Fundamentos de programación

Grado de Ingeniería del Software

Jorge García Gutiérrez
jorgarcia@us.es
F1.56

Lo último que vimos fue...

- Tipos básicos y wrappers.
- Variables y constantes
- If-else y switch
- While, for y for-extendido.
- Listas, conjuntos y Maps

Illustrations by Pixeltrue on icons8

Índice

- Metodología para definir tipos
- Inmutabilidad: Clases y Records
- Herencia e interfaces
- Tipo Object
- Tipo Comparable
- Restricciones y excepciones
- Constructor a partir de String

Illustrations by Pixeltrue on icons8

Índice

- **Metodología para definir tipos**
- Inmutabilidad: Clases y Records
- Herencia e interfaces
- Tipo Object
- Tipo Comparable
- Restricciones y excepciones
- Constructor a partir de String

Illustrations by Pixeltrue on icons8

Nombre del Tipo

- Propiedades:
  - Nombre de propiedad: Tipo, Consultable o no, Modificable o no, Derivada o no
  - ...
- Constructores:
  - Constructor 1
  - ...
- Restricciones:
  - Restricción 1
  - ...
- Operaciones:
  - ...
- Criterio de igualdad
- Representación como cadena
- Orden natural (si lo tiene)

```java
public class PuntoImpl implements Punto {

	// Atributos
	private Double x;
	private Double y;

	// Constructores
	public PuntoImpl (Double x1, Double y1) {
		this.x = x1;
		this.y = y1;
	}
	public PuntoImpl() {
		this.x = 0.;
		this.y = 0.;
	}

	// Observadores
	public Double getX() {
		return x;
	}
	public Double getY() {
		return y;
	}

	// Modificadores
	public void setX(Double x1) {
		x = x1;
	}
	public void setY(Double y1) {
		y = y1;
	}

	// Otras operaciones
	public Double getDistanciaAOtroPunto(Punto p) {
		Double dx = this.getX() - p.getX();
		Double dy = this.getY() - p.getY();
		return Math.sqrt(dx * dx + dy * dy);
	}
}
```

Índice

- Metodología para definir tipos
- **Inmutabilidad: Clases y Records**
- Herencia e interfaces
- Tipo Object
- Tipo Comparable
- Restricciones y excepciones
- Constructor a partir de String

Illustrations by Pixeltrue on icons8

```java
public record Persona(String DNI, String nombre, String apellidos,
        LocalDate fechaNacimiento) {
}
```

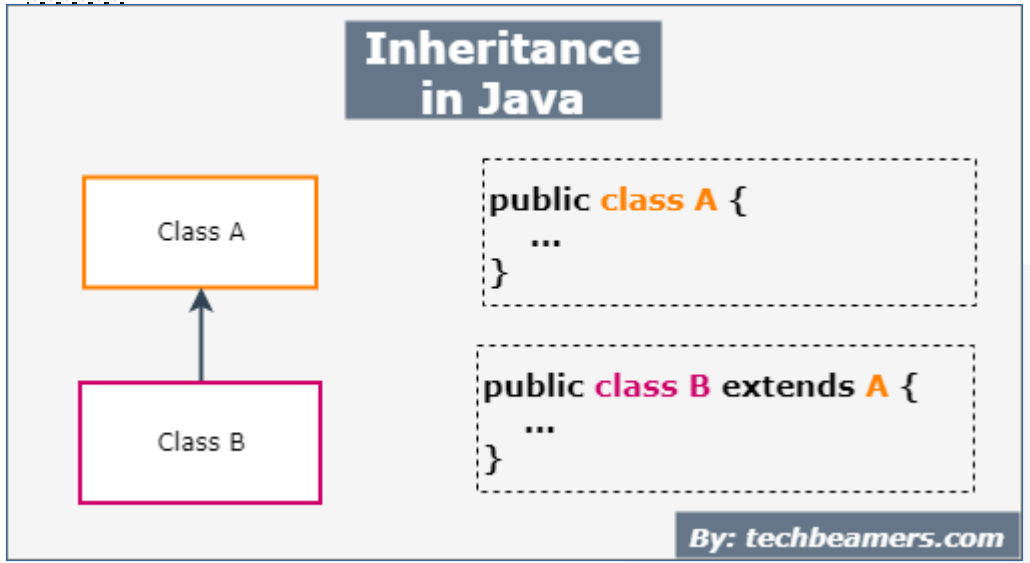
```java
Persona p = new Persona("11111111A", "Jane", "Doe", LocalDate.of(2001,5,17));
System.out.println("Nombre completo: " + p.nombre() + " " + p.apellidos());
```

```java
public class PuntoImpl implements Punto {

	// Atributos
	private Double x;
	private Double y;

	// Constructores
	public PuntoImpl (Double x1, Double y1) {
		this.x = x1;
		this.y = y1;
	}
	public PuntoImpl() {
		this.x = 0.;
		this.y = 0.;
	}

	// Observadores
	public Double getX() {
		return x;
	}
	public Double getY() {
		return y;
	}

	// Modificadores
	public void setX(Double x1) {
		x = x1;
	}
	public void setY(Double y1) {
		y = y1;
	}

	// Otras operaciones
	public Double getDistanciaAOtroPunto(Punto p) {
		Double dx = this.getX() - p.getX();
		Double dy = this.getY() - p.getY();
		return Math.sqrt(dx * dx + dy * dy);
	}
}
```

Índice

- Metodología para definir tipos
- Inmutabilidad: Clases y Records
- **Herencia e interfaces**
- Tipo Object
- Tipo Comparable
- Restricciones y excepciones
- Constructor a partir de String

Illustrations by Pixeltrue on icons8




```
public class Calculator {
    int add(int a , int b)
    {
        return a + b;
    }

    int sub(int a , int b)
    {
        return a - b;
    }
}

public class AdvancedCalculator extends Calculator {
    int mult(int a , int b)
    {
        return a * b;
    }

    int div(int a , int b)
    {
        return a / b;
    }
}
```

https://techbeamers.com/java-inheritance/

```
interface Animals{

 void makeSound();

}
```

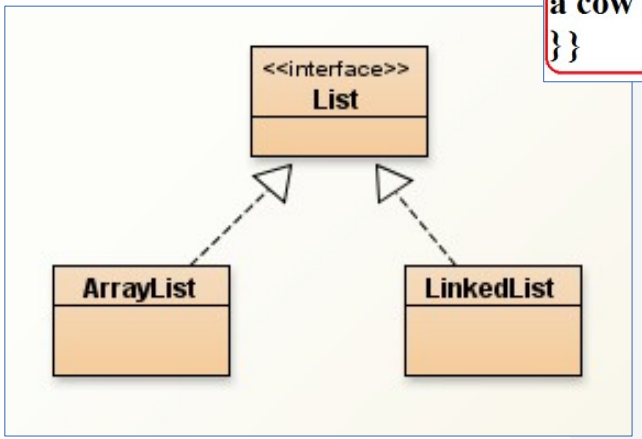
interface declaration in java

Abstract method

```
class Dog implements Animals{
 void makeSound() {
 System.out.println("The sound a
dog makes is woof woof!! ");
 }
}
```

```
class Cow implements
Animals{
void makeSound() {
System.out.print("The sound
a cow makes is moo");
}}
```

```
class Cat implements Animals {
 void makeSound() {
System.out.print("The sound a
cat makes is meow");
 }
}
```

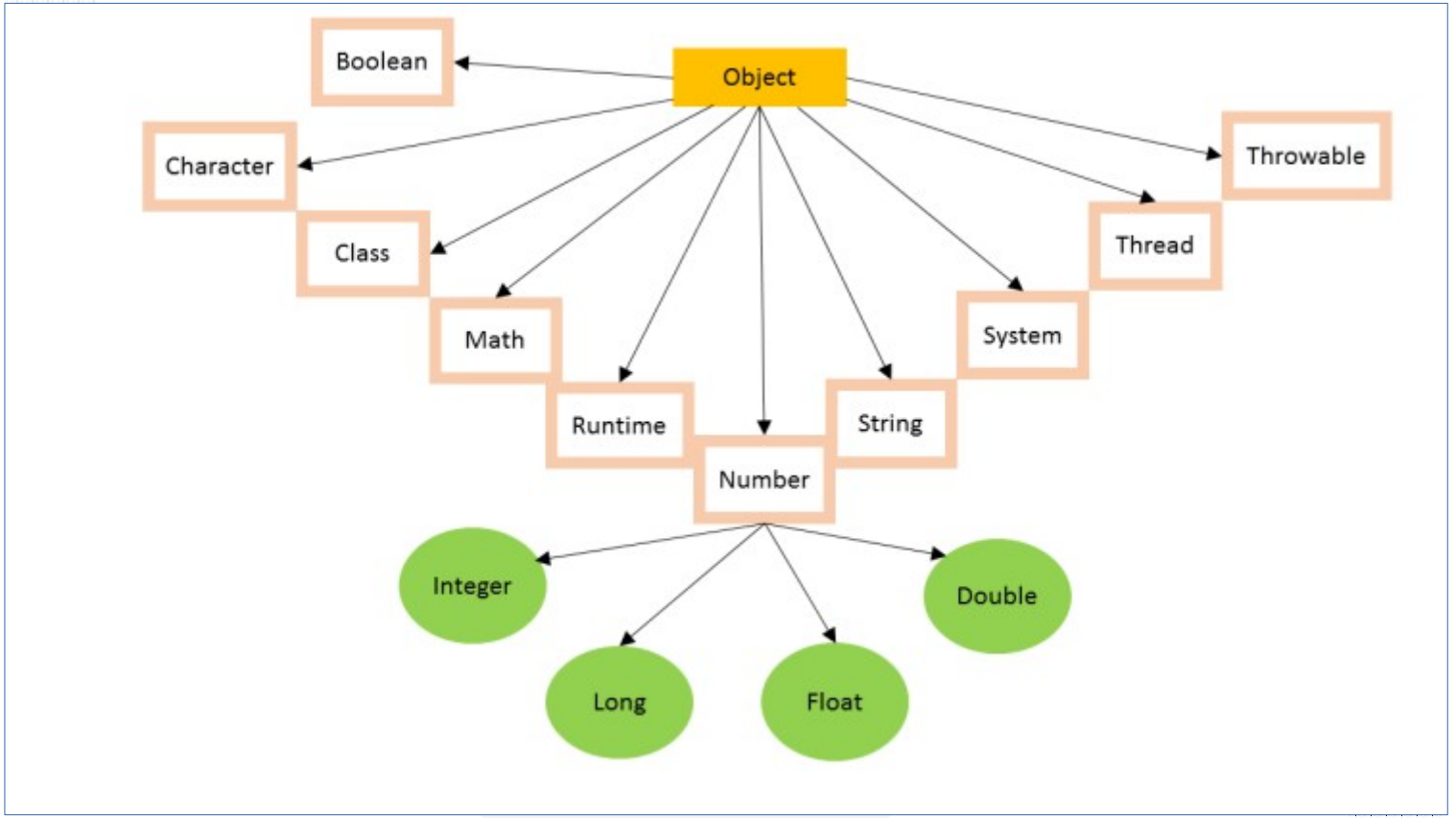


https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=671:para-que-sirven-las-interfaces-java-implementar-una-interfaz-del-api-ventajas-y-ejemplos-basicos-cu00697b&catid=68&Itemid=188

Índice

- Metodología para definir tipos
- Inmutabilidad: Clases y Records
- Herencia e interfaces
- **Tipo Object**
- Tipo Comparable
- Restricciones y excepciones
- Constructor a partir de String

Illustrations by Pixeltrue on icons8



https://www.linkedin.com/pulse/what-methods-object-class-java-omar-ismail/

Object methods

method	description
`protected Object clone()`	creates a copy of the object
`public boolean equals(Object o)`	returns whether two objects have the same state
`protected void finalize()`	called during garbage collection
`public Class<?> getClass()`	info about the object's type
`public int hashCode()`	a code suitable for putting this object into a hash collection
`public String toString()`	text representation of the object

`public void notify()`
`public void notifyAll()`
`public void wait()`
`public void wait(...)`
	methods related to concurrency and locking (seen later)

- What does this list of methods tell you about Java's design?

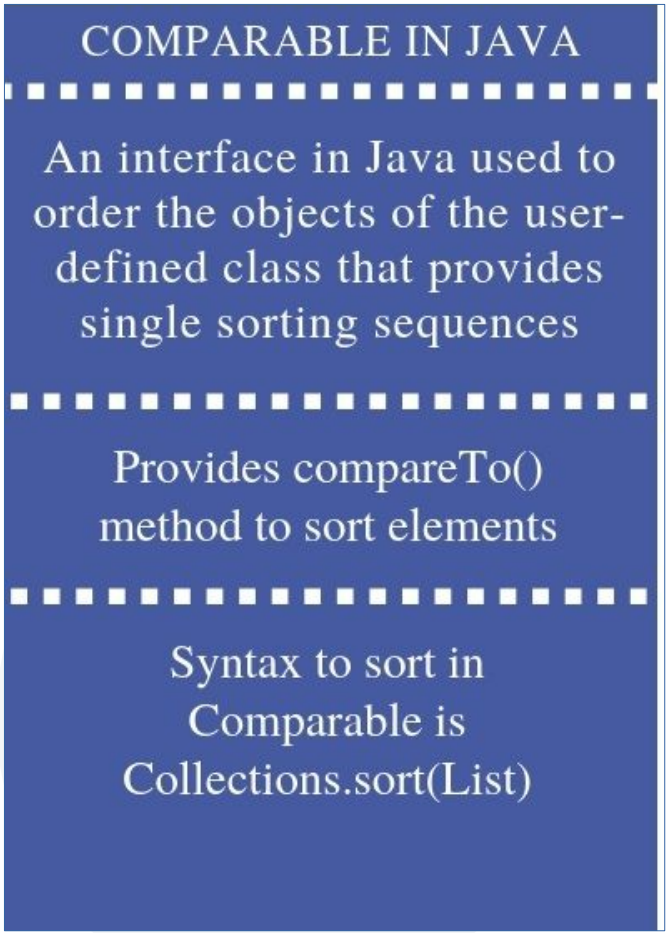
3

https://www.quora.com/What-is-the-significance-of-the-object-class-in-Java

Índice

- Metodología para definir tipos
- Inmutabilidad: Clases y Records
- Herencia e interfaces
- Tipo Object
- **Tipo Comparable**
- Restricciones y excepciones
- Constructor a partir de String

Illustrations by Pixeltrue on icons8



https://pediaa.com/what-is-the-difference-between-comparable-and-comparator-in-java/

Índice

- Metodología para definir tipos
- Inmutabilidad: Clases y Records
- Herencia e interfaces
- Tipo Object
- Tipo Comparable
- **Restricciones y excepciones**
- Constructor a partir de String

Illustrations by Pixeltrue on icons8

```java
public class ThrowAnException {
    public static void main (String[] args) {
        try {
            int result=divideInt(10,5);
            System.out.format("10 divided by 5 is %d%n", result);
            divideInt(10,0);
            System.out.format("10 divided by 0 is %d%n", result);
        }
        catch (IllegalArgumentException ex) {
            System.out.println(ex.getMessage());
        }
    }
    public static int divideInt(int i, int j) {
        if (j == 0) {
            throw new IllegalArgumentException("Divisor cannot be zero!");
        }
        return i/j;
    }
}
```

https://www.webucator.com/article/how-to-throw-an-exception-in-java/

Índice

- Metodología para definir tipos
- Inmutabilidad: Clases y Records
- Herencia e interfaces
- Tipo Object
- Tipo Comparable
- Restricciones y excepciones
- **Constructor a partir de String**

Illustrations by Pixeltrue on icons8

```java
public PersonaImpl(String s) {
        String[] sp = s.split(",");
        if (sp.length != 4) {
                throw new IllegalArgumentException(
                        "Cadena con formato no válido");
        }
        String nombre = sp[1].trim();
        checkNombre(nombre);
        this.dni = sp[0].trim();
        this.nombre = nombre;
        this.apellidos = sp[2].trim();
        this.fechaNacimiento = LocalDate.parse(sp[3].trim(),
                DateTimeFormatter.ofPattern("d/M/y"));
}
```

TAREA:

1. Definir y testear el tipo Vuelo en sus dos versiones (mutable e inmutable).