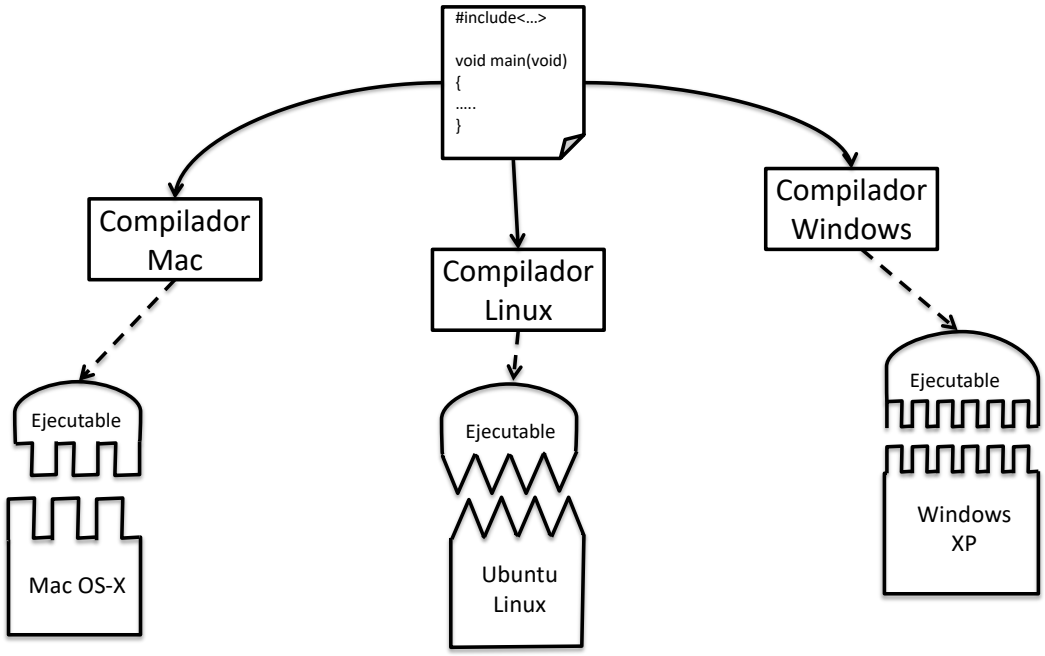
Introducción a la Plataforma Java

Programación Orientada a Objetos

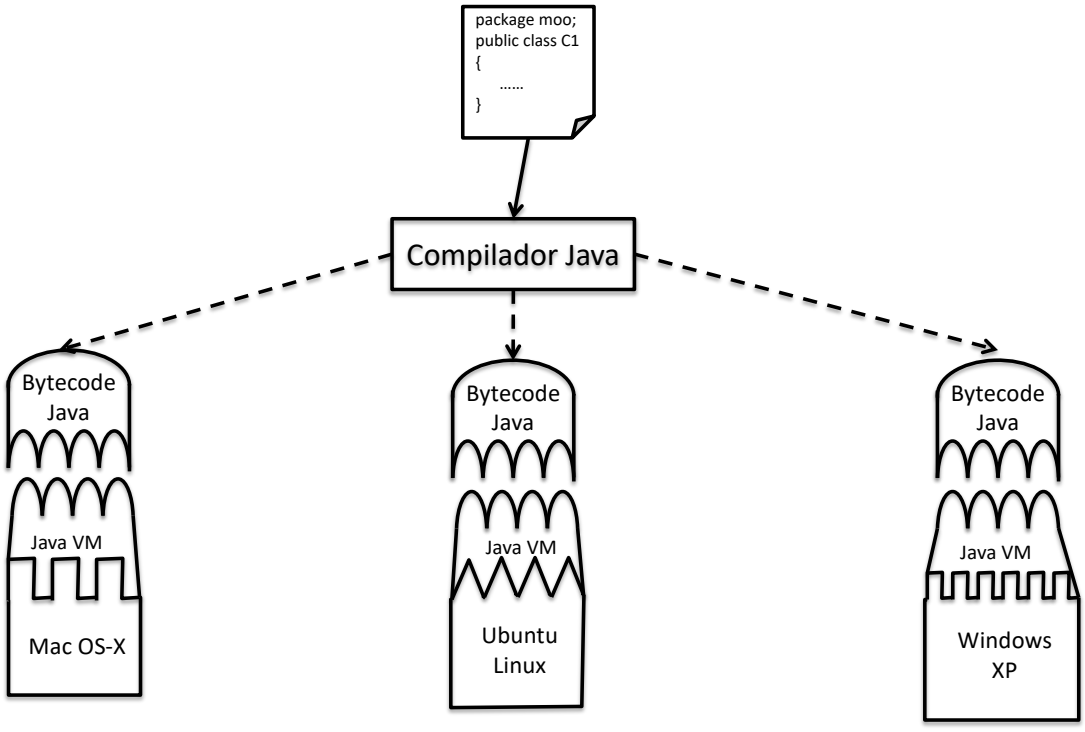
Filosofía de Java

- Java no es sólo un lenguaje, es una **Plataforma**
  - Orientada a Objetos
  - Independiente de la máquina
  - Gestión dinámica de memoria
  - Otras características que no se contemplan en el curso….

Aplicaciones dependientes de máquina



Aplicaciones independientes de la máquina



Gestión Dinámica de Memoria

- Lenguajes como C gestionan estáticamente la memoria
  - Inconvenientes cuando no sabemos cuánta memoria se debe usar (ej. Listas grandes)
    - ¿Reservamos más de la que sepamos que necesitaremos nunca? → Desperdicio de memoria
  - Crear y destruir memoria es costoso
- Java gestiona dinámicamente la memoria
  - Crear objetos cuando se vayan a usar
  - Destruir objetos cuando la memoria está llena → **Recolector de basura**

Introducción al Lenguaje Java

Programación Orientada a Objetos

Sintaxis

- Java adopta palabras y estructuras de C…
  - Tipos de datos básicos: int, long, char, float…
  - Control de flujo: if, while, do-while, for, switch.
  - Operaciones, llamadas a funciones: a = a + func(b)
- … y añade nuevas
  - Nuevos tipos: byte (-128..127), boolean (true/false).
  - Definición de clases: class
  - Visibilidad de elementos: public, protected, private…
  - Llamadas a métodos de objetos, acceso a atributos…
- … también quita o cambia
  - Directivas de preprocesador #include, #define…
  - Con Java se acabaron los punteros (¿Buena noticia? Sólo en parte)

Punteros vs. Referencias

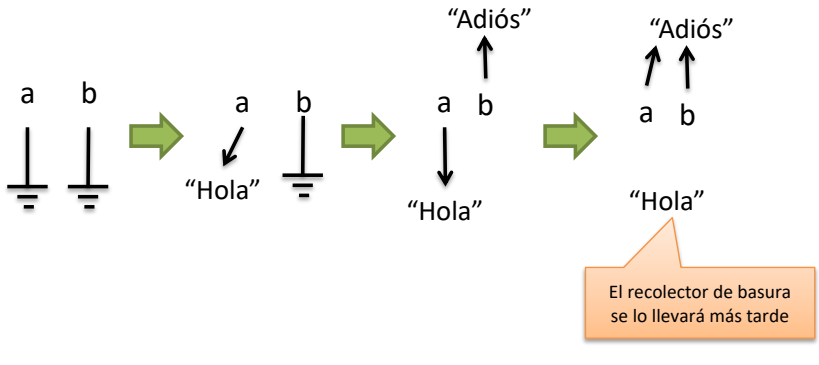
- Tipos de datos básicos: int, char… se tratan internamente como en C: **por valor**

```
int a, b, c;
a = 3;
b = a + 1;
c = a;
```

a	b	c
0	0	0
3	0	0
3	4	0
3	4	3

- Objetos y vectores: se tratan **por referencia**

```
String a,b;
a = “Hola”;
b = “Adiós”;
a = b;
```



Cadenas de texto

- En C tratábamos las cadenas de texto como un array de char

```
char nombre[100] = "Laia";
printf("%s tiene %d letras", nombre, strlen(nombre));
```

- En Java las tratamos como una clase especial, con la que podemos operar de manera más sencilla:

```
String nombre= "Laia";
System.out.println(
       nombre + " tiene " + nombre.length() + " letras");
      //concatenación      //método 'length'
```

Comparaciones de tipos básicos y referencias

```
int i = 2, j = 2;

boolean b1 = (i == j); // b1 == true or false?

String s1 = "Hola";
String s2 = "Hola";

boolean b2 = (s1 == s2);    //b2 == true or false?
boolean b3 = s1.equals(s2); //b3 == true or false?

s1 = s2;

boolean b4 = (s1 == s2); //b4 == true or false?
```