

# Programación Orientada a Objetos

## Entrada-Salida

CEIS

2024-2

# Agenda

## Conceptos

- En general

- En java

## Objetos

- Mecanismo

- Batalla Naval

## Texto-Archivos

- Mecanismo

- Batalla Naval

## Caracteres-Estandar

- Mecanismo

- Batalla Naval

- Final

# Agenda

## Conceptos

- En general

- En java

## Objetos

- Mecanismo

- Batalla Naval

## Texto-Archivos

- Mecanismo

- Batalla Naval

## Caracteres-Estandar

- Mecanismo

- Batalla Naval

- Final

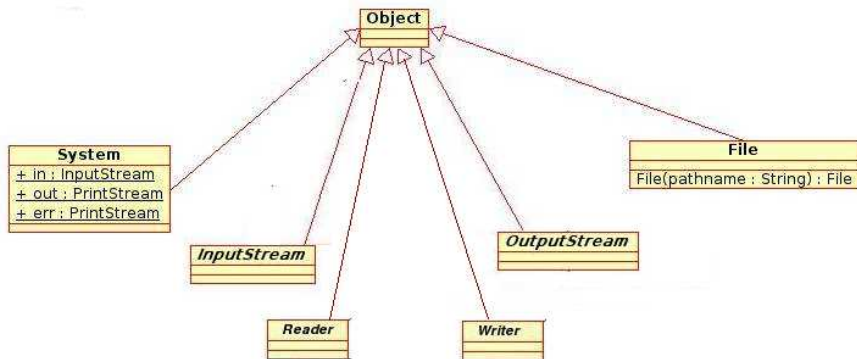
# Entrada-Salida

¿Desde dónde? ¿Hacia dónde?

¿Cómo?

- ▶ Como máquina, ¿qué es lo real?
- ▶ Como humanos, ¿qué queremos ver?
- ▶ Como lenguaje O.O, ¿qué querría ver?

# Entrada-Salida



# Entrada-Salida

java.io

## Class IOException

```
java.lang.Object
├ java.lang.Throwable
│   └ java.lang.Exception
│       └ java.io.IOException
```

All Implemented Interfaces:

[Serializable](#)

Direct Known Subclasses:

[ChangedCharSetException](#), [CharacterCodingException](#), [CharConversionException](#),  
[ClosedChannelException](#), [EOFException](#), [FileLockInterruptedException](#), [FileNotFoundException](#),  
[HttpRetryException](#), [IOException](#), [InterruptedIOException](#), [InvalidPropertiesFormatException](#),  
[JMXProviderException](#), [JMXServerErrorException](#), [MalformedURLException](#),  
[ObjectStreamException](#), [ProtocolException](#), [RemoteException](#), [SaslException](#), [SocketException](#),  
[SSLException](#), [SyncFailedException](#), [UnknownHostException](#), [UnknownServiceException](#),  
[UnsupportedEncodingException](#), [UTFDataFormatException](#), [ZipException](#)

# Agenda

## Conceptos

En general

En java

## Objetos

Mecanismo

Batalla Naval

## Texto-Archivos

Mecanismo

Batalla Naval

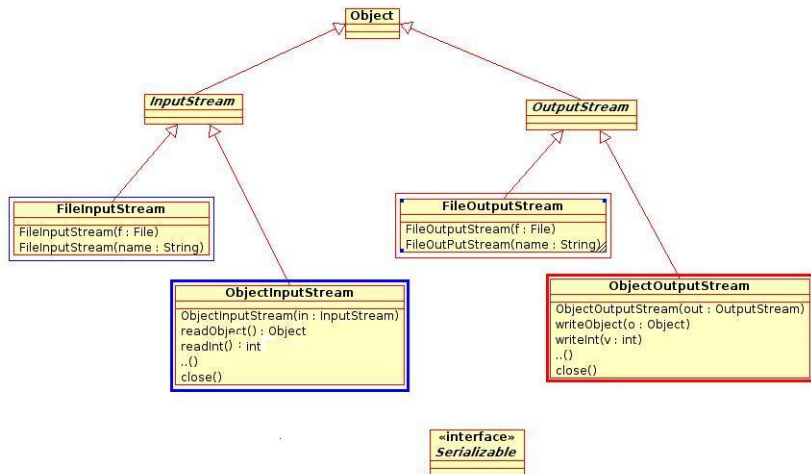
## Caracteres-Estandar

Mecanismo

Batalla Naval

Final

# Objetos





# Objetos

## Para escribir

```
Worm w = new Worm(6, 'a');  
System.out.println("w = " + w);  
ObjectOutputStream out = new ObjectOutputStream(  
    new FileOutputStream("worm.out"));  
out.writeObject("Worm storage\n");  
out.writeObject(w);  
out.close();
```



# Objetos

## Para leer

```
ObjectInputStream in = new ObjectInputStream(  
    new FileInputStream("worm.out"));  
String s = (String)in.readObject();  
Worm w2 = (Worm)in.readObject();  
System.out.println(s + "w2 = " + w2);  
in.close();
```



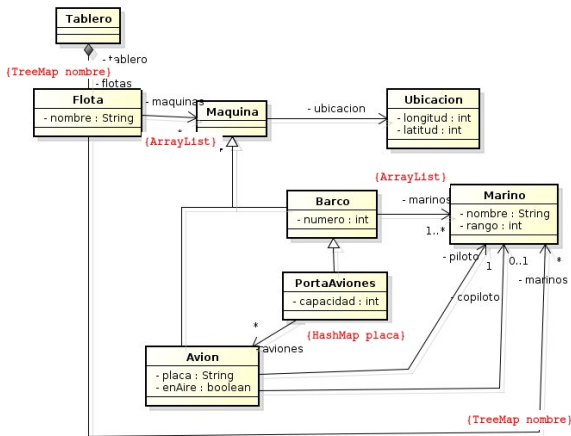
# Objetos

## Excepciones

```
// Throw exceptions to console:  
public static void main(String[] args)  
    throws ClassNotFoundException,  
        IOException {
```

# Batalla Naval

## Como objeto



- ▶ `salve(archivo:String)`
- ▶ `cargue(archivo:String)`

# Agenda

## Conceptos

En general

En java

## Objetos

Mecanismo

Batalla Naval

## Texto-Archivos

Mecanismo

Batalla Naval

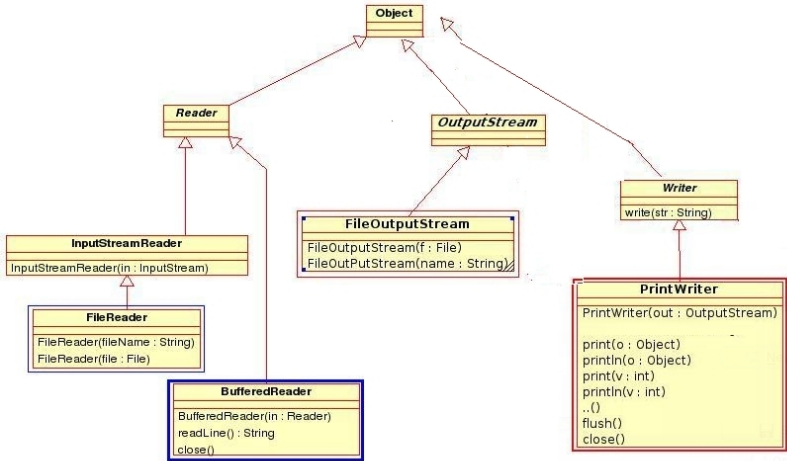
## Caracteres-Estandar

Mecanismo

Batalla Naval

Final

# Texto



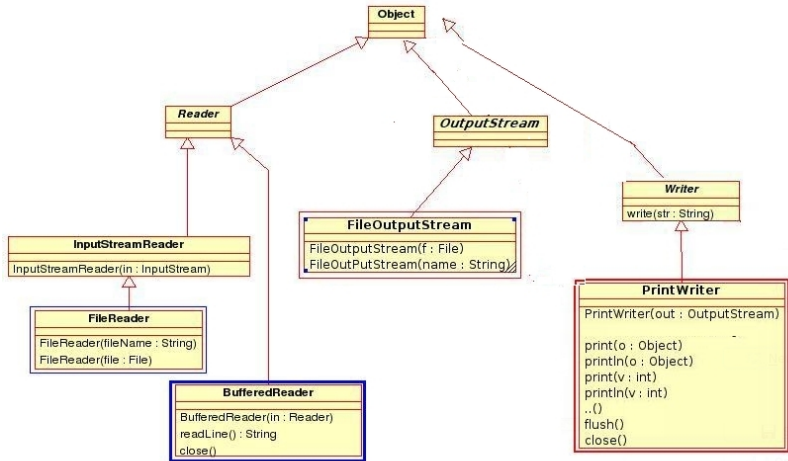
# Texto

## Escribir

```
PrintWriter pw = new PrintWriter(  
    new FileOutputStream(nameOfFileToBeWrittenTo) );  
  
while (we still have more data to output) {  
    pw.println(whatever data we wish to output);  
}  
  
// Close the PrintWriter, which automatically closes  
// the encapsulated FileOutputStream, as well.  
pw.close();
```



# Texto





# Texto

## Leer

```
BufferedReader bIn = new BufferedReader(  
    new FileReader(nameOfFileToBeReadFrom) );  
// Read the first line from the file.  
String line = bIn.readLine();  
while (line != null) {  
    line = line.trim();  
    line = bIn.readLine();  
}  
bIn.close();
```

## **trim**

```
public String trim()
```

Returns a copy of the string, with leading and trailing whitespace omitted.

If this `String` object represents an empty character sequence, or the first and last characters of character sequence represented by this `String` object both have codes greater than `'\u0020'` (the space character), then a reference to this `String` object is returned.

Otherwise, if there is no character with a code greater than `'\u0020'` in the string, then a new `String` object representing an empty string is created and returned.

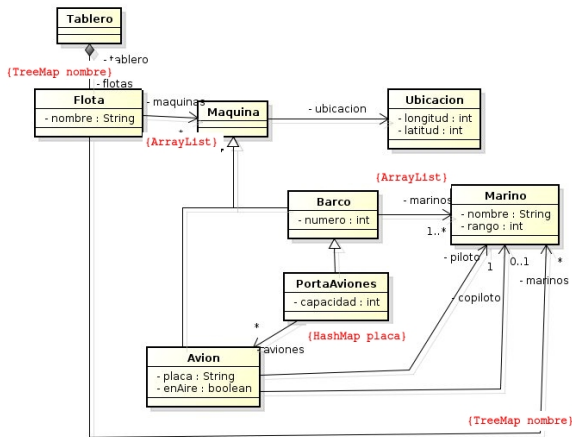
Otherwise, let  $k$  be the index of the first character in the string whose code is greater than `'\u0020'`, and let  $m$  be the index of the last character in the string whose code is greater than `'\u0020'`. A new `String` object is created, representing the substring of this string that begins with the character at index  $k$  and ends with the character at index  $m$ —that is, the result of `this.substring( $k$ ,  $m+1$ )`.

This method may be used to trim whitespace (as defined above) from the beginning and end of a string.

### **Returns:**

A copy of this string with leading and trailing white space removed, or this string if it has no leading or trailing white space.

# Batalla Naval



- Generar, a archivo, un informe de las flotas: número de flotas y nombre y número de máquinas de cada una
- Adicionar nuevas flotas, sus nombres están en un archivo

# Agenda

## Conceptos

En general

En java

## Objetos

Mecanismo

Batalla Naval

## Texto-Archivos

Mecanismo

Batalla Naval

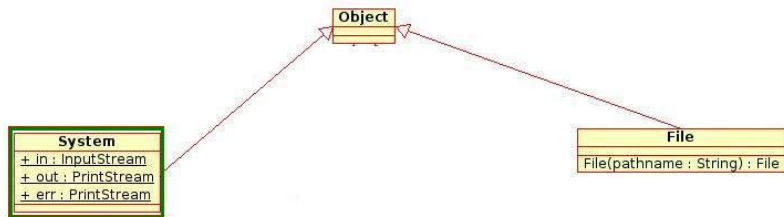
## Caracteres-Estandar

Mecanismo

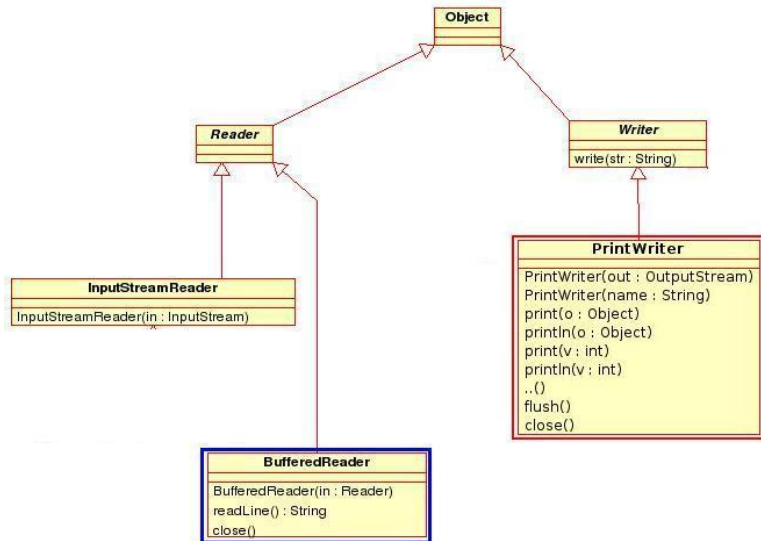
Batalla Naval

Final

# Entrada-Salida



# Caracteres



# Texto-Estandar

## Para leer

```
import java.io.*;

public class Echo {
    public static void main(String[] args)
        throws IOException {
        BufferedReader in = new BufferedReader(
            new InputStreamReader(System.in));
        String s;
        while((s = in.readLine()) != null && s.length() != 0)
            System.out.println(s);
        // An empty line or Ctrl-Z terminates the program
    }
} ///:~
```

# Texto-Estandar

## Para escribir

```
import java.io.*;

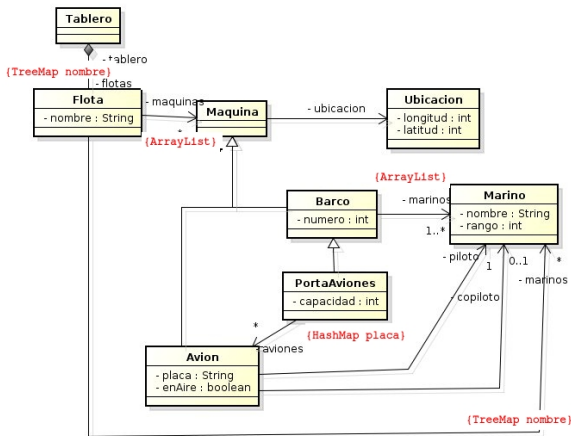
public class ChangeSystemOut {

    public static void main(String[] args) {
        PrintWriter out = new PrintWriter(System.out, true);
        out.println("Hello, world");
    }
} ///:~
```



# Batalla Naval

## BatallaNaval Consultor



- Un usuario REAL quiere consultar la información básica de una flota dado su nombre

Número de máquinas

# Texto-Estandar

## Redireccionando

```
//: cl2:Redirecting.java
// Demonstrates standard I/O redirection.
import java.io.*;
public class Redirecting {
    // Throw exceptions to console:
    public static void main(String[] args)
        throws IOException {
        PrintStream console = System.out;
        BufferedInputStream in = new BufferedInputStream(
            new FileInputStream("Redirecting.java"));
        PrintStream out = new PrintStream(
            new BufferedOutputStream(
                new FileOutputStream("test.out")));
        System.setIn(in);
        System.setOut(out);
        System.setErr(out);
        BufferedReader br = new BufferedReader(
            new InputStreamReader(System.in));
        String s;
        while((s = br.readLine()) != null)
            System.out.println(s);
        out.close(); // Remember this!
        System.setOut(console);
    }
} ////:~
```

# Texto-Estandar

## Redireccionando- De System.

static void	<a href="#">setErr(PrintStream err)</a> Reassigns the "standard" error output stream.
static void	<a href="#">setIn(InputStream in)</a> Reassigns the "standard" input stream.
static void	<a href="#">setOut(PrintStream out)</a> Reassigns the "standard" output stream.