

PAQUETES Y MODIFICADORES DE ACCESO



¿QUÉ ES UN PAQUETE?

- Unidad organizativa, que puede contener una o más clases (o interfaces) relacionadas desde un punto de vista lógico.
- Evita conflicto de nombres
- Proporciona protección de acceso.
- Sintaxis: palabra reservada **package**

```
package nombrepaquete; //todo en minúsculas
```

- Pueden crearse subpaquetes.

```
package iesteis.xestion.persoas;
```



EJEMPLO DE PAQUETE

The screenshot displays an IDE with two main panels. On the left, the 'Projects' panel shows a project named 'Bienvenida' with a source package 'ejemplos' containing the file 'Bienvenida.java'. On the right, the 'Bienvenida.java' editor shows the following code:

```
1  /*
2   * Ejemplo de uso de paquetes
3   */
4
5  package ejemplos;
6
7  /**
8   *
9   * @author FMA
10 */
11 public class Bienvenida {
12
13
14     public static void main(String[] args) {
15
16         System.out.println ("Bienvenido a Java");
17
18     }
19
20 }
21
```

Annotations and callouts in the image:

- A red box highlights 'ejemplos' in the project tree, with a callout 'Proyecto con un paquete: ejemplos'.
- A red box highlights 'Bienvenida.java' in the project tree, with a callout 'Clase incluida en el nuevo paquete'.
- A blue box highlights 'package ejemplos;' in the code, with a callout 'Paquete "ejemplos"'.
- A red box highlights 'public class Bienvenida {' in the code, with a callout 'Clase incluida en el nuevo paquete'.



USO DE PAQUETES

`java.lang` se importa automáticamente

Desde fuera de un paquete sólo se puede acceder a clases e interfaces **públicas**. Para acceder a las clases de otro paquete:

- Importamos la clase

```
import java.util.Random; // o todas con java.util.*  
...  
Random aleatorio = new Random();
```

- o utilizamos el nombre de la clase cualificado con el nombre del paquete

```
java.util.Random aleatorio = new java.util.Random();
```



MODIFICADORES DE ACCESO

Nivel de visibilidad de las clases, métodos y atributos

Modificador	Clase	Paquete	Subclase	Todos
public (público)				
protected (protegido)				
<i>Sin modificador</i> (friendly)				
private (privado)				



Sí puede acceder



No puede acceder



MODIFICADORES DE ACCESO

Las **Clases** deben ser:

- **public**: cualquiera podrá utilizarlas
- **Sin modificador**: solo las clases del mismo paquete

Los **métodos** deben ser lo más restrictivos:

- **public** para los interlocutores con el exterior
- **private** para métodos auxiliares de otros métodos

Los **atributos** deben ser:

- **private**, salvo para constantes

