Laboratorio S6
JUnit 5.0

Objetivos:

- Entender la necesidad de verificar el software
- Sistematizar pruebas unitarias en Java con JUnit 5

Herramientas que vamos a utilizar:

- Entorno de desarrollo Eclipse
- Librería jUnit

Entregable

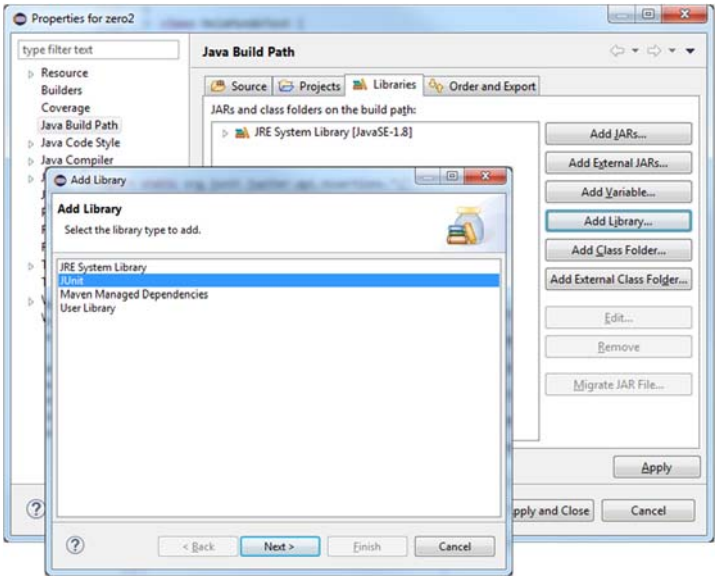
Se debe entregar un fichero **ZIP** con los **Proyectos Exportados** del laboratorio S6.

- Nombre del fichero: *apellido_nombre*.zip.
- La **entrega** es **individual** y se subirá a eGela
- **Fecha límite de entrega**: Viernes 5 de marzo a las 23:55

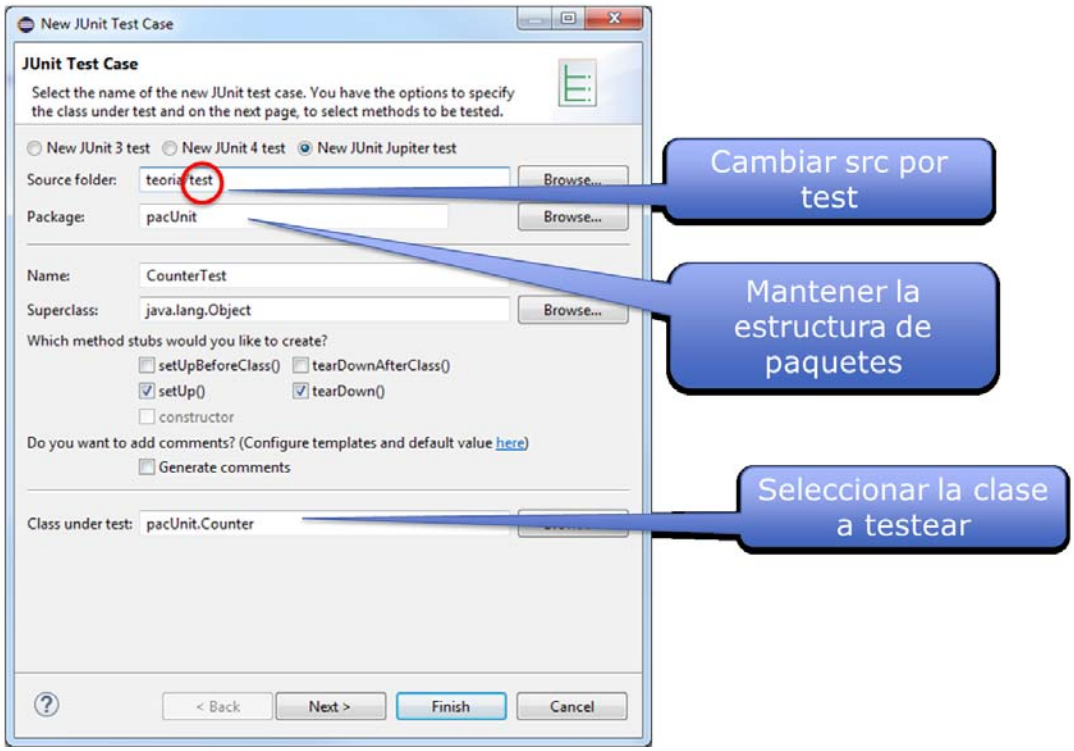
Tareas a realizar

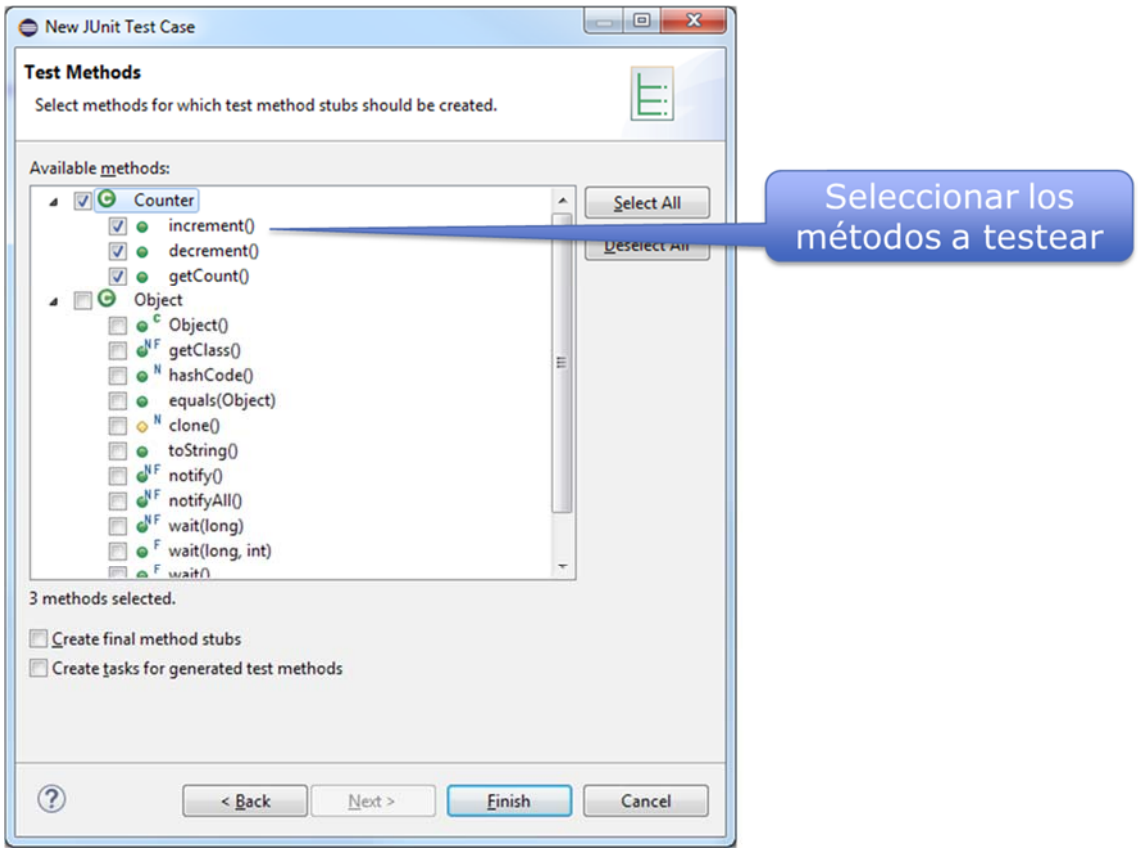
Primera parte

1) Importa el proyecto PROYCounter que tienes en eGela. Este proyecto tiene un paquete *packUnit* donde está la clase Counter que se va a verificar con JUnit.
2) Crea una carpeta *test* (*File* → *New* → *Source Folder*) que es donde generaremos las clases de test
3) Añade la librería de JUnit al path: ir a *Project* →*Properties* y en la opción *Java Build Path* selecciona la pestaña *Libraries*. Seleccionar *AddLibrary*. Finalmente seleccionar que se quiere añadir la librería JUnit (la versión 5)



4) Para generar la clase de test selecciona la clase a verificar y ejecuta *File-→New-→JUnitTestCase* (si no aparece seleccionar *File-→New-→Other* y buscar *JUnitTestCase*). Sigue los pasos marcados a continuación





5) En la clase de test C*ounterTest* que acabas de crear copia lo siguiente:

   a) Declara el atributo counter1

```
private Counter counter1;
```

   b) En el método setUp se generará el objeto que se utilizará por todos los métodos test

```
…
@BeforeEach
void setUp() {// Generate the objects to be used in all the test methods
        counter1 = new Counter();
}
```

   c) En el método testIncrement hay que comprobar que si incrementamos una vez, el valor devuelto es 1 y la segunda vez 2

```
@Test
void testIncrement() {
        assertEquals(1, counter1.increment());
        assertEquals(2, counter1.increment());
    }
```

   d) En el método testDecrement, comprobar que si se decrementa una vez, el valor devuelto es -1

```
@Test
void testDecrement() {
        assertTrue(counter1.decrement() == -1);

}
```

6) Ejecuta los Test, para ello *Run →Run as →JUnitTest*

7) Analizar qué ocurre

8) Cambia la constructora de la clase Counter e inicializa el atributo *count* a 10. Vuelve a ejecutar y mira lo que sucede

9) Usa en el método testDecrement la versión sobrecargada de assertTrue que admite añadir un String al final. Cambia el mensaje de error que se mostrará.

10) Añade al principio de la clase y antes del método testDecrement la anotación *@DisplayName*. Vuelve a ejecutar y mira lo que sucede

Segunda parte

11) Importa el proyecto LaboS06 que está disponible en eGela. Contiene una versión parcial del proyecto de servidor de correo con el que hemos estado trabajando en los laboratorios anteriores.

12) Verifica la implementación del método equalsReceiver. Casos de prueba a considerar:

    a) Cuando coinciden los receptores. Implementa este caso con dos mensajes que tengan el mismo receiver

    b) Cuando no coindicen los receptores. Implementa este caso con dos mensajes que no tengan el mismo receiver

    Para la implementación: Define una variable Email en la clase, inicialízala en el setUp. Implementa cada uno de los casos en un método de test diferente.

13) Verifica el método storeEmail. Casos que se deben probar:

- Almacenar un mensaje estando la lista de mensajes del servidor vacía
- Tener en el servidor un mensaje almacenado y almacenar otro

a) Copia el método setUp que se te proporciona a continuación

```
@BeforeEach
void setUp() {// Generate the objects to be used in all the test methods
       email1=new Email("Ainhoa", "Yeray","Weekend plan");
       email2=new Email("Yaiza", "Yeray","Meeting Schedule");
       mServer= new MailServer();
       list= new ArrayList<Email>();
}
```

b) Teniendo en cuenta la implementación del setUp, declara para la clase de test los atributos email1, email2, mServer y list.

c) Crea un método de test para cada uno de los casos.

d) Para que funcione adecuadamente debes tener sobrescrito en Email el método equals (tráelo del laboratorio anterior).

Parte complementaria

14) Verifica el correcto comportamiento del método storeEmail sin utilizar el método getEmailList en la clase MailServer.

15) Añade el método removeEmail implementado en la clase **`MailServer`** del laboratorio anterior.

16) Verifica el correcto comportamiento del método removeEmail sin utilizar el método getEmailList de la clase MailServer.