

Paquetes

Jorge I. Meza

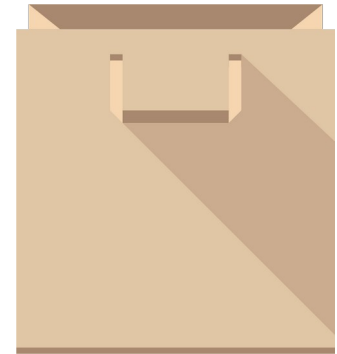
jimezam@autonoma.edu.co

Contenidos

- Definición
- Ventajas de su uso
- Reglas de nombrado
- Ejemplos
- Crear un nuevo paquete
- Usar clases del mismo paquete
- Usar clases de otros paquetes
- Acerca del paquete `java.lang`

Definición de Paquetes

- Son organizadores de clases. Permiten **agrupar las clases comunes** para hacer el código más organizado y fácil de gestionar.
- Es posible crear subpaquetes dentro de paquetes o de otros subpaquetes.
- Toda clase debe pertenecer a un paquete, no es buena práctica utilizar el *paquete por defecto*.
- La ubicación de los archivos de clases en subdirectorios debe imitar al paquete al cual pertenece.



Ventajas

- **Organización** al agrupar clases relacionadas, mejorando la legibilidad y mantenibilidad del código.
- **Evitar colisión de nombres** mediante la reducción de la probabilidad de que dos clases tengan el mismo nombre.
- **Jerarquía** al estructurar el nivel de las clases del proyecto.
- **Acceso controlado** mediante modificadores de acceso (`public`, `protected`, `private`).
- **Reutilización** al encapsular clases comunes.

Reglas de nombrado

- Todo en minúsculas
- Palabras separadas por puntos
- Dominio invertido
- Ejemplos
 - `com.ejemplo.proyecto`
 - `org.apache.commons`
 - `net.sourceforge.myapp`

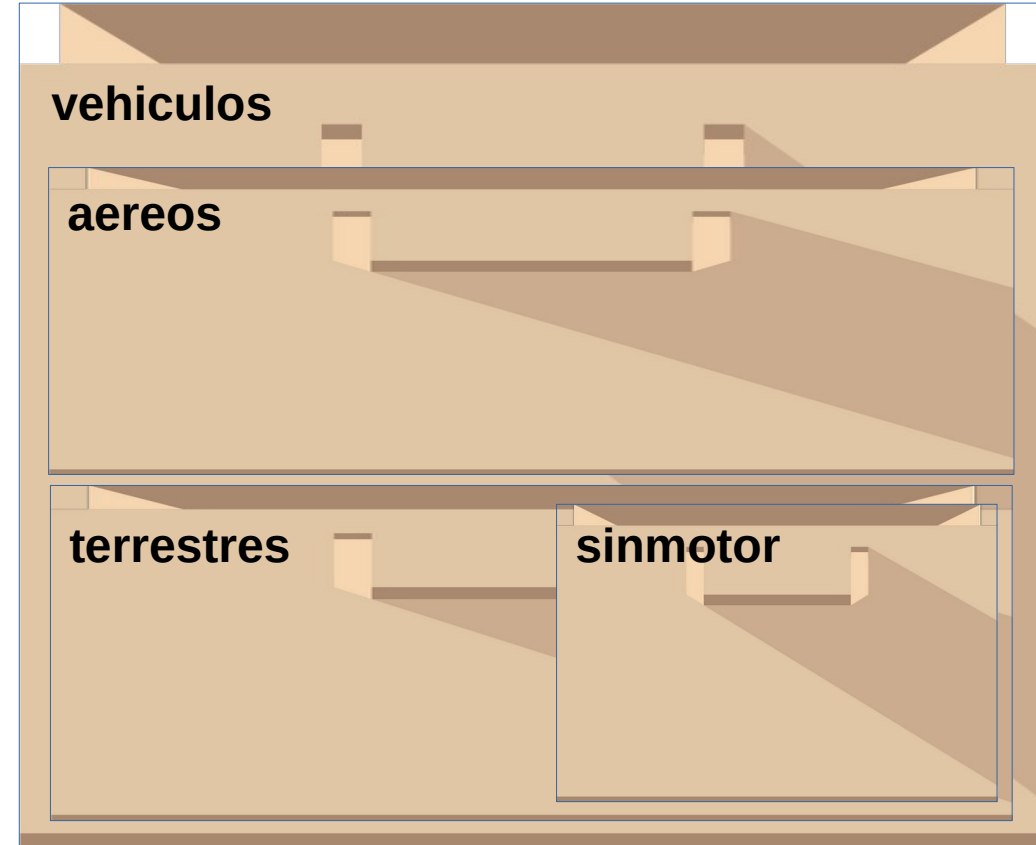


Reglas de nombrado

- **Ser descriptivo:** El nombre del paquete debe reflejar de manera clara el contenido de las clases que contiene.
- **Evitar nombres genéricos:** Utilizar nombres específicos y significativos para evitar ambigüedades.
- **Ser consistente:** Aplica las mismas reglas de nombrado a todos los paquetes de tu proyecto.

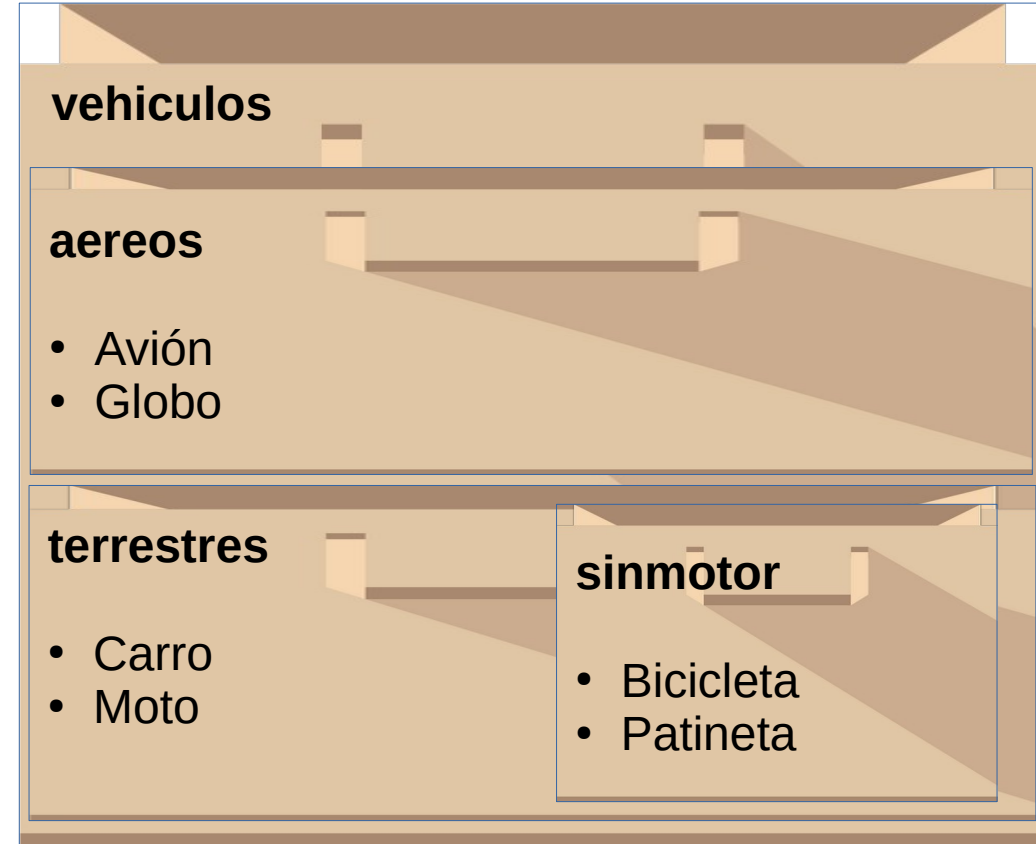
Ejemplos de Paquetes

- `vehiculos`
- `Vehiculos.aereos`
- `vehiculos.terrestres`
- `vehiculos.terrestres.sinmotor`



Ejemplos de Paquetes

- `vehiculos`
- `Vehiculos.aereos`
- `vehiculos.terrestres`
- `vehiculos.terrestres.sinmotor`



Ejemplos de Paquetes

- `granja`
- `granja.verduras`
- `granja.animales`
- `granja.animales.servicio`
- `granja.animales.domesticos`



Crear un Paquete

- La primera línea del código de una clase debe ser la sentencia `package` seguida por el nombre del paquete al cual pertenece dicha clase.

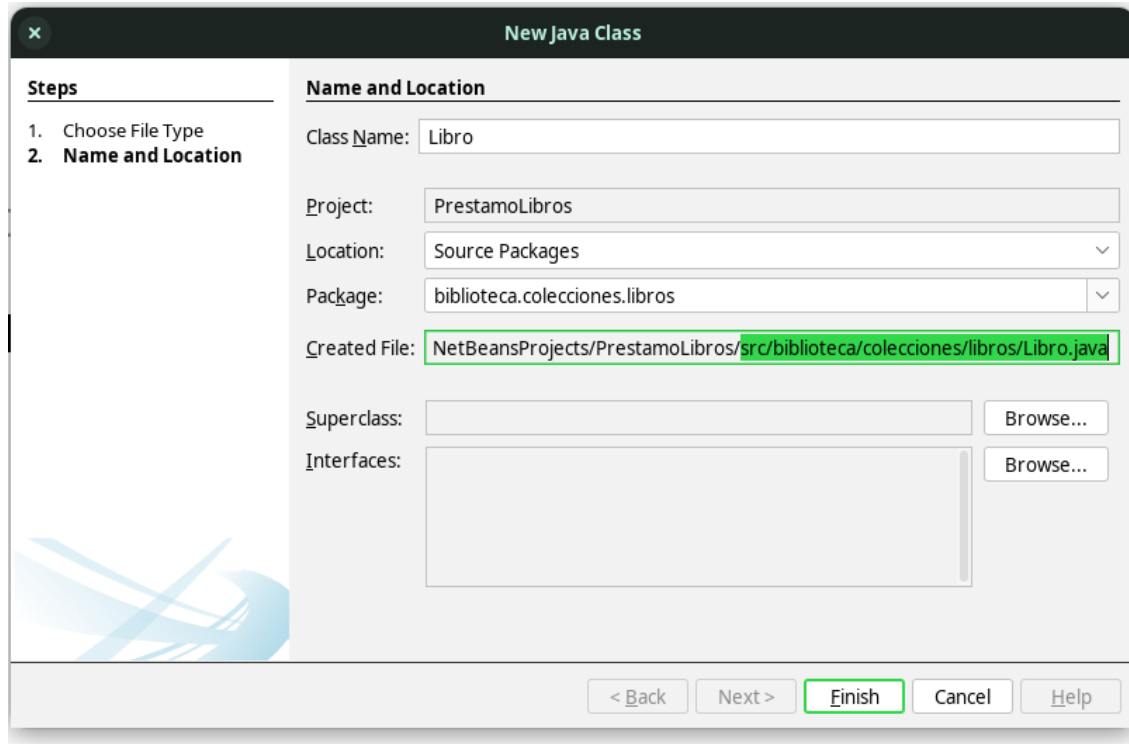
Por ejemplo:

```
package biblioteca.colecciones.libros;
```

Crear un Paquete

- La ruta del paquete debe coincidir con la ruta de directorios en la cual se guarda el archivo de la clase.
- Si se utiliza un IDE como Netbeans, él realiza esta tarea de manera automática.

Crear un Paquete



New Java Class

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

Project:

Location:

Package:

Created File:










Superclass:

Interfaces:

Crear un Paquete

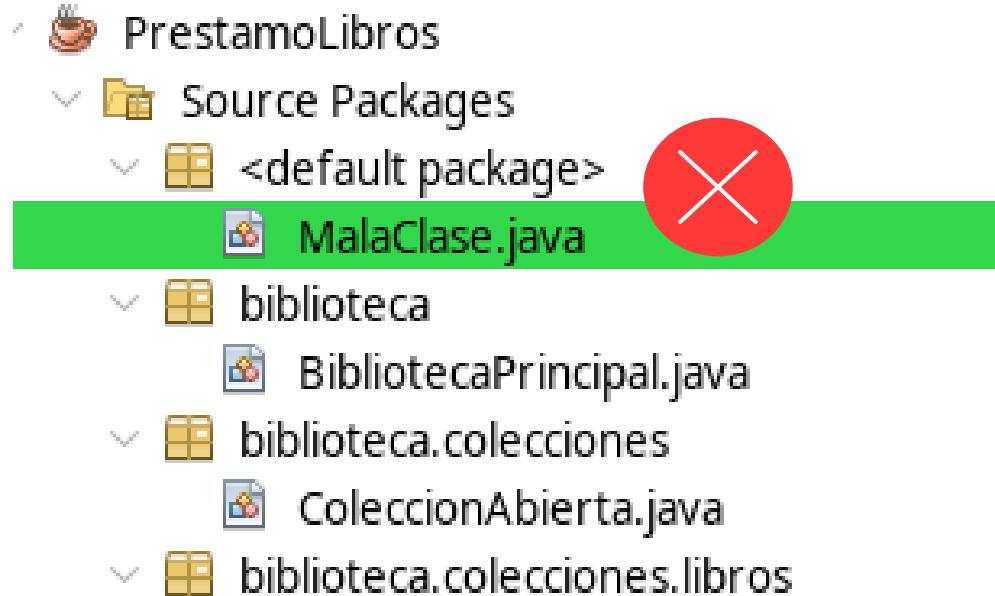
```
1 package biblioteca.colecciones.libros;  
2  
3 public class Libro {  
4     // ...  
5 }
```

Crear un Paquete

- ✓  PrestamoLibros
 - ✓  Source Packages
 - ✓  biblioteca
 -  BibliotecaPrincipal.java
 - ✓  biblioteca.colecciones
 -  ColeccionAbierta.java
 - ✓  biblioteca.colecciones.libros
 -  Libro.java
 - >  Libraries

Crear un Paquete

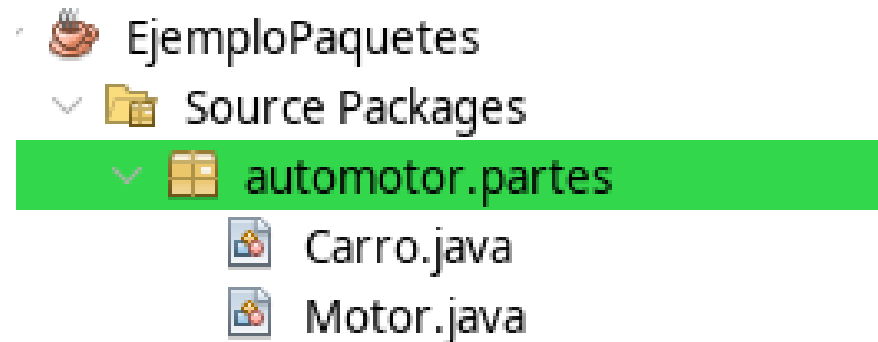
- **Nunca** ubicar clases en el *paquete por defecto*.



Mala práctica

Utilizar clases del mismo paquete

- Dos clases que pertenecen al mismo paquete pueden utilizarse entre si directamente.
- Suponga que `Carro` y `Motor` pertenecen al paquete `automotores.partes`.



Utilizar clases del mismo paquete

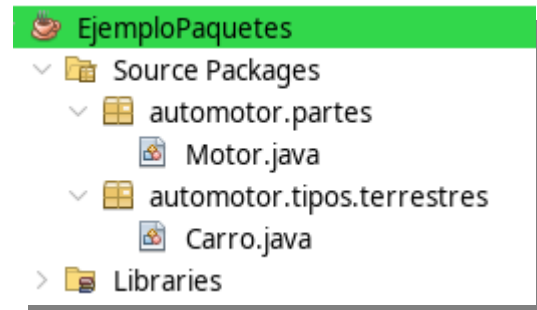
```
1  package automotor.partes;  
2  
3  public class Carro {  
4      private Motor motor;  
5  
6      public Carro() {  
7          motor = new Motor();  
8      }  
9  }
```

Utilizar clases de otro paquete

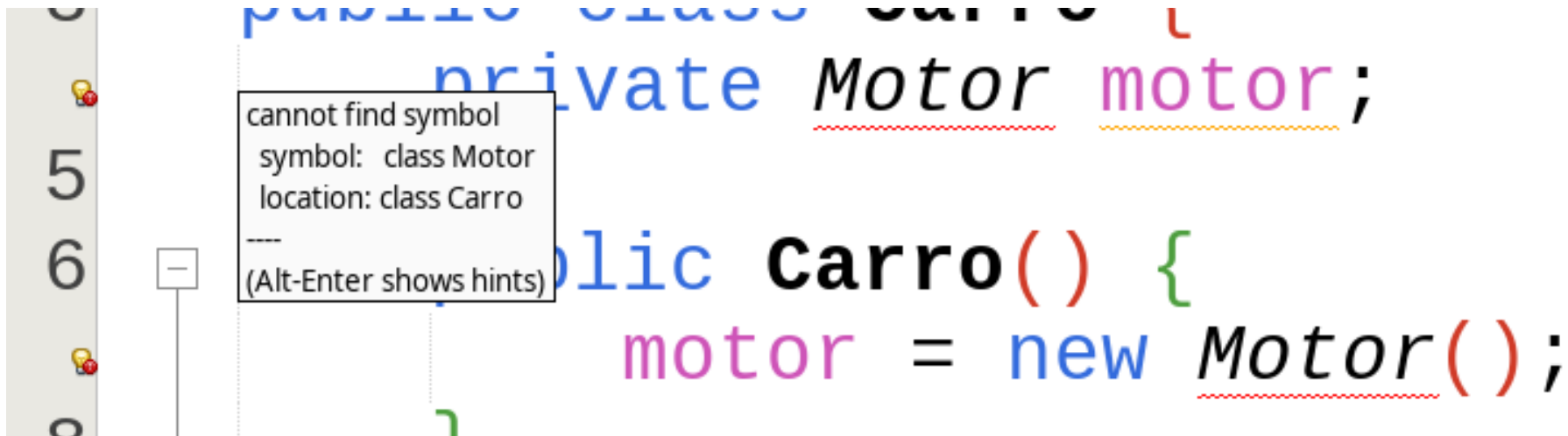
- Dos clases que pertenecen a paquetes diferentes, no pueden utilizarse entre si directamente.
- La clase que planea utilizar a la otra, debe *importarla*.
- Para hacer esto, después de la sentencia `package`, debe agregar una o más sentencias `import` con el nombre completo de la clase (*paquete.nombre*).

Utilizar clases de otro paquete

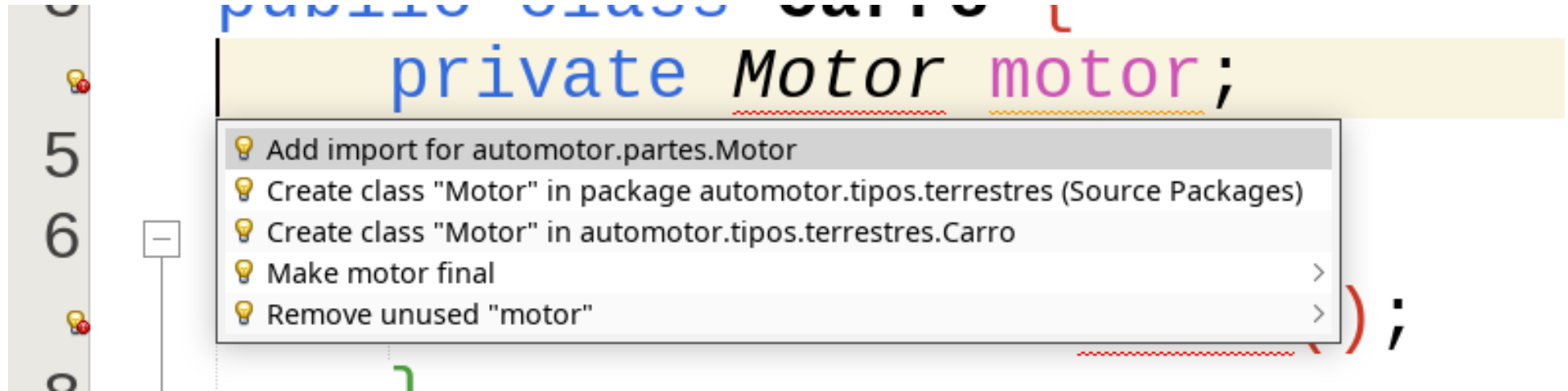
```
1 package automotor.tipos.terrestres;  
2  
3 public class Carro {  
4     private Motor motor;  
5  
6     public Carro() {  
7         motor = new Motor();  
8     }  
9 }
```



Utilizar clases de otro paquete



Utilizar clases de otro paquete



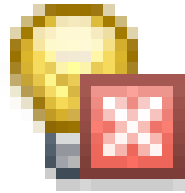
Utilizar clases de otro paquete

```
1  package automotor.tipos.terrestres;  
2  
3  import automotor.partes.Motor;  
4  
5  public class Carro {  
6      private Motor motor;  
7  
8      public Carro() {  
9          motor = new Motor();  
10     }  
11 }
```

¡Cuidado con los bombillos!



Warning



Error

- **¡Son sugerencias!**
¡El desarrollador es usted!
- No acepte una solución si no es la que usted requiere o no conoce que significa.

Paquete `java.lang`

- Las clases del paquete `java.lang` tienen una característica particular y es que se importan automáticamente, haciendo que sus clases se encuentren disponibles en cualquier momento.
- A continuación se listan algunas de las principales clases encontradas en este paquete.

<https://docs.oracle.com/en%2Fjava%2Fjavase%2F11%2Fdocs%2Fapi%2F%2F/java.base/java/lang/package-summary.html>

Paquete `java.lang`

- Interfaces:
`Comparable` (comparar objetos), `Iterable` (crear colecciones), `Runnable` (hilos).
- Clases:
 - Clases de recubrimiento: `Boolean`, `Byte`, `Character`, `Double`, `Float`, `Integer`, `Long`, `Short`
 - Enumeraciones: `Enum`
 - Operaciones matemáticas: `Math`

Paquete `java.lang`

- Clases:
 - La super clase de toda clase en Java: `Object`
 - Manejo de cadenas de caracteres: `String`
 - Interacción con el sistema, incluyendo la impresión de caracteres a través de la consola: `System`
 - Creación de hilos de ejecución: `Thread`

Paquete `java.lang`

- **Excepciones:**
 - **Excepciones comunes como** `ArithmeticException`,
`ArrayIndexOutOfBoundsException`,
`NegativeArraySizeException`,
`NullPointerException`,
`StringIndexOutOfBoundsException`.
 - **Clases base para la creación de Excepciones:**
 - `Exception`
 - `RuntimeException`
 - `Error`

¿Alguna



pregunta?