

Resumen del capítulo 1: Declaraciones y Accesos de control

- **Clase:** Un modelo que describe el tipo de estado y comportamiento que los objetos de su tipo soportan
- **Objeto:** Cuando la máquina virtual de Java (JVM) encuentra nuevas palabras, ésta las usa para la apropiación de la clase para crear un objeto que es una instancia de esa clase.
- **Estado (instancia de variables):** Cada objeto (instancia de una clase) que tendrá su propio conjunto único de instancia de variables como definido en la clase.
- **Métodos:** Cuando un programador crea clases, se crea métodos para la clase. Los métodos son donde las lógicas de las clases se guardan, son donde los algoritmos son ejecutado y manipula los datos.

Identificadores y palabras clave

Todos los componentes de Java justamente hablando acerca de Clases, variables y métodos necesitan nombres. En Java esos nombres son llamados identificadores

Herencia

Definido como una clase se reusa en otra clase. En Java, lo puedes definir en general (mas abstracto) superclases y estos se extiende con subclasses más específicas. La superclass no sabe nada de las clases que la heredan, pero todas las subclasses que hereda las superclass debe declarar explícitamente la relación de herencia.

Una subclase que hereda desde una superclase es automáticamente dado accesible a variables de instancia y métodos definidos por la superclase, pero también es libre de anular métodos de las superclases para definir más comportamientos en específico.

Interfaces

Un poderoso compañero para la herencia es el uso de las interfaces. Las interfaces son como una Superclase abstracta al 100% que define los métodos que una subclase debe admitir, pero no como ellos deben ser soportados. En otras palabras, una interfaz Animal prodría declarar que todas las clases de implementación de animales tienen un método comer(), pero la interfaz Animal no proporciona alguna lógica para el método comer().

Encontrando otras clases

Para cada clase debe tener enfocada su responsabilidad, para el que fue creado.

Identificadores y Java Beans

Identificadores legales: Las reglas del compilador usan para determinar ya sea si un nombre es legal.

Reglas de denominación de JavaBean Listener

- Los nombres de los métodos de escucha utilizados para registrar un **listener** con un origen de evento deben usar el prefijo **add**, seguido del tipo **listener**. Por ejemplo : **addActionListener()** es un nombre válido para eventos de acción.
- Los nombres de los métodos de escucha utilizados para eliminar (anular el registro) un **listener** debe usar el prefijo **remove**, seguido del tipo **listener** (usando las mismas reglas que el método de agregar registro).
- El tipo **listener** que se agregará o eliminará se debe pasar como el argumento **el método**.
- Los nombres de los métodos de escucha deben terminar con la palabra "Escucha!"

Declaración de reglas en Archivos fuente

- Allí puede estar solo una clase pública por archivo código fuente.
- Los comentarios pueden aparecer en el inicio o al final de cualquier linea en el archivo código fuente.
- Si está allí una clase pública en el archivo, el nombre del archivo debe de tener el nombre de la clase.
- Si la clase es parte de un paquete, la declaración del paquete debe ser la primera linea en el archivo de código fuente, antes de cualquier declaración de importación.
- Un archivo puede tener más de una clase no pública.
- Los archivos sin clases públicas pueden tener un nombre que no coincida con ningún de las clás en el archivo.

Declaraciones de clases y modificadores

Modificadores de acceso: public, protected, private

Modificadores no de acceso: strictfp, final, abstract y etcétera.

Clases de acceso

Acceso significa visibilidad, el ejemplo más común es cuando una clase accede a otra clase y puede instanciar con las variables o métodos dependiendo de su modificador de acceso

Acceso default: ningún modificador que lo precede en la declaración, esto obtienes el acceso de control cuando no se escribe un modificador en la declaración de clase. Piensa del Acceso default como un acceso de nivel paquete, porque unas classes con acceso de fault solo pueden por clases con el mismo paquete.

Acceso public: todas las clases en el universo Java tienen acceso a las clases públicas. Si tratas de usar un paquete diferente desde la clase tu necesitas importarlo de la clase public.

Ejemplo:

```
package cert;  
public class Beverage {  
    // insert todo logic here  
}
```

- Los identificadores deben iniciar con una letra, el símbolo de pesos (\$), o un conector de carácter como el guion bajo (_). Los identificadores nunca se puede iniciar con un número.
- Despues del primer carácter, pueden contener alguna combinación de letras, simbolos de moneda o números
- En práctica, no hay límite para el número de caracteres que un identificador puede tener.
- No puedes usar las palabras clave de Java.
- Los identificadores de Java son distinguidos de mayúsculas y minúsculas.

Ejemplos de identificadores legales

```
int a;  
int $c;  
int --75-2-W;  
int this_is_a_very-detailed-name_for_an_identifier;
```

Ejemplos de identificadores ilegales

```
int :b;  
int -d;  
int 7g;
```

Clases e Interfaces: La primera letra debe ser en mayúsculas, y si varias palabras están unidas para formar el nombre, la primera letra de la siguiente palabra debe ser en mayúsculas

Métodos: La primera letra debe ser en minúscula, y para las palabras siguientes que forman el nombre, la primera letra en mayúscula, algunos ejemplos

getBalance

hacerCalculo

Variables: Como en los métodos.

Constantes: Constantes en Java son creadas por marcadores static y final. Se deben nombrar usando letras mayúsculas con el guion bajo como separador:

MIN-HEIGHT

Java Bean, Reglas de nombre de propiedad

- Si la propiedad no es booleana, El prefijo del método obtener debe ser get, por ejemplo, getSize() es un nombre obtenido válido de JavaBeans para la propiedad nombrada size.
- Si la propiedad es un booleano, El prefijo de los métodos Obtener es o bien get o is
- Los prefijos de los métodos poner deben ser set. Por ejemplo, setSize() es un nombre válido de JavaBeans para la propiedad nombrada size.
- Para completar el nombre de un método de obtención o establecimiento, cambie la primera letra del nombre de la propiedad a mayúsculas y luego agrégala al prefijo apropiado
- Los métodos de establecimiento deben ser marcados public, con un tipo de retorno vacío.