Tema 3b: Estructura General de un programa en Java

Modificadores de acceso

- Son keywords en Java que dan información al compilador sobre el ámbito en el que una clase, un método y/o una variable pueden manejarse por su nombre o bien que atributo poseen.

- Los modificadores de acceso son:
  - ✓ private
  - ✓ default
  - ✓ protected
  - ✓ public

Modificadores de acceso

- private class Uno
  - ✓ La clase sólo puede referenciarse dentro del mismo archivo en el que se define.
  - ✓ Pueden definirse dentro de otras clases en el mismo archivo.
  - ✓ No recomendable.
- En un método
  - ✓ Sólo puede referenciarse (ejecutar) dentro del mismo archivo en el que se define.
  - ✓ Se pueden definir métodos private dentro de cualquier clase.
  - ✓ No puede heredarse.
- private variable
  - ✓ Sólo puede usarse (leer/escribir) dentro del mismo archivo donde se define. No puede heredarse (no accesible a la subclase).

Modificadores de acceso

- default (class Uno)
  - ✓ Puede usarse en el mismo paquete.
  - ✓ No se escribe

Visibilidad	private	default	protected	public
Desde la propia clase	Si	Si	Si	Si
Desde otra clase en el propio paquete	No	Si	Si	Si
Desde otra clase fuera del paquete	No	No	No	Si
Desde una subclase en el propio paquete	No	Si	Si	Si
Desde una subclase fuera del propio paquete	No	No	Si	si

Modificadores de acceso

- protected class Uno
  - » La clase puede referenciarse desde cualquier parte del código dentro del mismo paquete y además en cualquier clase heredada así como las subclases de este clase.
  - » Puede haber una o mas clases con este modificador.
- Método
  - » Referencia en el mismo paquete.
- protected variable
  - » Referencia en el mismo paquete. Es recomendable hacer uso de private.

Modificadores de acceso

- Public class Uno
  - » Puede referenciarse desde cualquier parte del código no importando el paquete en el que esté.
  - » Puede haber cero o una clase de este tipo.

- Public class Uno
  - » Puede referenciarse desde cualquier parte del código no importando el archivo o paquete donde se ubique.

Modificadores de comportamiento

- Son** :
  - ✓ static
  - ✓ final
  - ✓ abstract
  - ✓ Synchronized
  - ✓ native
  - ✓ volatile
  - ✓ transient
  - ✓ stritcfp

Clase

- Es el modelo de un objeto que se está describiendo.

- Sintaxis:

```
[mod. acceso][mod. Comportam.] class Identificador{
   // declaración de atributos
   // declaración del constructor
  // declaración de métodos.

}
```

Método

- Es el elemento funcional de un objeto.

- Sintaxis:

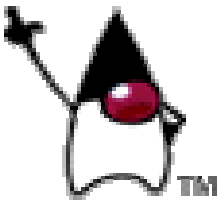
```
[mod. acceso][mod. Comportam.]
    <tipo_retorno>Identificador ([argumentos]){
   // declaración de variables
   // acciones.

}
```

Objeto

- Es una instancia actual de una clase, cada objeto se obtiene con la keyword new.

- Referencia

  Puede concebirse como un apuntador al objeto.



Objeto

- Instancia

  Es una variable de tipo class denominada objeto.

- Para crearla se debe direccionar a un apuntador, esta operación es llamada referencia.

- Sintaxis:

NomClase Identificador =

new NomClase([parámetros]);

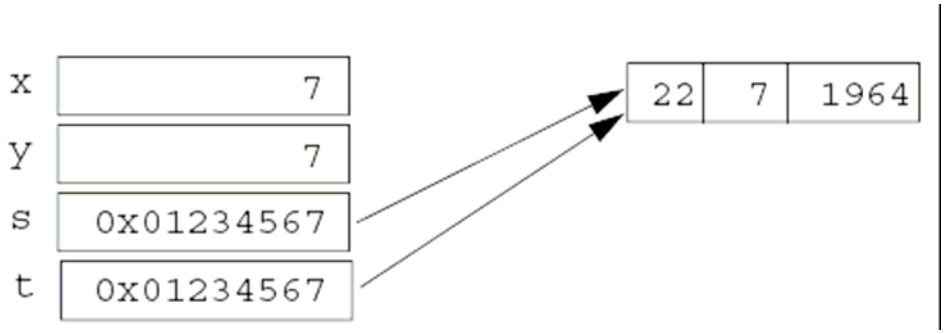
- Ejemplo:

ClaseUno obj=new ClaseUno();

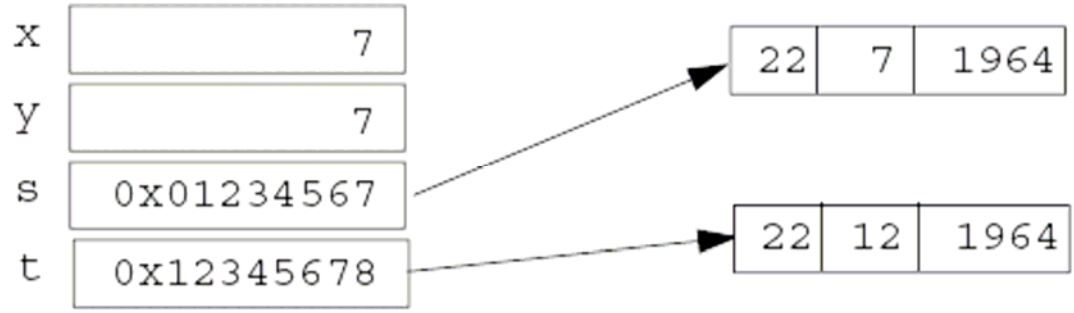
Objeto

- El stack y el heap

```
int x = 7;
int y = x;
MyDate s = new MyDate(22, 7, 1964);
MyDate t = s;
```




```
t = new MyDate(22, 12, 1964); // reassign the variable
```



Constructor

- Es un conjunto de instrucciones diseñadas para inicializar una instancia.

- Sintaxis:

```
[mod. acceso] NombreClase ([argumentos]){
  // acciones.

}
```

- Todas las clases contienen uno por default (no se “ve”) el cuál no tiene retorn, ni cuerpo, ni parámetros.

Paquetes

- Cuando se tienen muchas clases pueden usarse los paquetes, dónde las clases están físicamente en directorios diferentes.

- Sintaxis:

  package directorio.subdirectorio1,subdirectorioN;

- Debe de ir en la 1ra línea.

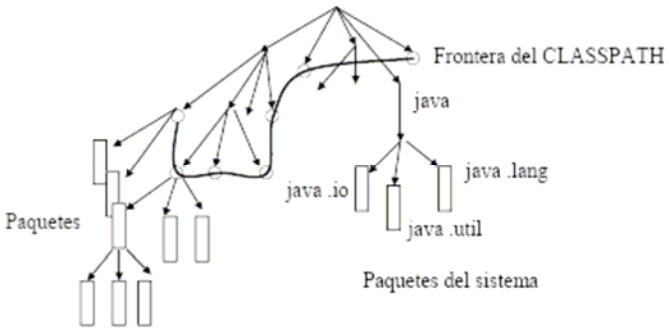
Paquetes

- Cuando se tienen muchas clases pueden usarse los paquetes, dónde las clases están físicamente en directorios diferentes.

Paquetes

- Las bibliotecas se organizan en *paquetes* (`package`): mecanismos lógicos para agrupar clases relacionadas.
- Todas las clases de un paquete deben estar localizadas en un mismo subdirectorio.
- Los paquetes del sistema cuelgan de varios subdirectorios específicos:
  - `.../java`
  - `.../javax`
- La variable `CLASSPATH` contiene una lista con todos caminos de búsqueda de los demás paquetes.

Estructura de las bibliotecas en Java



La palabra reservada import

- Cuando se usan paquetes se requiere
  la keyword *import* para indicar al compilador dónde encontrar las clases utilizadas.

- Sintaxis:
- import directorio.subdirectorioN.nombreClase;
- import directorio.subdirectorioN.*;

Ejemplos

- ✓ package mipaquete;
import subdirectorio.*;
class MiClase{ ... }

- ✓ //comentario
package mipaquete;
import subdirectorio.*;
class MiClase{ ... }

- ● package mipaquete;
//comentario
import subdirectorio.*;
class MiClase{ ... }

- ✓ //comentario
import subdirectorio.*;
class MiClase{ ... }

Tipos de variables

- Variables locales
  - ✓ Son las declaradas dentro de un método o las que se tienen como parámetros o en bloques de código.
  - ✓ Llamadas también variables automáticas, temporales o stack.
  - ✓ Su tiempo de vida es hasta que el método existe.

- Variables de referencia
  - ✓ Son las definidas fuera de un método y son creadas cuando el objeto es construido usando new.

Tipos de variables

- Clasificación variables referencia:

- V. de clase

  Son de claradas usando static. Estas variables son creadas cuando la clase está cargada y el tiempo de vida es el tiempo de vida de la clase.

- V. de instancia

  Son llamadas también como variables miembro. La vida de ellas es hasta que exista el objeto.