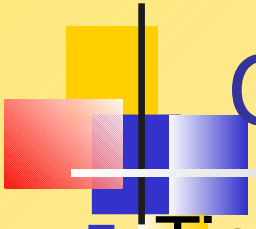
UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Objetos y Clases en Java

ELO-329: Diseño y Programación
Orientados a Objetos

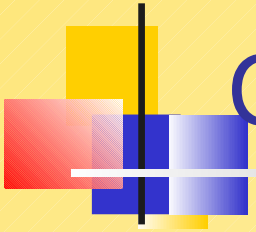
Creación de objetos nuevos

- Se usa el constructor de la clase
  MiClase a = new MiClase();
- Todos los objetos son creados en el heap (memoria asignada dinámicamente durante la ejecución).
- Lo que se retorna es una referencia al nuevo objeto (puede ser pensada como puntero).
- Nota — no existe destructor (en C++ sí)
  Java tiene un proceso de recolección de basura (Garbage Collection) que automáticamente recupera zonas no referenciadas.



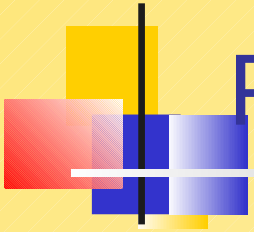
Constructores

- Tiene igual nombre que la clase
- Pueden tener parámetros
- Son invocados principalmente con **new**
- No tiene tipo retornado
- No **return** explícito
- Java provee constructor por defecto **()**
- Podemos proveer uno o más constructores. Esto es un tipo de sobrecarga de métodos (igual nombre con distintos parámetros)
- El compilador busca el constructor usando firma nombre constructor + lista de parámetros



Constructores

- Inicializa objetos nuevos:
  - 1. Localiza memoria
  - 2. Asigna valores por defecto a variables (0, 0.0, null, …)
  - 3. Llama constructor de Superclase (más adelante)
  - 4. Sentencias restantes son ejecutadas
- La primera sentencia puede ser:
  - super(…) para llamar al constructor de la clase base (o padre o superclase)
  - this(…) invoca a otro constructor



Referencias

- Los objetos son referenciados
- Esta es una forma "controlada" de usar: Direcciones y punteros
- Al declarar una variable de una clase obtenemos una referencia a la variable.
- En caso de tipos primitivos (8) se tiene la variable y acceso directo (no es referencia)
  - byte, short, int, long, float, double, char, boolean

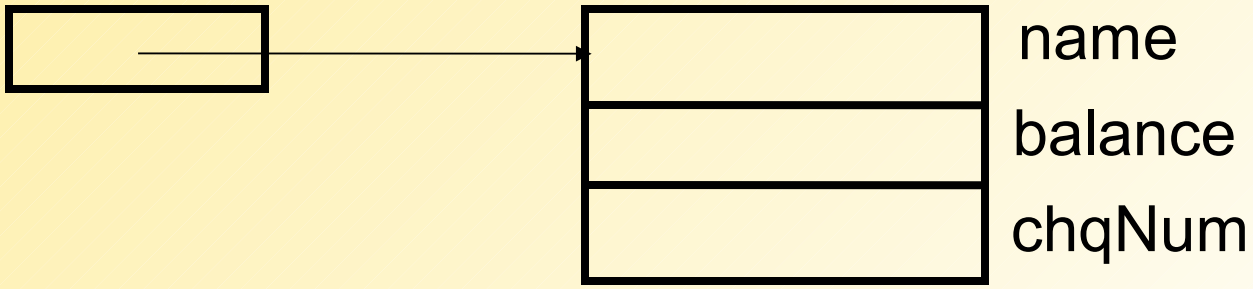
Definiendo variables

Cheque pejAcct;

pejAcct  Referencia nula

pejAcct.deposit(1000000); // error

pejAcct = new Cheque("Peter", 1000, 40);

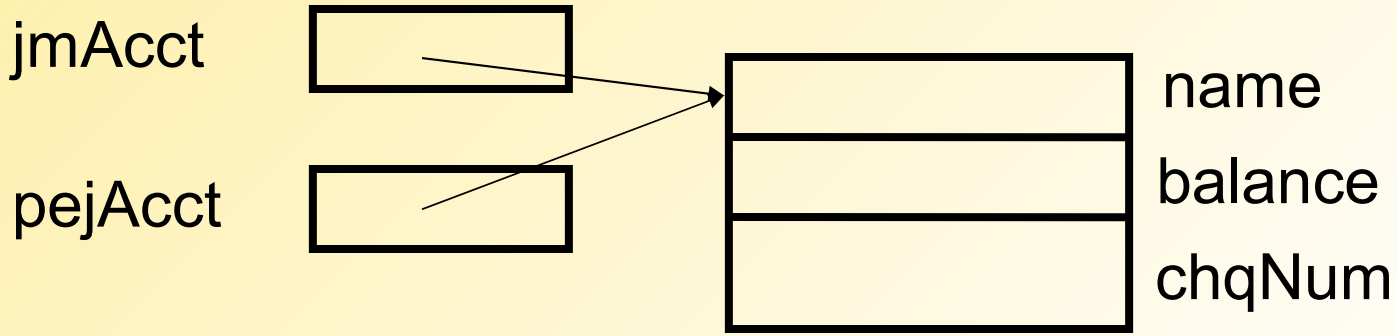


Este ejemplo asume que la clase Cheque ya existe y posee miembros datos: name, balance y chqNum

Asignación

Cheque jmAcct;

jmAcct = pejAcct;



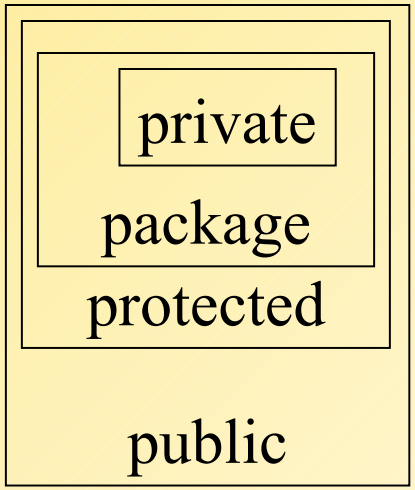
Implicancias de referencias

- La identidad de objetos son **referencias**
  - referencia significa puntero (i.e. no el contenido)
- = es copiar la referencia
  - Usar método **clone** para crear copia del objeto completo.
- == es comparación de referencias
  - Usar **equals** para comparar contenidos
- aMethod(pejAcct) pasa un referencia
- aMethod(tipo_básico) pasa el valor
- return pejAcct retorna una referencia
  - Usar clone para crear una copia, y luego retornarla

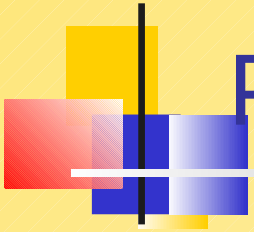
Control de acceso

- *Modificador de acceso*
- public
- protected
- "omitido"
- private

- *Visibilidad*
- Todas partes
- en sub-classes & pkg
- En el paquete
- Sólo en la clase



Modificador	Clase	Package	Subclass	World
public	si	si	si	si
protected	si	si	si	no
omitido	si	si	no	no
private	si	no	no	no



Paquetes en Java (package)

- Existen para garantizar unicidad en los nombres de clases.
- Si queremos referirnos a la clase Date, podemos usar:
  - java.util.Date hoydia = new java.util.Date();
  - Una forma reducida es usar:
  - import java.util.Date;
  - Date hotdia = new Date();
- Si deseamos usar varias clases de un mismo paquete:
  - import java.util.*;

Paquetes en Java (cont.)

- Para incluir una clase en un paquete, al inicio del archivo indicar:
  - Package cl.utfsm.elo.elo329
- Esto implica que debe existir los directorios: cl, dentro de él utfsm, dentro de éste elo y finalmente el directorio elo329. En este último ponemos los archivos del paquete.
- Para compilar estos archivos usamos:
  - javac cl/utfsm/elo/elo329/archivo.java
- Para correr el archivo usamos:
  - java cl.utfsm.elo.elo329.archivo

Documentación

- Para la clase ponerla inmediatamente antes de la clase y ser encerrado entre /** y */
- Para los métodos: usar los rótulos
  - @param *variable* descripción
  - @return descripción
  - @throws descripción de clase
- Para los datos públicos: /** ...*/
- Comentarios Generales:
  - @author nombre
  - @version texto
  - @since texto
  - @see link
  - Ejemplo: @see cl.utfsm.elo.Employee#raiseSalary(double)

Documentación

- Se pueden usar todo tipo de rótulos html incrustados.
- ¿Cómo generar la documentación?:
  - javadoc -d docDirectory *.java
- Para la documentación de un paquete:
  - javadoc -d docDirectory nameOfPackage
- Ejemplo:
  - Account.java
  - index.html generado con javadoc -d AccountDoc *.java

Rutas para clases

- Primero incluir la ruta del compilador y máquina virtual java en la variable PATH.
- Luego la ruta para la búsqueda de todas las clases: CLASSPATH
  - El compilador y el interprete java buscan los archivos en el directorio actual.
  - Si el proyecto está compuesto por varias clases en diferentes directorios, javac y java buscan las clases en los directorios indicados en la variable de ambiente CLASSPATH.
- En Linux ELO ésta se configura con
  - export CLASSPATH=/home/user/classdir1: /home/user/classdir2:.
- El Windows también se debe fijar la variable de ambiente.