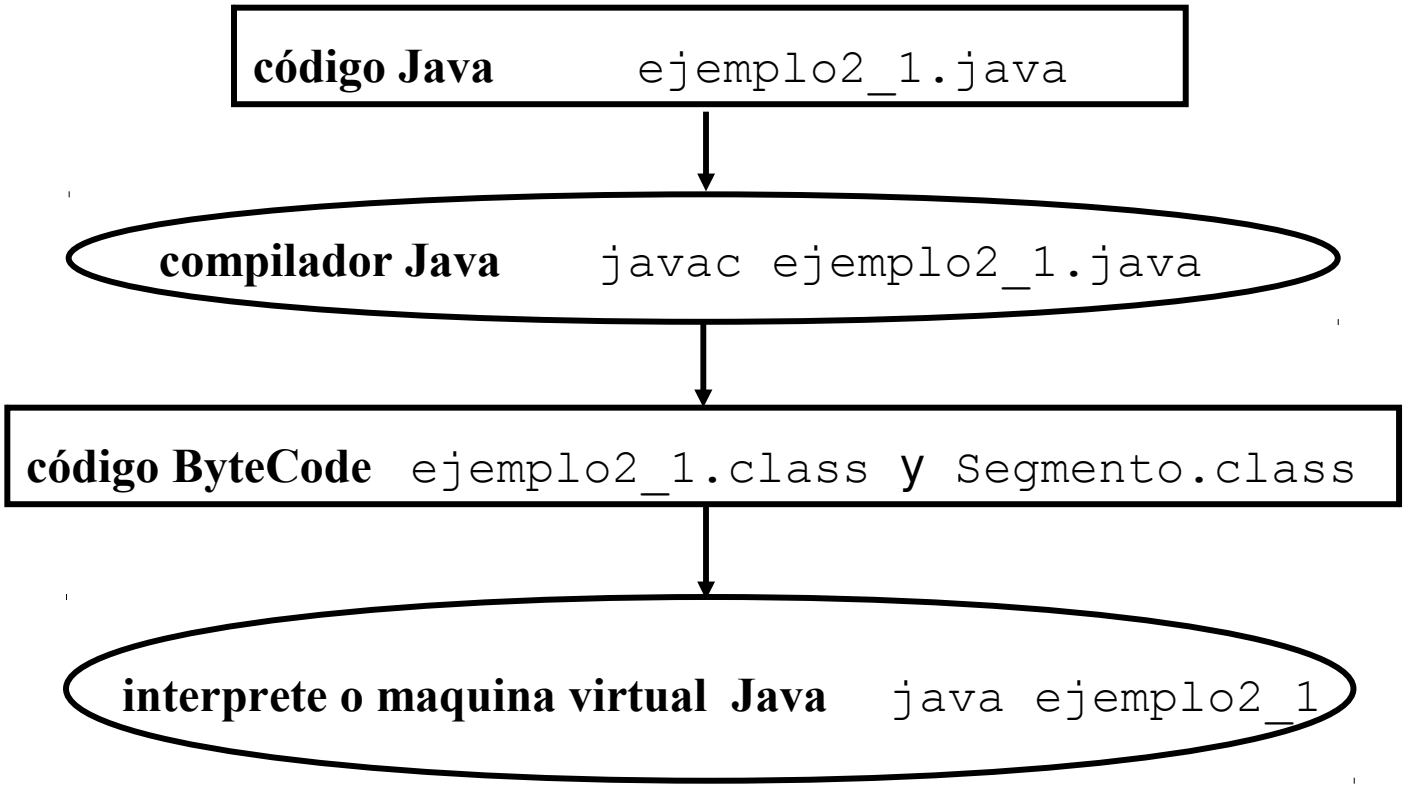
```
import java.lang.*; /* No haría falta ya que se carga por defecto */
class Segmento {
    //Variables
    public static int numero;
    public int longitud;
    //Constructores
    Segmento(){
          longitud = 0;
    }
    Segmento( int l){
          longitud = l;
    }
    //Métodos
    public static void incrementar(){
          numero++;
    }
    public void sumarLongitud(int l){
          longitud+=l;         /*longitud = logitud + l;*/
    }
    public  void sumarSegmento(Segmento s) {
          longitud += s.longitud;        /* longitud = logintud s.longitud */
    }
}
```

Programas Básicos

```
public class ejemplo2_1 {
    public static void main(String[] args) {
          Segmento seg0; /* Declaración del objeto*/

          Segmento.numero++;
          seg0 = new Segmento(); /* Instanciación */
          System.out.println("Número de segmentos creados = "+ seg0.numero);
          System.out.println("Longitud = "+ seg0.longitud);

          Segmento seg1 = new Segmento(10); /* Declaración e instanciación */
          seg1.numero++;
          System.out.println("Número de segmentos creados = "+ seg0.numero);
          System.out.println("Longitud = "+ seg1.longitud);

          Segmento.incrementar();
          seg0.incrementar();
          seg1.incrementar();
           System.out.println("El valor de numero para los objetos seg0 y seg1 es "
          +seg0.numero+" y "+seg1.numero +" respectivamente");

          seg0.sumarLongitud(5);
          seg1.sumarSegmento(seg0);
          System.out.println("Longitud de seg0 = "+seg0.longitud);
          System.out.println("Longitud de seg1 = "+seg1.longitud);
    }
}
```

Programas Básicos

- Compilación y ejecución de un programa Java



Programas Básicos

- Programa Holamundo.java

```
import  java.lang.System;

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

- Programa Fecha.java

```
import java.util.Date;

/** Muestra la fecha actual */
class Fecha {

    public static void main( String args[] ) {
        // Obtiene la fecha del sistema
        Date hoy = new Date();

        // La imprime en la consola
        System.out.println( hoy );
        }
    }
```

Programas Básicos

- Programa Argurmentos.java

```
class Argumentos {
        public static void main( String args[] ) {
            for( int i=0; i < args.length; i++ )
                System.out.println( args[i] );
            }
        }
```

Programas Básicos

- Programa Nombre.java

```
import java.io.*;

/** Pide que escribas tu nombre y luego te lo muestra */
public class Nombre {
    public static void main( String args[] ) {

    StringBuffer nombreTmp = new StringBuffer();        // un String pero dinámico
    int c;

    System.out.print("Escribe tu nombre: ");
    try {
       while ( (c = System.in.read()) != '\n') {
       nombreTmp.append((char)c);
     }
     }catch(IOException e) {
       System.out.println("Se ha producido el error: " + e);
       return;
      }

    String nombre = new String(nombreTmp.toString());
    System.out.println("Tu nombre es: " + nombre);
    }
}
```

Programas Básicos

- Programa Linea.java

```
import java.io.*;

/** Pide que escribas una linea y luego te la muestra */
public class Linea
{

    public static void main( String args[] ) {
    String nuevalinea;
    BufferedReader entradateclado = new BufferedReader(new InputStreamReader(System.in));

    System.out.print("Escribe una linea: ");

    try {
       nuevalinea = entradateclado.readLine();
    }
    catch(IOException e)  {
       System.out.println("Se ha producido el error: " + e);
       return;
      }

     System.out.println("La linea escrita es: " + nuevalinea);
     }
}
```

Programas Básicos

- Tratamiento de Ficheros
  - Clase de comprobación (File)
    - Chequea las propiedades del fichero como: nombre, ruta, tamaño, tipo,...etc.
    - Se necesita importar el paquete `java.io (import java.io.*;)` donde está la clase File.
    - Se instancia con:

```
File <NombreInstancia> = new File("<Ruta>/<NombreFichero>");
```

    - Métodos de la clase File

Método	Significado
String getName()	Devuelve el nombre del fichero.
String getPath()	Devuelve el path o ruta completa.
boolean renameTo(NuevoNombre)	Renombra el fichero.
boolean exists()	Comprueba si el fichero existe.
boolean canWrite()	Comprueba si el fichero se puede escribir.
boolean canRead()	Comprueba si el fichero se puede leer.
boolean isFile()	Comprueba si es un fichero.
boolean isDirectory()	Comprueba si es un directorio.
long length()	Devuelve el tamaño del fichero.
boolean mkdir()	Crea un directorio.
boolean delete()	Borra el fichero.
String[] list()	Devuelve la lista de ficheros de un directorio.

Programas Básicos

- Tratamiento de Ficheros
  - Clase de comprobación (File)
    - Ejemplo CompruebaFicheros.java

```
import java.io.*;

/** Informacion de Ficheros pasados por argumentos en la linea comandos **/
class CompruebaFicheros {

 public static void main( String args[] ) throws IOException {
    // Se comprueba que nos han indicado algún fichero
    if( args.length > 0 ) {

      // Vamos comprobando cada uno de los ficheros que se hayan pasado
      for( int i=0; i < args.length; i++ ) {
        // Se crea un objeto File para tener una referencia al fichero
        File f = new File( args[i] );
        // Se presenta el nombre y directorio donde se encuentra
        System.out.println( "Nombre: "+f.getName() );
        System.out.println( "Camino: "+f.getPath() );
        // Si el fichero existe se presentan los permisos de lectura y
        // escritura y su longitud en bytes
```

Programas Básicos

- Tratamiento de Ficheros
  - Clase de comprobación (File)
    - Ejemplo CompruebaFicheros.java

```
if( f.exists() ) {
          System.out.print( "Fichero existente " );
          System.out.print( (f.canRead() ? " y se puede Leer" : "" ) );
          System.out.print( (f.canWrite() ? " y se puese Escribir" : "" ));
          System.out.println( "La longitud del fichero es de "+
            f.length()+" bytes" );
          }
        else {
          System.out.println( "El fichero no existe." );
          }
        }
      }
    else {
      System.out.println( "Debe indicar un fichero." );
      }
    }
```

Programas Básicos

- Tratamiento de Ficheros
  - Lectura de ficheros
    - FileInputStream
      - FileInputStream está incluida en el paquete `java.io`
      - Instanciación a partir de un `String`

        ```
        FileInputStream <Fichero> = new FileInputStream("<NombreFichero>");
        ```
      - Instanciación a partir de un `File`

        ```
        FileInputStream <Fichero> = new FileInputStream(<ObjetoFile>);
        ```
      - Métodos de lectura
        - `int read() -> lee un byte o -1 al final del stream`
        - `int read(byte b[]) -> llena b y devuelve el número de bytes leidos`
      - Método para cerrar el fichero
        - void close()
    - BufferedReader : más eficiente para leer caracteres, lineas, arrays, etc.
      - BufferedReader está incluida en el paqute java.io
      - Instanciación

      ```
      BufferedReader <Fichero> = new BufferedReader(new FileReader("<NombreFichero>"));
      ```
      - Métodos de lectura
        - `int read() -> lee un caracter`
        - `int read(char[] cbuf, int off, int len) -> lee en una porción de un array`
        - `int readLine() -> lee una linea de texto`
      - Método para cerrar el fichero
        - void close()

Programas Básicos

- Tratamiento de Ficheros
  - Escritura de ficheros
    - FileOutputStream
      - FileOutputStream está incluida en el paquete `java.io`
      - Instanciación a partir de un `String`

```
FileOutputStream <Fichero> = new FileOutputStream("<NombreFichero>");
```

      - Instanciación a partir de un `File`

```
FileOutputStream <Fichero> = new FileOutputStream(<ObjetoFile>);
```

      - Métodos de escritura
        - `int write(int b) -> escribe un byte`
        - `int write(byte b[]) -> escribe un array de byte`
      - Método para cerrar el fichero
        - void close()
    - BufferedWriter : más eficiente para escribir caracteres, lineas, arrays, etc.
      - BufferedWriter está incluida en el paqute java.io
      - Instanciación

```
BufferedWriter <Fichero> = new BufferedWriter(new FileWriter("<NombreFichero>"));
```

      - Métodos de lectura
        - `int write(int ) -> escribe un caracter`
        - `int write(char[] cbuf, int off, int len) -> escribe una porción de un array`
        - `int write(String, int off, int len) -> escribe una porción de una cadena`
        - `void newLine() -> escribe un separador de línea`
      - Método para cerrar el fichero
        - void close()

Programas Básicos

- Ejemplo LeeEscribeFichero.java

```
import java.io.*;
/** Escritura y lectura en fichero */
public class LeeEscribeFichero {

 public static void main(String[] args)
 {
       String cadena = "Primera linea.\nSegunda linea.\nUltima linea.";
       String tmp = null;

       try {
          BufferedWriter fout = new BufferedWriter(new FileWriter("fichero.txt") );
          fout.write(cadena,0,cadena.length());
          fout.close();

          //File f=new File(Fichero);
          BufferedReader fin = new BufferedReader(new FileReader("fichero.txt") );

          System.out.println("El contenido del fichero es:");
          while ( (tmp = fin.readLine()) != null )
           System.out.println(tmp);
        }
        catch(IOException e){
          System.out.println("Error al escribir");
        }
   }
}
```