

Comprensión de las definiciones de clases

Mirando adentro de las clases

Principales conceptos a ser cubiertos

- campos
- constructores
- métodos
- parámetros
- instrucciones de asignación
- Instrucciones condicionales

Máquina Expendedora de Boletos Simple

- Exploración del comportamiento de una máquina ingenua.
 - Use el proyecto *maquina-de-boletos-simple*.
 - Las máquinas sólo suministran boletos de un precio fijo.
 - ¿Cómo se determina ese precio?
 - ¿Cómo se ingresa el dinero en la máquina?
 - ¿Cómo contabiliza la máquina el dinero que se ha ingresado?
 - Ej: 2.1, 2.2, 2.3, 2.4, 2.5

Visión interna de las Máquinas Expendedoras de Boletos

- Interactuar con un objeto nos da pistas de su comportamiento.
- Mirar en su interior nos permite determinar como es provisto o implementado ese comportamiento.
- Todas las clases en JAVA tienen una visión interna similar.
- Abrir la clase MaquinaDeBoletos

Estructura básica de una clase

```
public class MaquinaDeBoletos
{
    La parte interior de una
    clase es omitida.
}
```

Envoltorio exterior
de la Maquina de
Boletos

```
public class NombreClase
{
    Campos
    Constructores
    Métodos
}
```

Contenidos de
una clase

Ej.: 2.6, 2.7, 2.8, 2.9, 2.10

Campos

- Los Campos almacenan los valores de un objeto.
- Se los conoce como variables de instancia.
- La opción *Inspect* permite ver los campos de un objeto.
- Los campos definen el estado de un objeto.
- Crear un objeto de la `MaquinaDeBoletos`
- Ej.: 2.11, 2.12, 2.13, 2.14, 2.15

```
public class MaquinaDeBoletos
{
    private int precio;
    private int saldo;
    private int total;

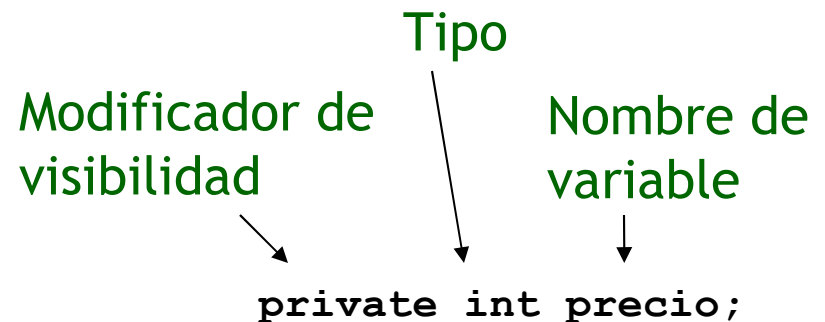
    Otros detalles omitidos.
}
```

Tipo

Modificador de
visibilidad

Nombre de
variable

`private int precio;`

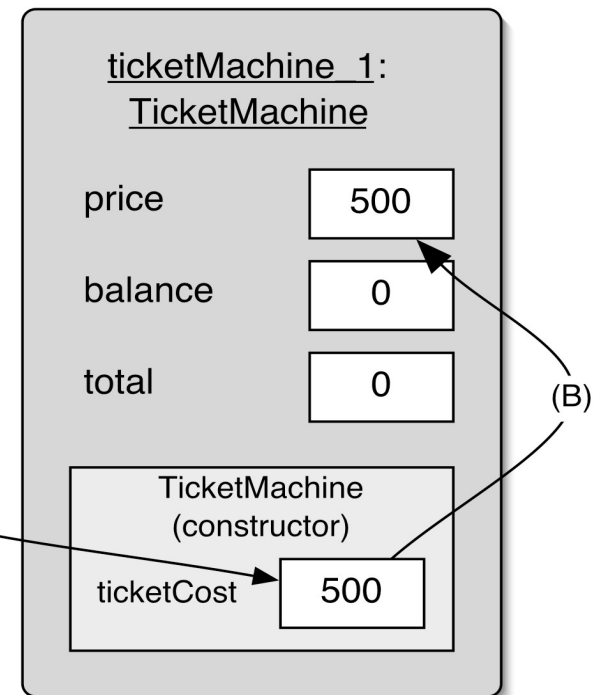
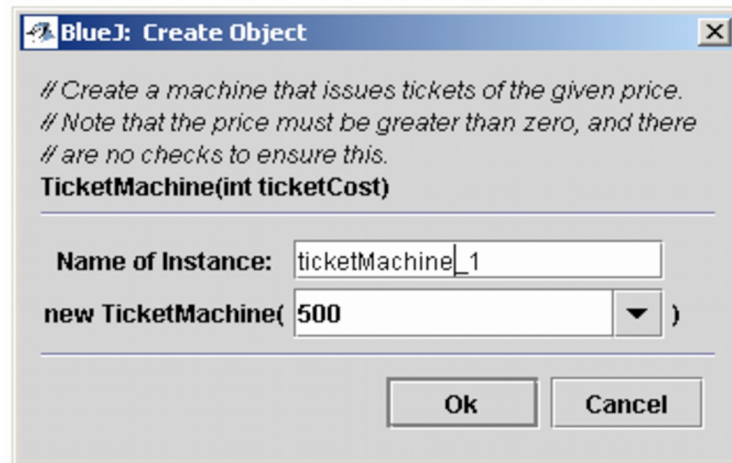


Constructores

- Los constructores inicializan un objeto.
 - Tienen el mismo nombre que su clase.
 - Almacenan valores iniciales en los campos.
 - A menudo reciben parámetros externos para esos valores.
 - Inicializar un objeto de la MaquinaDeBoletos.
- ```
public MaquinaDeBoletos(int precioDelBoleto)
{
 precio = precioDelBoleto;
 saldo = 0;
 total = 0;
}
```



# Pasado de datos via parámetros(A) y asignación(B)



Ej.: 2.16, 2.17, 2.18



# Asignación

- Los valores son almacenados en los campos (y otras variables) via instrucciones de asignación:
  - *variable = expresión;*
  - `precio = precioDelBoleto;`
- Una variable almacena un único valor por lo que cualquier valor previo se pierde.
- Ej.: 2.19, 2.20

# Métodos de Acceso

- Los métodos implementan el comportamiento de los objetos.
- Los de acceso proveen información acerca de un objeto.
- Los métodos tienen una estructura constituida por un encabezado y un cuerpo.
- El encabezado define la *signatura* del método.  

```
public int getPrecio()
```
- El cuerpo incluye las instrucciones del método.

# Métodos de Acceso

Modificador de visibilidad

Tipo retornado

Nombre del método

Lista de  
parámetros  
(vacía)

**public int getPrecio()**

Instrucción  
de retorno

**return precio;**

Comienzo y fin del cuerpo (bloque)



Ej.: 2.21, 2.22, 2.23, 2.24, 2.25, 2.26, 2.27

# Métodos Modificadores

- Tienen una estructura similar: encabezado y cuerpo.
- Usados para *modificar* el estado de un objeto.
- Se logra cambiando el valor de uno o más campos.
  - Típicamente contienen instrucciones de asignación.
  - Típicamente reciben parámetros.

# Métodos Modificadores

Modificador de visibilidad

Tipo retornado

Nombre del método

Parámetro

```
public void ingresarDinero (int cantidad)
{
 saldo = saldo + cantidad;
}
```

Campo que es mutado

Instrucción de asignación

Ej.: 2.29, 2.30, 2.31, 2.32

# Imprimir desde Métodos

```
public void imprimirBoleto()
{
 // Simula la impresión de un boleto.
 System.out.println("#####");
 System.out.println("# La Línea BlueJ");
 System.out.println("# Boleto");
 System.out.println("# " + precio + " cvos.");
 System.out.println("#####");
 System.out.println();
 // Actualiza el total recaudado con el saldo.
 total = total + saldo;
 // Pone en cero el saldo.
 saldo = 0;
}
```

Ej.: 2.33, 2.34, 2.35, 2.36, 2.37, 2.38



# Reflexión sobre la Máquina de Boletos Simple

- Su comportamiento es inadecuado de varias maneras:
  - No verifica los montos ingresado.
  - No da vueltos.
  - No se verifica una inicialización sensible.
- Cómo podemos mejorarla?
  - Se necesita un comportamiento mas sofisticado.
- Ej.: 2.39, 2.40, 2.41, 2.42

# Hacer elecciones

Abrir proyecto maquina-de-boletos-mejorada

```
public void ingresarDinero(int cantidad)
{
 if(monto > 0) {
 saldo = saldo + cantidad;
 }
 else {
 System.out.println("Use un monto positivo: " +
 monto);
 }
}
```

# Sentencia condicional

Clave if

condición lógica a ser verificada

Acciones a ejecutar si la condición es verdadera

```
if(se verifica una condición lógica) {
```

```
 Ejecuta estas instrucciones si da verdadero
```

```
}
```

```
else {
```

```
 Ejecuta estas instrucciones si da falso
```

```
}
```

Clave else

Acciones a ejecutar si la condición es falsa

Ej.: 2.43, 2.44, 2.45, 2.46, 2.47, 2.48, 2.49, 2.50, 2.51, 2.52

# Variables Locales

- Los campos son una suerte de variables.
  - Almacenan valores durante la vida de un objeto.
  - Se pueden acceder dentro de la clase.
- Los métodos pueden incluir variables con vida más corta.
  - Existen en tanto el método se esté ejecutando.
  - Sólo se pueden acceder dentro del método.

# Variables locales

Una variable local

No hay modificador  
de visibilidad

```
public reintegraSaldo()
{
 int cantidadAReintegrar;
 cantidadAReintegrar = saldo;
 saldo = 0;
 return cantidadAReintegrar;
}
```

Ej.: 2.53, 2.54

# Repaso

- Los cuerpos de las clases contienen campos, constructores and métodos.
- Los campos almacenan valores que determinan el estado de un objeto.
- Los constructores inicializan un objeto.
- Los métodos implementan el comportamiento de un objeto.



# Repaso

- Los campos, parámetros y las variables locales son todos variables.
- Los campos persisten durante la vida de un objeto.
- Los parámetros son usados para recibir los valores en un constructor o en un método.
- Las variables locales son usadas para un almacenamiento de corta vida.

# Repaso

- Los Objetos pueden tomar decisiones via instrucciones condicionales (if).
- Un test con resultado verdadero o falso permiten tomar uno o dos cursos de acción alternativos.