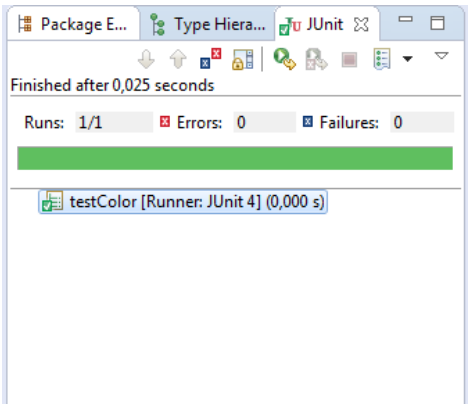
En este trabajo realizaremos diversos tests sobre el proyecto “Concesionario de Coches”. En estos tests comprobamos la existencia de los tres colores (azul, rojo y plata), las dos marcas (BMW y SEAT), todos los modelos disponibles, etc.

Test Color:

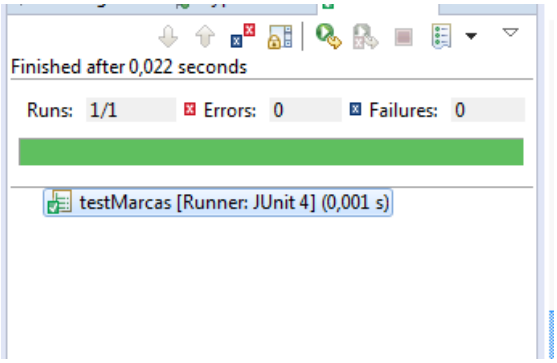



```
    //Test para comprobar que existen estos colores
    @Test
    public void testColor() {
        assertNotNull(Color.AZUL);
        assertNotNull(Color.ROJO);
        assertNotNull(Color.PLATA);
    }
    //Test para comprobar que existen estas marcas
```

De esta forma hemos podido comprobar que existen esos tres colores.

Test Marcas:

Test para comprobar que existen las marcas BMW y SEAT.

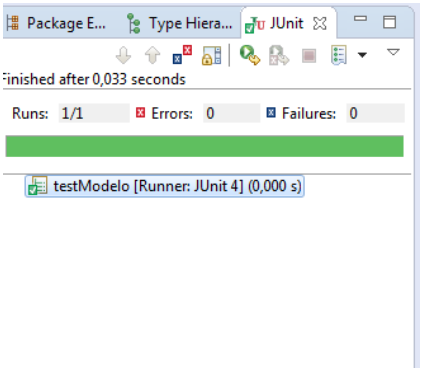



```
    //Test para comprobar que existen estas marcas
        @Test
        public void testMarcas() {
            assertNotNull(Marca.BMW);
            assertNotNull(Marca.SEAT);

        }
```

Test Modelo:

Comprobamos que los modelos sean los correctos.

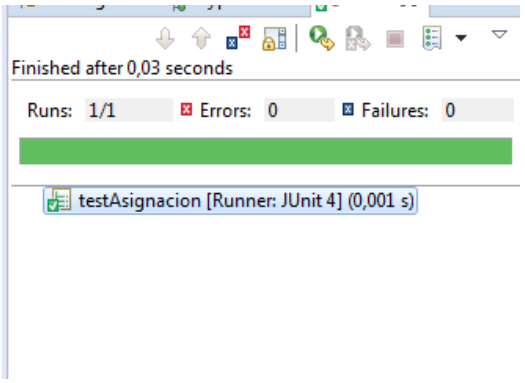



```
    //Test para comprobar que existen estos modelos
        @Test
        public void testModelo() {
            assertNotNull(Modelo.CORDOBA);
            assertNotNull(Modelo.IBIZA);
            assertNotNull(Modelo.TOLEDO);
            assertNotNull(Modelo.SERIE1);
            assertNotNull(Modelo.SERIE2);
            assertNotNull(Modelo.SERIE3);
            assertNotNull(Modelo.SERIE5);

        }
```

Test Asignación Marcas-Modelos:

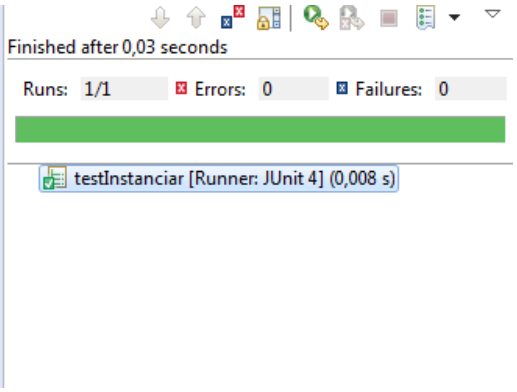
Comprobamos que las marcas tienen asociados correctamente los modelos.




```
//Asignacion de modelos con marcas
    @Test
    public void testAsignacion() {
        assertEquals(Modelo.CORDOBA.getMarca(), Marca.SEAT );
        assertEquals(Modelo.IBIZA.getMarca(), Marca.SEAT);
        assertEquals(Modelo.TOLEDO.getMarca(), Marca.SEAT);
        assertEquals(Modelo.SERIE1.getMarca(), Marca.BMW);
        assertEquals(Modelo.SERIE2.getMarca(), Marca.BMW);
        assertEquals(Modelo.SERIE3.getMarca(), Marca.BMW);
        assertEquals(Modelo.SERIE5.getMarca(), Marca.BMW);

    }
```

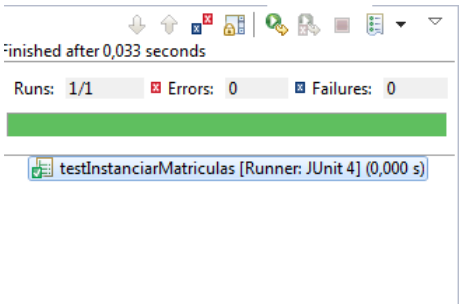
Test Instanciar:




```
//Comprobacion instanciar
    @Test
    public void testInstanciar() {
    assertNotNull(Coche.instanciarCoche("1111BBB", Color.PLATA, Modelo.TOLEDO));
    assertNotNull(Coche.instanciarCoche("1111 BBB", Color.PLATA, Modelo.TOLEDO));
    assertNotNull(Coche.instanciarCoche("1111-BBB", Color.PLATA, Modelo.TOLEDO));
    }
```

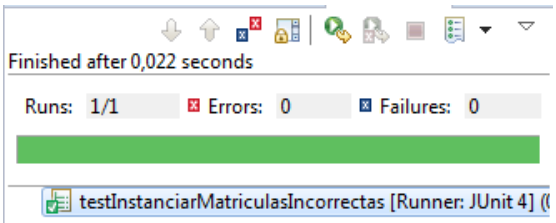
Test Instanciar Matriculas:

En este caso comprobamos si funciona correctamente el método instanciarCoche(matricula);




```
//Comprobacion instanciar matriculas correctas
    @Test
    public void testInstanciarMatriculas() {
    assertNotNull(Coche.instanciarCoche("1111BBB"));
    assertNotNull(Coche.instanciarCoche("1111 BBB"));
    assertNotNull(Coche.instanciarCoche("1111-BBB"));
    }
```

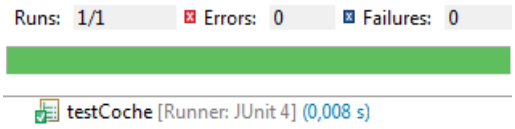
Test Instanciar Matriculas Incorrectas:




```
//Comprobacion instanciar matriculas incorrectas
    @Test
    public void testInstanciarMatriculasIncorrectas() {
    assertNull(Coche.instanciarCoche("1111BB"));
    assertNull(Coche.instanciarCoche("1111AAA"));
    assertNull(Coche.instanciarCoche("1111_BBB"));
    assertNull(Coche.instanciarCoche("11111BBB"));
    assertNull(Coche.instanciarCoche("BBB1111"));
    assertNull(Coche.instanciarCoche("111ABBB"));
    }
```

Ahora estamos comprobando si el método instancia matriculas con formato incorrectas.

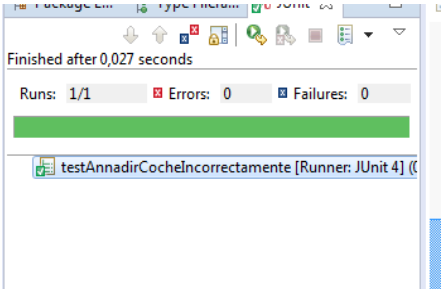
Test posibles con coche:




```
//Añadir coches correctamente
    @Test
    public void testCoche() {
    assertTrue(concesionario.annadir("1111BBB", Color.AZUL, Modelo.CORDOBA));
    assertTrue(concesionario.annadir("1111 BBB", Color.AZUL, Modelo.CORDOBA));
    assertTrue(concesionario.annadir("1111-BBB", Color.AZUL, Modelo.CORDOBA));
    //Tamaño concesionario
    assertEquals(concesionario.size(),3);
    assertNotEquals(concesionario.size(),2);
    //Eliminar coche existente
    assertTrue(concesionario.eliminar("1111BBB"));
    //Eliminar coche inexistente
    assertFalse(concesionario.eliminar("1111BBC"));
    //Añadir coche repetido
    assertFalse(concesionario.annadir("1111 BBB", Color.AZUL, Modelo.CORDOBA));
    //Mostrar coche existente
    assertNotNull(concesionario.get("1111 BBB"));
    //Mostrar coche inexistente
    assertNull(concesionario.get("2222BBB"));
    }
```

En este test, hemos comprobado si se añaden bien los coches, si se calcula bien los tamaños del arraylist, si el método eliminar, elimina correctamente coches que existen y no puede eliminar coches que no existen, añadimos un coche que ya esté y mostramos un coche existente y no existente. En todos los casos salen las pruebas satisfactoriamente.

Test añadir coche incorrectamente:

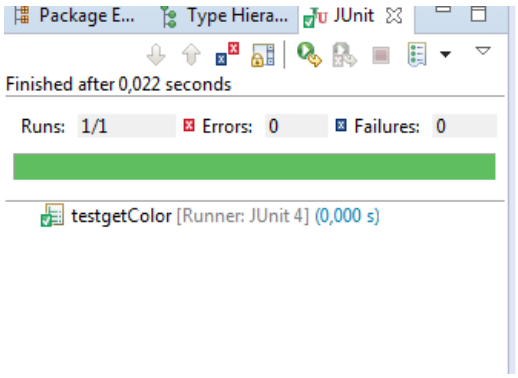



```
//Añadir coches incorrectamente
    @Test
    public void testAnnadirCocheIncorrectamente() {
    assertFalse(concesionario.annadir("1111BB", Color.AZUL, Modelo.CORDOBA));
    assertFalse(concesionario.annadir("1111AAA", Color.AZUL, Modelo.CORDOBA));
    assertFalse(concesionario.annadir("1111_BBB", Color.AZUL, Modelo.CORDOBA));
    assertFalse(concesionario.annadir("11111BBB", Color.AZUL, Modelo.CORDOBA));
    assertFalse(concesionario.annadir("BBB1111", Color.AZUL, Modelo.CORDOBA));
    assertFalse(concesionario.annadir("111ABBB", Color.AZUL, Modelo.CORDOBA));

    }
```

En esta prueba intentamos añadir coches inválidos a nuestro concesionario.

Test con getCochesColor:




```
//Obtener los colores de un coche
    @Test
    public void testgetColor() {
    assertNotNull(concesionario.getCochesColor(Color.AZUL));
    assertNotNull(concesionario.getCochesColor(Color.PLATA));
    assertNotNull(concesionario.getCochesColor(Color.ROJO));


    }
```

Por último, comprobamos la funcionalidad del método getCochesColor.