API Java y Paquetes

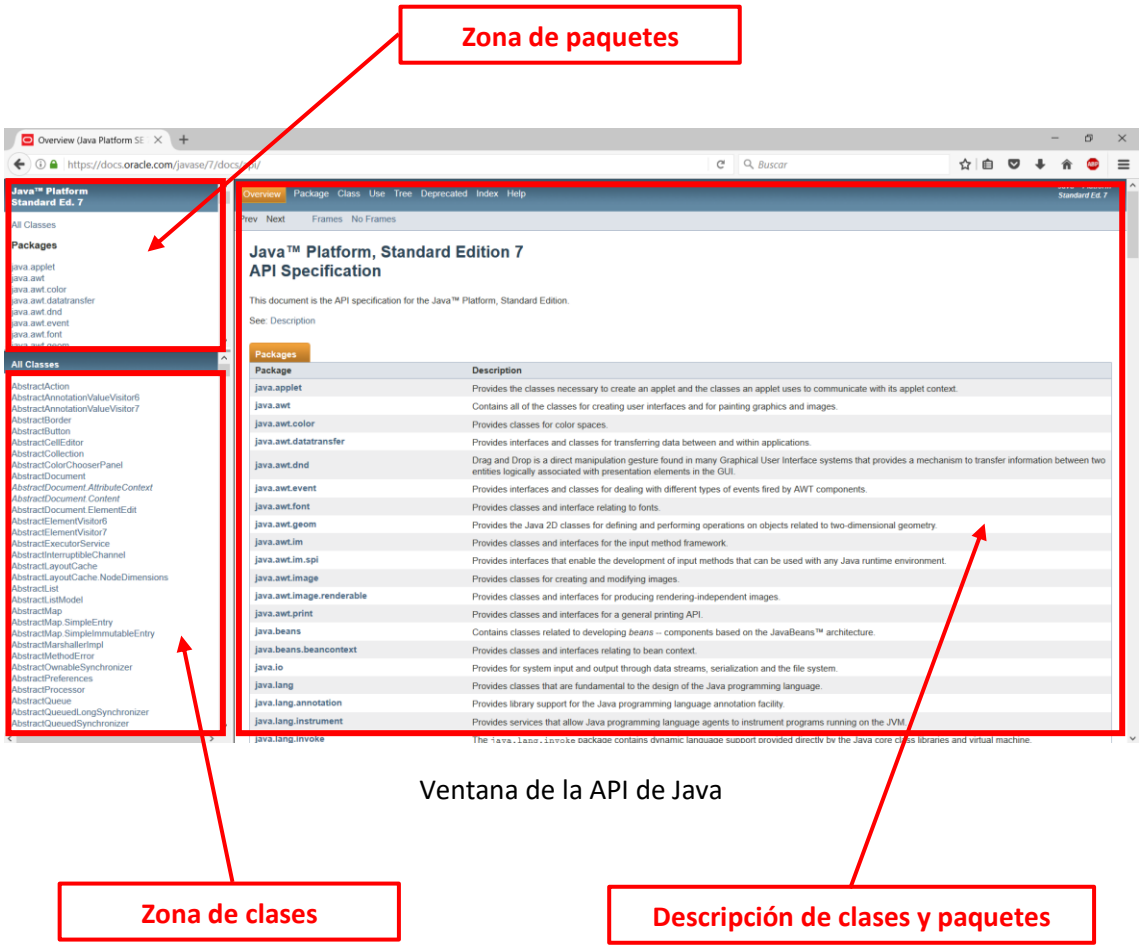
API Java

Resulta prácticamente imposible para cualquier programador recordar todos los métodos y clases de Java.

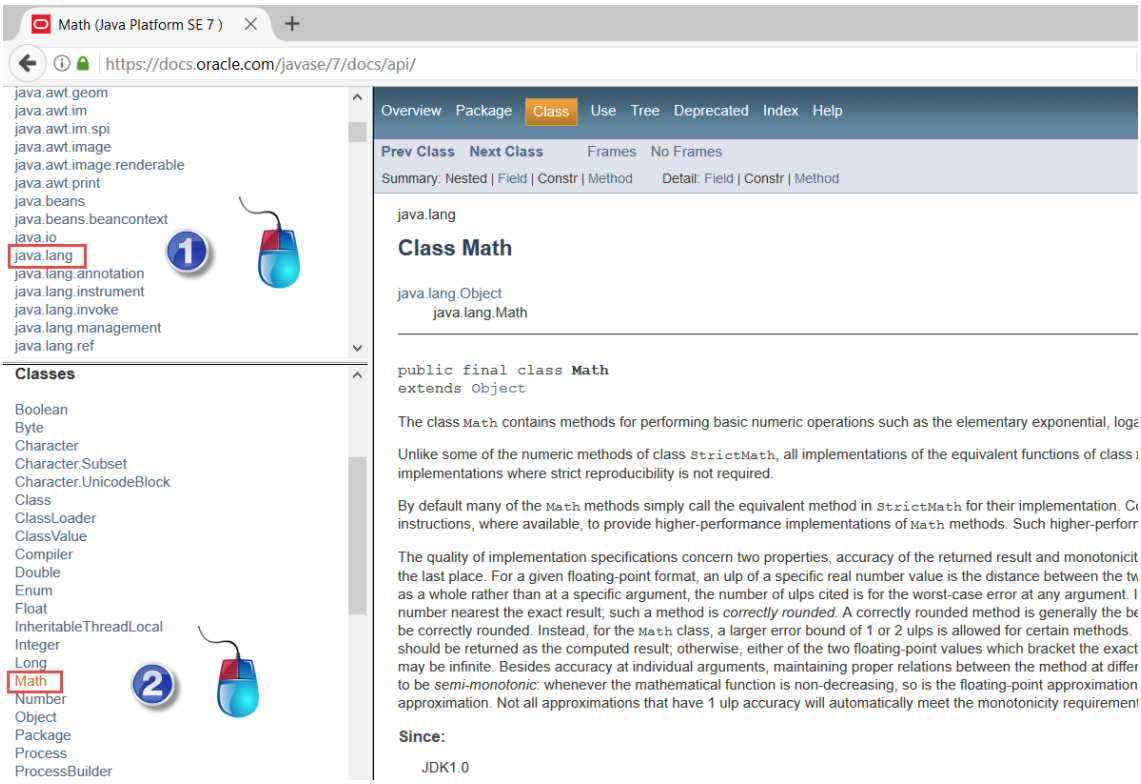
La documentación en línea de la API nos permite buscar todas las clases y métodos de la biblioteca estándar. Está en **formato HTML.**

Para ver la documentación en línea hay que dirigirse la siguiente URL: https://docs.oracle.com/javase/7/docs/api/ (esta URL puede cambiar con el tiempo y con la aparición de nuevas versiones de Java).

La pantalla se organiza en tres zonas o marcos. **El marco de la parte superior izquierda nos muestra todos los paquetes disponibles**. Por debajo en un marco un poco más grande, aparece un listado con todas las clases. Al hacer clic en cualquiera de esas clases, se nos muestra en el marco principal de la derecha toda la documentación referente a esa clase.

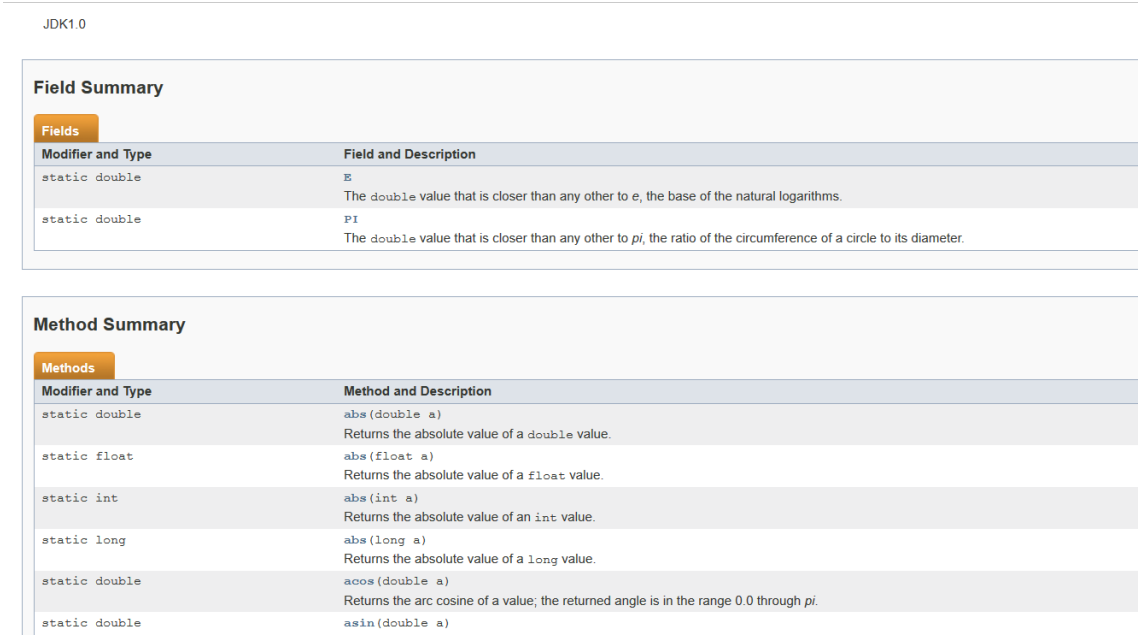


Ventana de la API de Java



Selección de paquete y clase en la API

Tal y como se describe en la imagen anterior, al seleccionar un paquete en el marco superior izquierdo **(zona paquetes)**, la API nos muestra todas las clases pertenecientes al paquete seleccionado en el marco inferior izquierdo **(zona clases).** Seleccionando una clase en concreto, la API nos abre la clase en el marco principal mostrando una descripción de la misma, los campos pertenecientes a la clase, sus constructores y sus métodos.



Parte de la descripción de la clase Math en el marco principal de la API

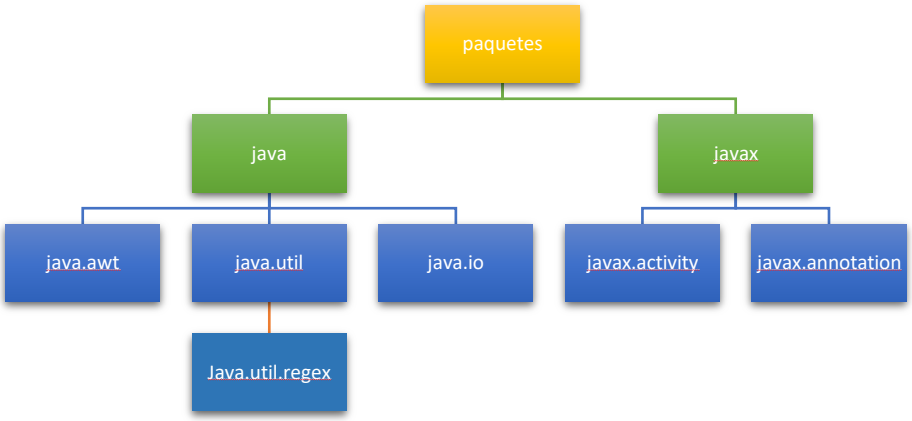
Paquetes

Al igual que ocurre con las clases en Java, donde tenemos clases predefinidas (vienen con el lenguaje y se encuentran agrupadas y ordenadas en la API) y clases propias (las que creamos como programadores), si hablamos de paquetes nos encontramos con paquetes predefinidos y paquetes propios. Los paquetes predefinidos son los paquetes propios del lenguaje y podemos verlos ordenados alfabéticamente en la API en la zona de paquetes.

Los paquetes propios son aquellos que creamos nosotros como programadores con el objetivo de ordenar nuestras clases.

- Paquetes predefinidos de la API:

Se encuentran agrupados en dos grandes categorías tal y como aparece en el siguiente esquema:



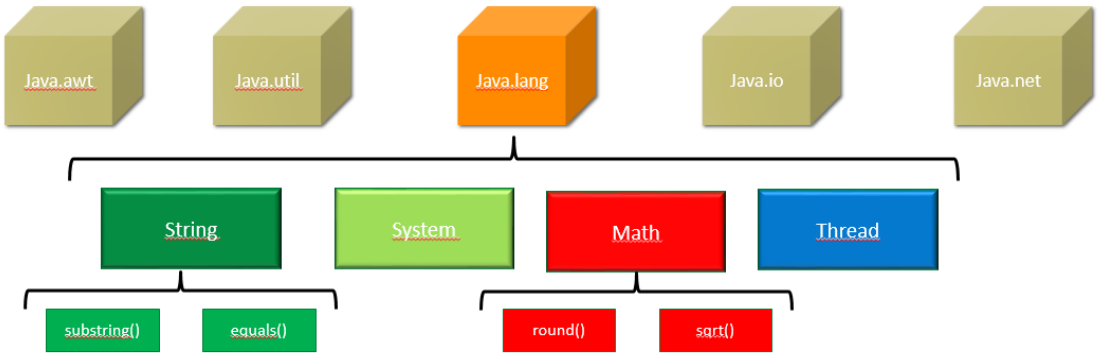
Como se observa en el esquema anterior, hay dos grandes categorías de paquetes en la API: el **paquete java** y el **paquete javax**. De cada una de estas categorías cuelgan el resto de paquetes.

Algo que ayuda a entender mejor el concepto de "paquete" es imaginarnos a los mismos como si fueran carpetas de nuestro Sistema Operativo cuya función principal será tener organizadas las clases. Y al igual que ocurre con las carpetas, dentro de un paquete puede haber otro paquete o sub-paquete.

Las clases que se encuentran dentro de un mismo paquete, son clases cuya función será similar. Por ejemplo, dentro del paquete de la API **java.awt** nos encontramos con todas las clases que sirven para crear interfaces de usuario (ventanas, botones, colores, gráficos etc).

En esta estructura de paquetes en la API, hay un paquete que se considera **paquete principal** o paquete por defecto: **el paquete java.lang**. A la hora de programar, si utilizamos clases pertenecientes a este paquete no tenemos que hacer ninguna operación adicional en nuestros programas. Pero si utilizamos cualquier clase que no pertenezca al paquete java.lang, estamos obligados a importar esa clase que queremos utilizar. De lo contrario la máquina virtual de java no reconocerá la clase que estás intentando utilizar y tu programa no funcionará.

¿Cómo importamos una clase que no pertenece al paquete java.lang?

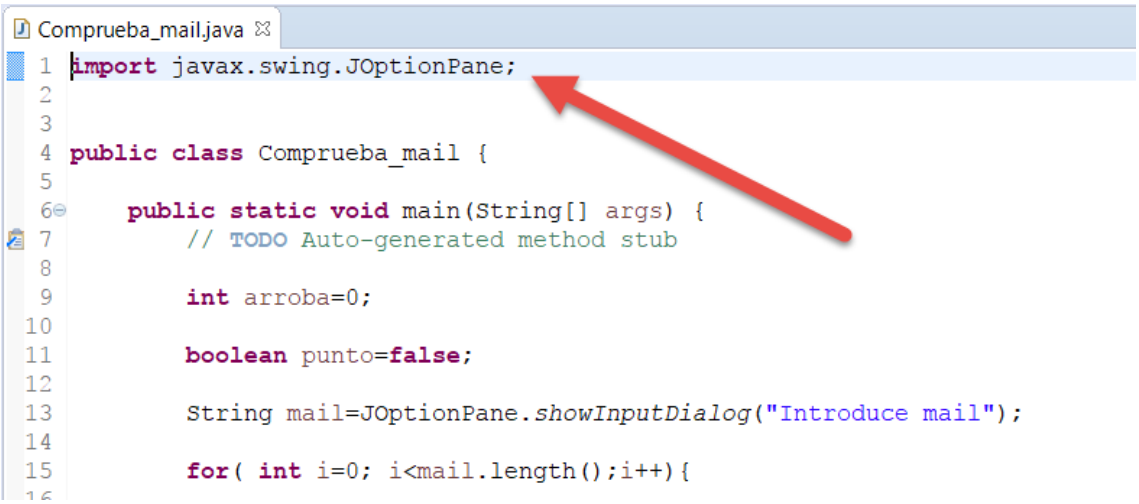


Representación de jerarquía del paquete java.lang

(con solo unas pocas clases de ejemplo)

- Importación de clases:

Para importar una clase que no pertenece al paquete por defecto (java.lang), utilizamos la palabra reservada **import** junto con la ruta de la clase a importar utilizando la nomenclatura del punto, es decir, **nombre_del_paquete.clase**. Esta instrucción debe ir siempre al comienzo del código de nuestros programas como primera instrucción en el código. En la imagen siguiente se ve un ejemplo:


Comprueba_mail.java

```
1  import javax.swing.JOptionPane;
2
3
4  public class Comprueba_mail {
5
6      public static void main(String[] args) {
7          // TODO Auto-generated method stub
8
9          int arroba=0;
10
11         boolean punto=false;
12
13         String mail=JOptionPane.showInputDialog("Introduce mail");
14
15         for( int i=0; i<mail.length();i++){
```

En el ejemplo de la imagen importamos la clase **JOptionPane** que pertenece al paquete **javax.swing** para poder utilizarla en nuestro programa.

En el caso de necesitar utilizar más de una clase de paquetes diferentes a java.lang, agregamos tantas directivas import como necesitemos tal y como se ve en el ejemplo de la siguiente imagen:

*PruebaAcciones.java

```
1  package graficos;
2
3
4  import java.awt.event.*;
5
6  import javax.swing.JFrame;
7
8  public class PruebaAcciones {
9
10     public static void main(String[] args) {
11         // TODO Auto-generated method stub
12
13         MarcoAccion marco=new MarcoAccion();
14
15         marco.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
16
17         marco.setVisible(true);
```

Es habitual a la hora de crear aplicaciones complejas el necesitar varias clases de un mismo paquete. En estos casos en vez de importar una a una todas las clases, es más cómodo importar todo el paquete utilizando la instrucción **import nombre_paquete.nombre.*;**

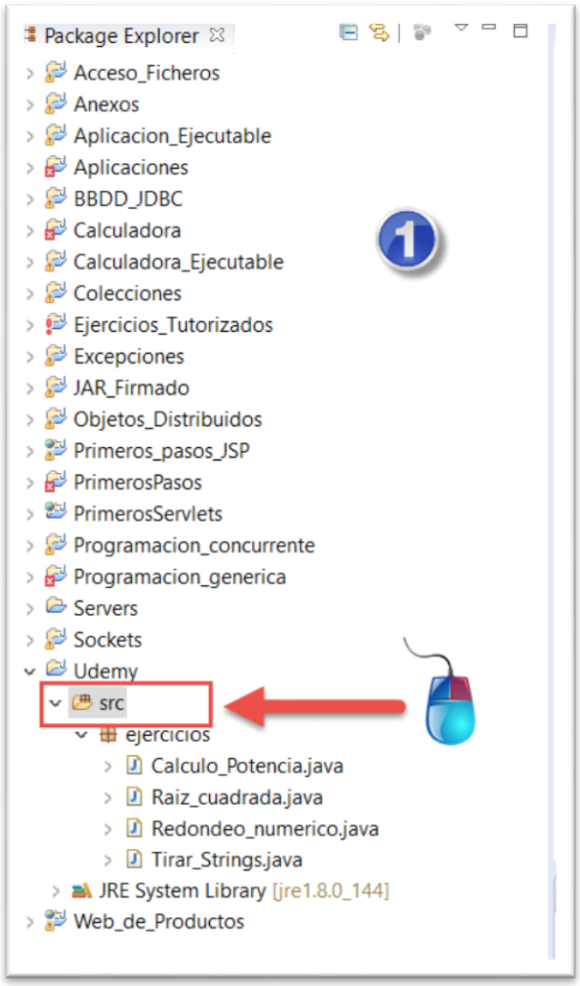
El * le indica al intérprete java que importamos todas las clases pertenecientes al paquete. De esta forma a pesar de consumir algo más de recursos (se necesita más espacio en memoria para almacenar el paquete importado), es mucho más cómodo en caso de utilizar varias clases. En la imagen siguiente se ve un ejemplo:

PruebaArea.java

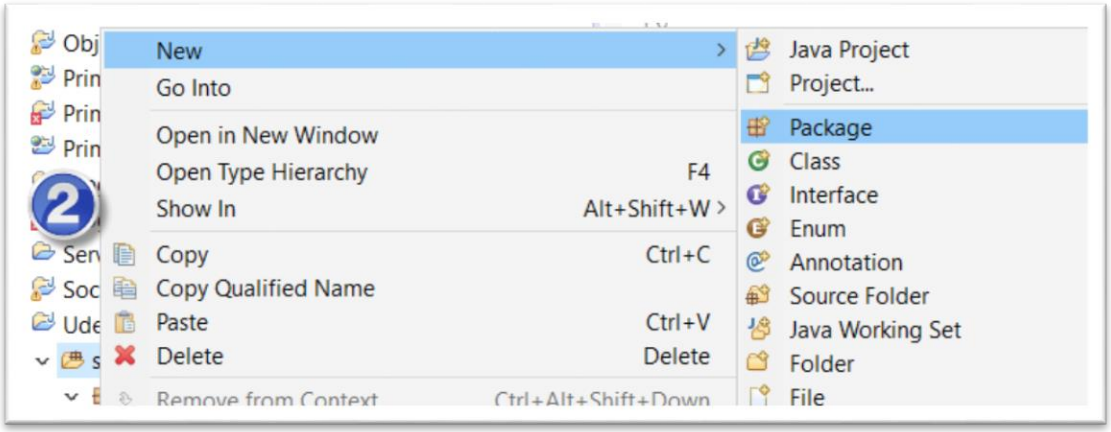
```
1  package graficos;
2
3  import java.awt.*;
4  import java.awt.event.*;
5
6  import javax.swing.*;
7
8  public class PruebaArea {
9
10     public static void main(String[] args) {
11         // TODO Auto-generated method stub
12
13         MarcoPruebaArea mimarco=new MarcoPruebaArea();
14
15         mimarco.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
16
17         mimarco.setVisible(true);
18
19     }
20
21 }
22
23
24 class MarcoPruebaArea extends JFrame{
25
```

En la imagen se aprecia cómo se han importado tres paquetes enteros: **java.awt, java.awt.event y javax.swing**.

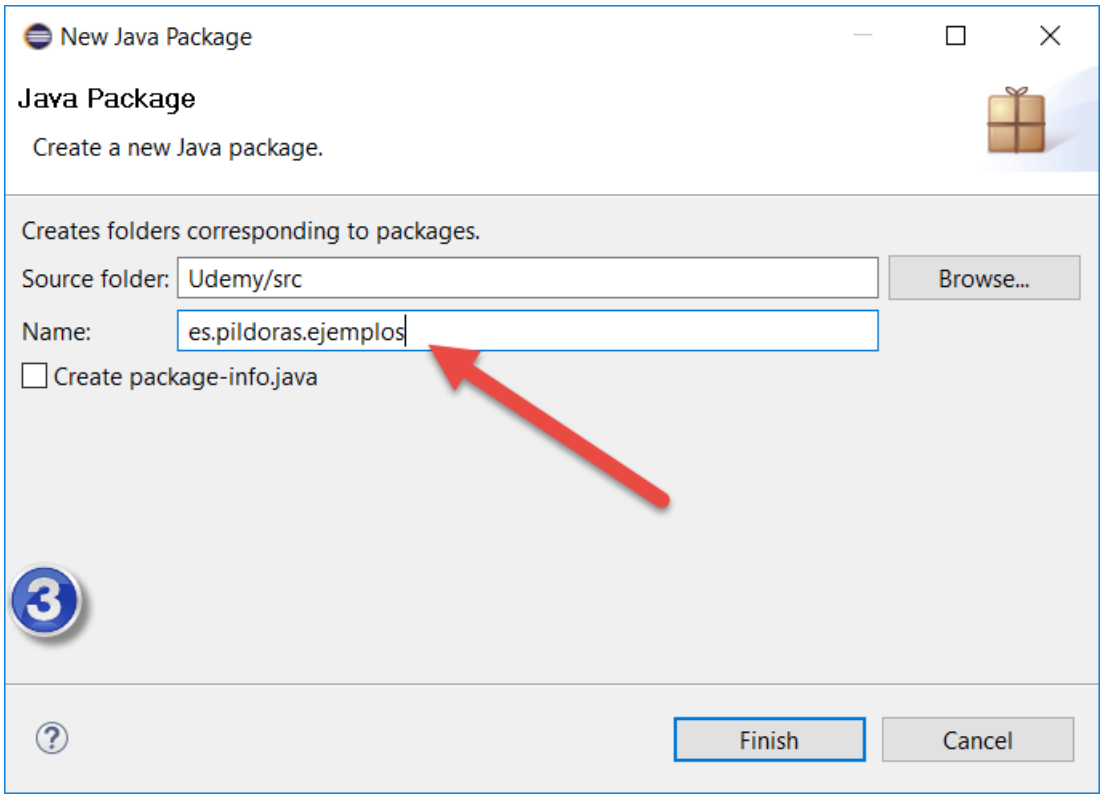
Creación de paquetes propios (con Eclipse)



1. Hacemos clic con el botón derecho del ratón sobre la carpeta “src” del proyecto

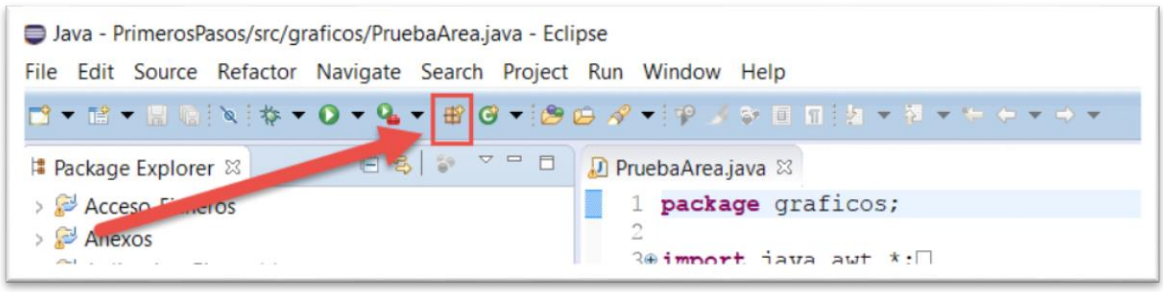


2. Escogemos la opción “**New**” – “**Package**”.

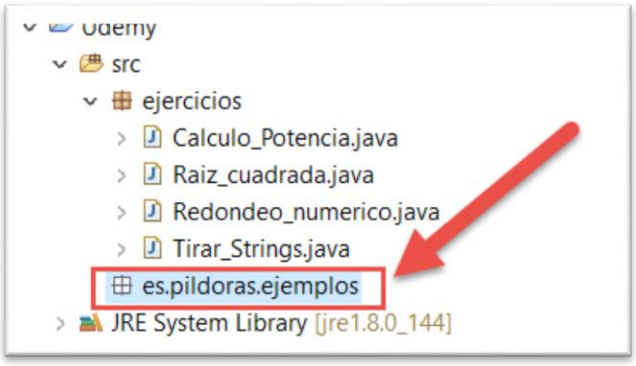


3. Especificamos un nombre para el paquete en el cuadro de texto "**Name**". Debemos escribirlo en minúsculas, sin espacios en blanco y utilizando la nomenclatura del punto. Por convención se utilizan nombres de dominio al revés para nombrar paquetes.

También podemos utilizar el botón correspondiente de la barra de herramientas de Eclipse tal y como se muestra en la imagen siguiente:



Una vez creado el paquete, lo veremos dentro de la carpeta "**src**" de nuestro proyecto tal y como refleja la imagen a continuación:



Una vez creado el paquete tan solo hay que seleccionarlo previamente con el ratón para crear clases en su interior.