**Vídeos - FLIP - Tema 1 Introducción**

**1 -** [**Motivación**](https://media.upv.es/player/?id=c979c820-922b-11e7-ad73-4bd35f901b1a)

**2 - Conceptos esenciales en lenguajes de programación**

[Tipos. Sistemas de tipos](https://media.upv.es/player/?id=c60173d0-9237-11e7-ad73-4bd35f901b1a)

[Polimorfismo ad-hoc](https://media.upv.es/player/?id=d0a96390-92d9-11e7-ad73-4bd35f901b1a)

[Polimorfismo universal](https://media.upv.es/player/?id=55dbb2b0-92ef-11e7-ad73-4bd35f901b1a)

[polimorfismo de inclusión (o herencia)](https://media.upv.es/player/?id=e87e2f70-b031-11e8-97bc-43894ab77085)

NOTA 1: En el vídeo anterior, en el ejemplo final (Employee, Executive), se comenta que no es necesario definir el constructor de Executive ya que usará el constructor de la superclase por defecto. Esto no es correcto, ya que solo funciona cuando se trata de un constructor sin argumentos. En este ejemplo, debería definirse el método constructor de la clase Executive como sigue:

**public Executive(String name, int salary) {  
      super(name,salary);  
  }**

NOTA 2: En la versión disponible de las transparencias se ha incluido una tabla que detalla los atributos de visibilidad para el caso de Java, incluyendo el rol de los paquetes.

[polimorfismo de inclusión (o herencia)-parte 2](https://media.upv.es/player/?id=55051780-b041-11e8-97bc-43894ab77085)

[Reflexión](https://media.upv.es/player/?id=461b1330-b04d-11e8-97bc-43894ab77085)

Procedimientos y control de flujo

[Paso de parámetros](https://media.upv.es/player/?id=d109ab10-9491-11e7-ad73-4bd35f901b1a)

[Ejercicio - Paso de parámetros](https://media.upv.es/player/?id=1f1e1a50-fe40-11ea-a44c-87616fca7ff1)

Antes de pasar a ver la noción de ámbito de las variables, el siguiente es un ejemplo de paso de parámetros por necesidad en un lenguaje funcional:

Dada la siguiente función que devuelve el segundo argumento multiplicado por dos

sel2nd x y = 2\*y

 si la invocamos con la expresión sel2nd (2\*3) 5  con paso de parámetros por necesidad no se calcularía el valor de la expresión 2\*3  al no utilizarse en el cuerpo de la función.

[Ámbito (o alcance) de las variables](https://media.upv.es/player/?id=cec27600-9498-11e7-ad73-4bd35f901b1a)

[Ejercicio - Ámbito (o alcance) de las variables](https://media.upv.es/player/?id=844d4770-fe4f-11ea-a44c-87616fca7ff1)

[Gestión de memoria](https://media.upv.es/player/?id=17fff210-949f-11e7-ad73-4bd35f901b1a)

**3 - Principales paradigmas de programación: imperativo, funcional, lógico, OO, concurrente**

[Paradigma imperativo](https://media.upv.es/player/?id=149c6e10-7ff5-11e6-908a-45bd90a177e0)

[Paradigma declarativo](https://media.upv.es/player/?id=e98ee960-7fee-11e6-908a-45bd90a177e0)

[Programación declarativa vs programación imperativa](https://media.upv.es/player/?id=ab6672d0-7ff2-11e6-908a-45bd90a177e0)

[Paradigma orientado a objetos](https://media.upv.es/player/?id=edcf9c20-7ff5-11e6-908a-45bd90a177e0)

[Paradigma concurrente (y paralelo)](https://media.upv.es/player/?id=e2482010-b7a3-11e8-a361-599725480ca3)

**4 - Otros paradigmas. Basado en interacción, emergente**s

[Paradigma basado en interacción](https://media.upv.es/player/?id=ddb50e70-7ffe-11e6-908a-45bd90a177e0)

[Otros paradigmas emergentes](https://media.upv.es/player/?id=591543e0-8000-11e6-908a-45bd90a177e0)