ULHI
Urrutiko Lanbide Heziketako Institutua
Instituto de Formación Profesional a Distancia

UD 7: Clases inmutables en Java

Curso 2018/19

EUSKO JAURLARITZA
GOBIERNO VASCO
HEZKUNTZA, HIZKUNTZA POLITIKA ETA KULTURA SAILA
Lanbide Heziketako Sailburuordetza
DEPARTAMENTO DE EDUCACIÓN, POLÍTICA LINGÜÍSTICA Y CULTURA
Viceconsejería de Formación Profesional

Añadir un objeto

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

   Ruta miRuta = new Ruta("Mi excursión");

   Punto unPunto = new Punto(0, 0,
   "Inicio");
   miRuta.addPunto(unPunto);

   System.out.println(miRuta);

   unPunto.setEtiqueta("Fin");
   System.out.println(miRuta);

}
```

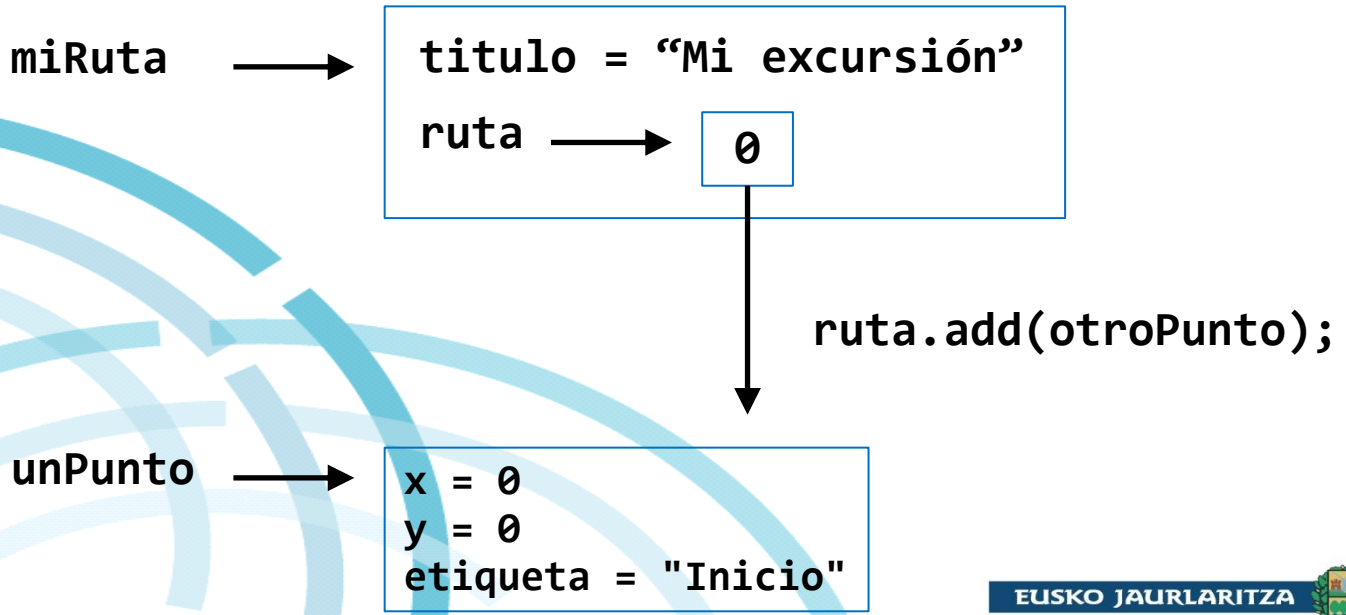
```
public class Ruta {
   private String titulo;
   private ArrayList<Punto> ruta;

   public void addPunto(Punto otroPunto) {
      ruta.add(otroPunto);
   }

}
```

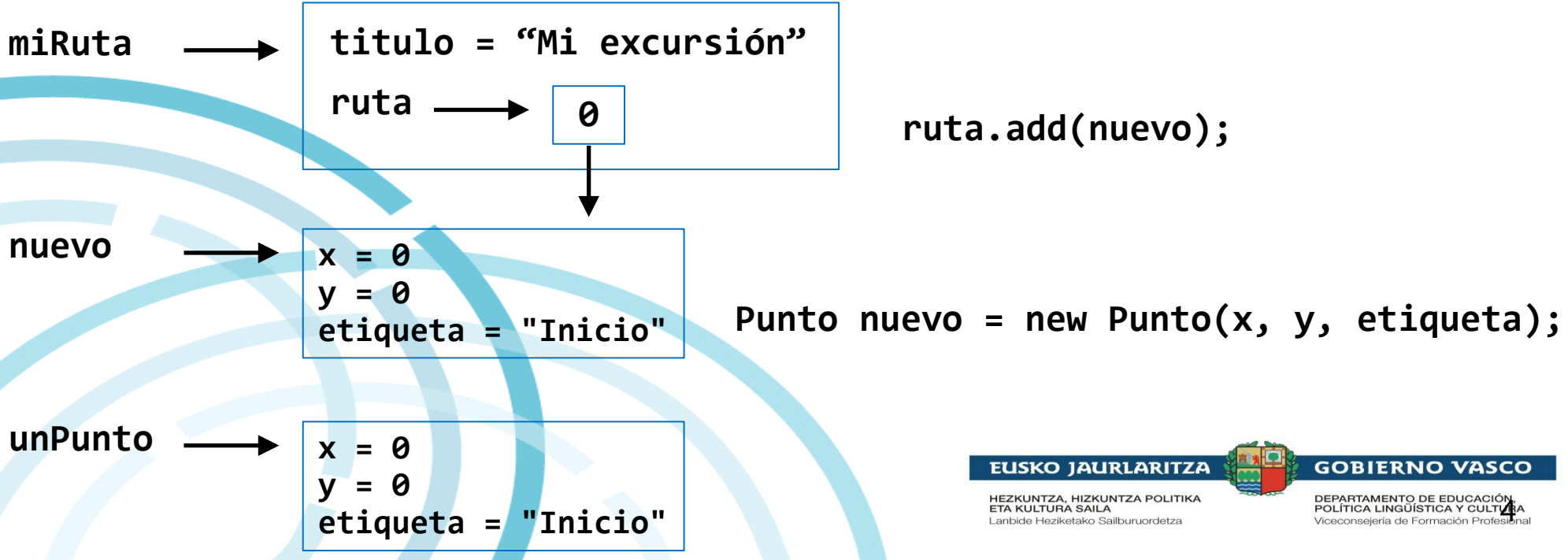
Añadir un objeto

- Añadimos el objeto que recibimos directamente
- Si modificamos el elemento 0 de ruta, el valor de unPunto cambia
- Si cambiamos el objeto unPunto, estamos modificamos ruta



Añadir un objeto

- Queremos desconectar ruta del objeto unPunto
- Debemos **hacer una copia antes de la asignación**



Añadir un objeto

```
public void addPunto(Punto otroPunto) {
   int x = otroPunto.getX();
   int y = otroPunto.getY();
   String etiqueta = otroPunto.getEtiqueta();
   Punto nuevo = new Punto(x, y, etiqueta);
   ruta.add(otroPunto);

}
```

- Recordar que los objetos se pasan por referencia.
- Evitar asignar objetos directamente a los atributos privados. Realmente estamos creando un acceso a ellos.
- Hacer una copia antes de asignarlos.

Devolver un objeto

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

   Ruta miRuta = new Ruta("Mi excursión");

   Punto unPunto = new Punto(0, 0,
   "Inicio");
   miRuta.addPunto(unPunto);
   System.out.println(miRuta);

   Punto otroPunto = miRuta.getPunto(0);
   otroPunto.setCoordenadas(10, 10);
   System.out.println(miRuta);

}
```

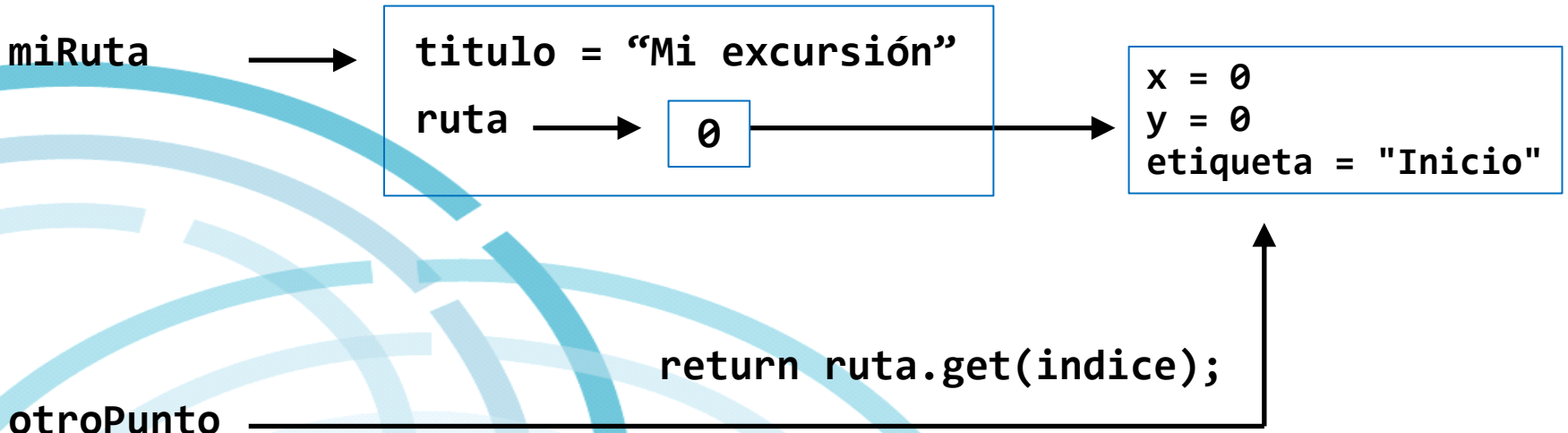
```
public class Ruta {
   private String titulo;
   private ArrayList<Punto> ruta;

   public Punto getPunto(int indice) {
      return ruta.get(indice);
   }

}
```

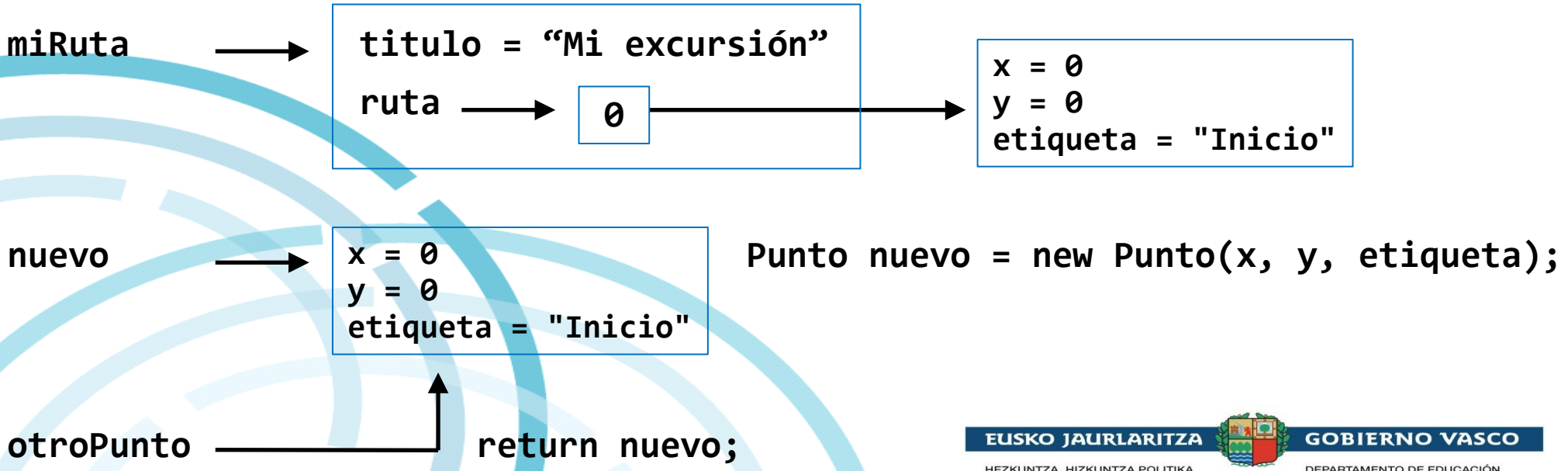
Devolver un objeto

- Devolvemos el elemento 0 y lo asignamos a otro objeto
- Si cambiamos el elemento 0 de ruta, el valor de otroPunto cambia
- Si cambiamos el objeto otroPunto, estamos modificamos ruta



Devolver un objeto

- Queremos desconectar ruta del objeto otroPunto
- Debemos **hacer una copia antes de devolverlo**



Devolver un objeto

```
public Punto getPunto(int indice) {
   Punto unPunto = ruta.get(indice);
   int x = unPunto.getX();
   int y = unPunto.getY();
   String etiqueta = unPunto.getEtiqueta();
   Punto nuevo = new Punto(x, y, etiqueta);
   return nuevo;

}
```

- Evitar devolver objetos que representan atributos privados. Realmente estamos creando un acceso a ellos.
- Hacer una copia antes.