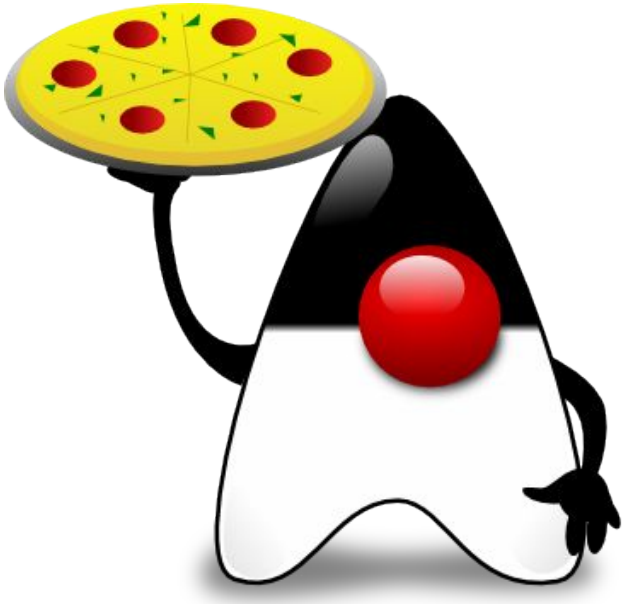
Clase 16: Enum

Programación & Laboratorio III

Agenda

- **Enum - Características**
- Ejemplo 1
- Ejemplo 2
- Más características
- Ejemplo 3
- Operador “==”



Enum - Características

- La palabra reservada “enum” fue introducida en Java 5 para representar el tipo especial de clase que siempre extiende de java.lang.Enum.
- Constantes definidas de esta manera hacen el código más legible, permiten verificación en tiempo de compilación, documentan por adelantado la lista de valores aceptados y evitan el comportamiento inesperado si es que se reciben valores no válidos.
- Ejemplo:

```
public enum PizzaStatus {
     ORDERED,
     READY,
     DELIVERED;
}
```

Enum - Características (2)

- Un tipo enumerado restringe los posibles valores que puede tomar una variable. Esto ayuda a reducir los errores en el código y permite algunos usos especiales interesantes.
- Por convención, los nombres de los valores que puede tomar **se escriben en letras mayúsculas** para recordarnos que son valores fijos (que en cierto modo podemos ver como constantes).
- Una vez declarado el tipo enumerado, todavía no existen variables hasta que no las creemos explícitamente:

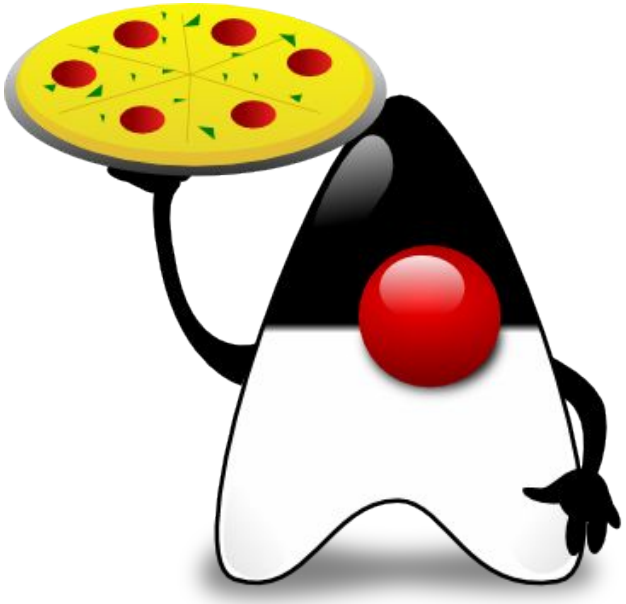
```
TipoEnumerado nombreVariable;  → PizzaStatus status;
```

Enum - Características (3)

- Un tipo enumerado puede ser declarado dentro o fuera de una clase, pero no dentro de un método. Por tanto no podemos declarar un enum dentro de un método main; si lo hacemos nos saltará el error de compilación "`enum types must not be local`" (los tipos enumerados no deben ser locales a un método).
- Declararemos un tipo enumerado como una clase aparte o dentro de otra, pero fuera de cualquier método o constructor.

Agenda

- ~~Enum - Características~~
- **Ejemplo 1**
- Ejemplo 2
- Más características
- Ejemplo 3
- Operador “==”

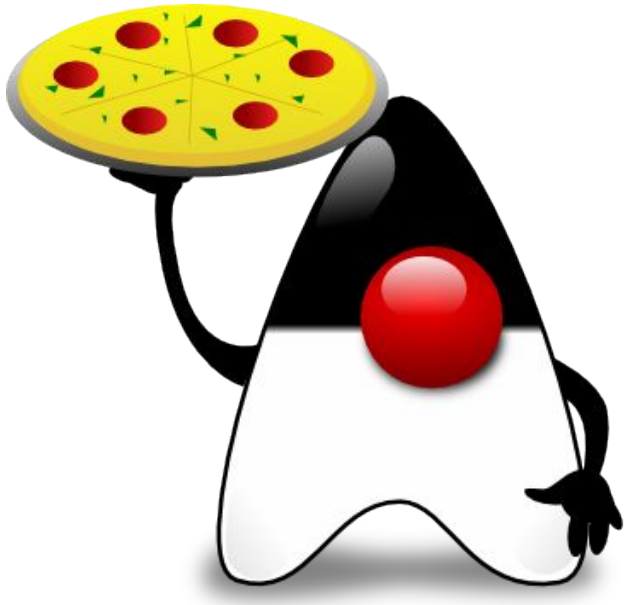


Ejemplo 1: Enum declarado dentro de clase

```
public class Pizza {

    private PizzaStatus status;

    public enum PizzaStatus {
        ORDERED,
        READY,
        DELIVERED;
    }

    public boolean isDeliverable() {
       if (status == PizzaStatus.READY) {
          return true;
       }
       return false;
    }
}
```

Agenda

- ~~Enum - Características~~
- ~~Ejemplo 1~~
- **Ejemplo 2**
- Más características
- Ejemplo 3
- Operador “==”

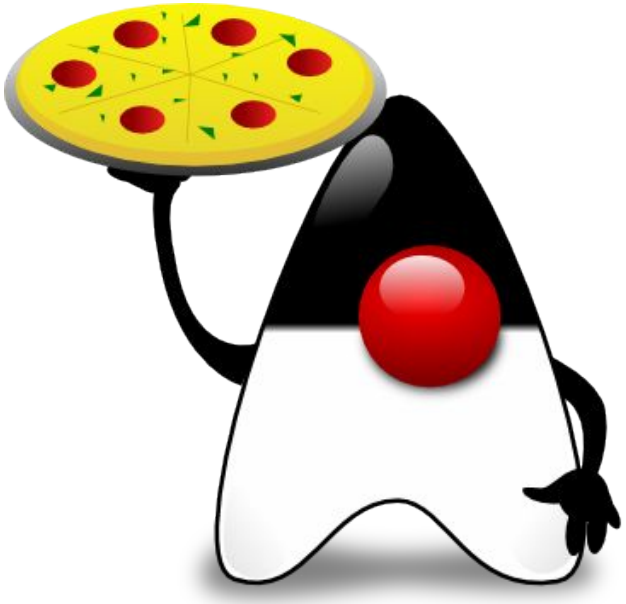


Ejemplo 2: Enum como clase

```
 public enum PizzaStatus {
    ORDERED,
    READY,
    DELIVERED;

    public boolean isDeliverable() {
       if (status == PizzaStatus.READY) {
          return true;
       }
       return false;
    }
}
```

Agenda

- ~~Enum - Características~~
- ~~Ejemplo 1~~
- ~~Ejemplo 2~~
- **Más características**
- Ejemplo 3
- Operador “==”

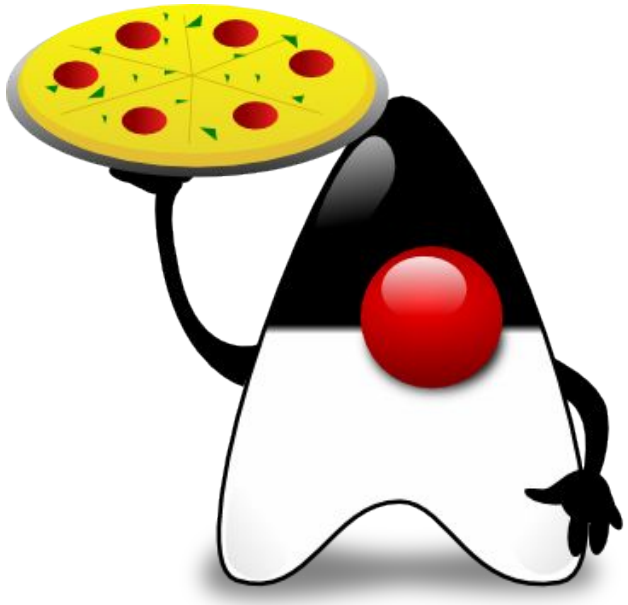


Más características

- Se dispone automáticamente de métodos especiales como por ejemplo el método `values()`, que el compilador agrega automáticamente cuando se crea un enum. Este método devuelve un array conteniendo todos los valores del enumerado en el orden en que son declarados
- Un tipo enumerado puede añadir campos constantes al objeto enumerado y recuperar esos campos. Para ello se usa un constructor especial para tipos enumerados.

Agenda

- ~~Enum - Características~~
- ~~Ejemplo 1~~
- ~~Ejemplo 2~~
- ~~Más características~~
- **Ejemplo 3**
- Operador “==”



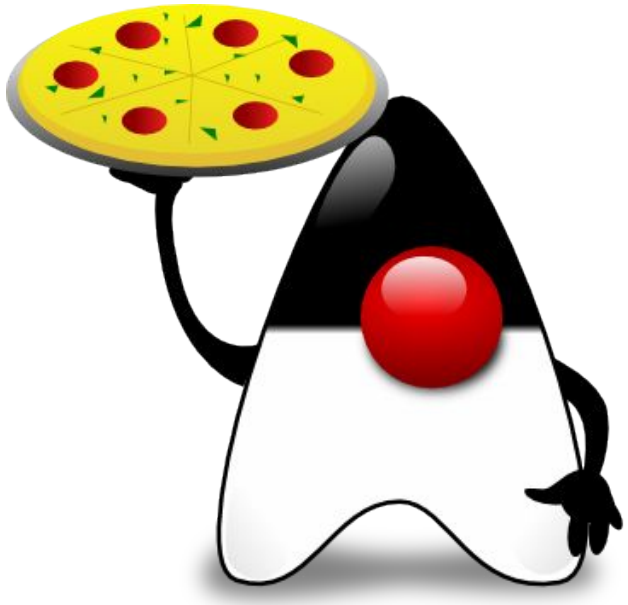
Ejemplo 3: Enum con constructor

```
public enum TipoDeMadera {
      ROBLE ("Castaño verdoso"),
      CAOBA ("Marrón oscuro"),
      NOGAL("Marrón rojizo");

       private String color;

       TipoDeMadera (String color) {
           this.color = color;
       }

      public String getColor() {
          return color;
      }
}
```

Por defecto el constructor es “private”, es redundante escribirlo.

Agenda

- ~~Enum - Características~~
- ~~Ejemplo 1~~
- ~~Ejemplo 2~~
- ~~Más características~~
- ~~Ejemplo 3~~
- **Operador “==”**



Operador “==”

- Usando el tipo enum nos aseguramos que sólo exista una instancia de la constante en la JVM. por lo tanto se puede usar el operador “==” para comparar dos variables del mismo tipo enum.
- Seguridad en tiempo de ejecución:

```
1)   if(pizza.getStatus().equals(Pizza.PizzaStatus.DELIVERED));
2)   if(pizza.getStatus() == Pizza.PizzaStatus.DELIVERED);
```

En la opción 1) si el status de pizza es null, obtendremos una excepción del tipo `NullPointerException`. No así en la opción 2)

Bibliografía oficial

- https://docs.oracle.com/javase/tutorial/java/javaOO/enum.html
- https://docs.oracle.com/javase/8/docs/technotes/guides/language/enums.html