Cuidando a Pancho

Caso de Estudio: Cuidando a Pancho

Fundamentos de Ingeniería del Software 2023-24

UPM - ETSISI

Pruebas Unitarias

1.CLASE CONTROLADORMASCOTA

-La clase **controladorMascota** dispone de los métodos públicos **getMascotaRiac()**, ***altaMascota()*** y ***altaMascotaExotica()***, de los cuales los dos últimos mencionados utilizan una serie de métodos privados (***solicitarDatosMascota()***, ***registrarMascota()***, ***comprobarRiacPoliza(String riac, int poliza)***) para realizar dichas funcionalidades, por lo que los distintos casos de prueba que obtengamos vendrán dados por los distintos caminos que se obtengan en la ejecución de dichos métodos privados.

Para la creación y comprobación de las pruebas se ha tenido que crear un usuario "PruebaControladorMascota" para comprobar las distintas funcionalidades sin que salten errores.

2.CASOS DE PRUEBA ALTAMASCOTA()

-Para la realización de las pruebas de caja negra de **altaMascota()** se han seguido los siguientes casos de prueba:

- **testAltaMascotaOk():** La ejecución correcta del método que debe comprobar si se ha introducido una mascota en la lista de mascotas.
- **testAltaMascotaRiacNulo():** Corresponde a la ejecución cuando el RIAC introducido es nulo, comprueba si salta su correspondiente error.
- **testAltaMascotaRiacInvalido():** Corresponde a la ejecución cuando el RIAC introducido no es válido, comprueba si salta su correspondiente error.
- **testAltaMascotaRiacRepetido():** Corresponde a la ejecución cuando se intenta dar de alta a una mascota con un RIAC que ya se encuentra en la lista de mascotas, comprueba si salta su correspondiente error.
- **testAltaMascotaPolizaNula():** Corresponde a la ejecución cuando el número de póliza introducido es nulo, comprueba si salta su correspondiente error.
- **testAltaMascotaPolizaNegativa():** Corresponde a la ejecución cuando el número de póliza introducido es <=0, comprueba si salta su correspondiente error.
- **testAltaMascotaPolizaRepetida():** Corresponde a la ejecución cuando se intenta dar de alta a una mascota con un número de póliza que ya se encuentra en la lista de mascotas, comprueba si salta su correspondiente error.

3.CASOS DE PRUEBA ALTAMASCOTAEXOTICA()

-Para la realización de las pruebas de caja negra de **altaMascotaExótica()** se han seguido los mismos casos de prueba que en **altaMascota()**, debido a que la única diferencia entre un método y otro se encuentra en que **altaMascota()** genera una mascota normal y **altaMascotaExotica()** genera una mascota exótica.

4.CASOS DE PRUEBA GETMASCOTARIAC()

-Para la realización de las pruebas de caja negra de getMascotaRiac() se han seguido los distintos casos de prueba:

- **testGetMascotaRiacOk():** Ejecución del método en la que devuelve la mascota correspondiente al RIAC en la lista de mascotas.
- **testGetMascotaRiacMal():** Ejecución del método en donde no devuelve ninguna mascota que se corresponda con el RIAC en la lista de mascotas.

```java
@Test
public void testAltaMascotaOk() throws IOException {
    String input = "1234567890,12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascota();

    assertEquals( expected: 1, listaMascotas.size());
}

Adrián Yepes
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaRiacNulo() throws IOException {
    String input = ",12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascota();
}

Adrián Yepes
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaRiacInvalido() throws IOException {
    String input = "12345,12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascota();
}
```

```java
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaRiacRepetido() throws IOException {
    Mascota mascota = new Mascota( riac: "1234567890",  poliza: 12345,  nombre: "Firulais",  localidad: "Madrid",  descripcion: "Gato", dueno);
    listaMascotas.add(mascota);

    String input = "1234567890,12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascota();
}

Adrián Yepes
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaPolizaNula() throws IOException {
    String input = "1234567890,,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascota();
}

Adrián Yepes
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaPolizaNegativa() throws IOException {
    String input = "1234567890,-5,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascota();
}
```

```java
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaPolizaRepetida() throws IOException {
    Mascota mascota = new Mascota( riac: "1234567890",  poliza: 12345,  nombre: "Firulais",  localidad: "Madrid",  descripcion: "Gato", dueno);
    listaMascotas.add(mascota);

    String input = "1234567890,12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascota();
}
```

```java
@Test
public void testAltaMascotaExoticaOk() throws IOException {
    String input = "1234567890,12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascotaExotica();

    assertEquals( expected: 1, listaMascotas.size());
}

Adrián Yepes
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaExoticaRiacNulo() throws IOException {
    String input = ",12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascotaExotica();
}

Adrián Yepes
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaExoticaRiacInvalido() throws IOException {
    String input = "12345,12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascotaExotica();
}
```

```java
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaExoticaRiacRepetido() throws IOException {
    Mascota mascota = new Mascota( riac: "1234567890",  poliza: 12345,  nombre: "Firulais",  localidad: "Madrid",  descripcion: "Gato", dueno);
    listaMascotas.add(mascota);

    String input = "1234567890,12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascotaExotica();
}

Adrián Yepes
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaExoticaPolizaNula() throws IOException {
    String input = "1234567890,,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascotaExotica();
}

Adrián Yepes
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaExoticaPolizaNegativa() throws IOException {
    String input = "1234567890,-5,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascotaExotica();
}
```

```java
@Test(expected = IllegalArgumentException.class)
public void testAltaMascotaExoticaPolizaRepetida() throws IOException {
    Mascota mascota = new Mascota( riac: "1234567890",  poliza: 12345,  nombre: "Firulais",  localidad: "Madrid",  descripcion: "Gato", dueno);
    listaMascotas.add(mascota);

    String input = "1234567890,12345,Firulais,Madrid,Gato\n";
    System.setIn(new ByteArrayInputStream(input.getBytes()));

    iSistema.setScanner(new Scanner(System.in));
    controladorMascota.altaMascotaExotica();
}
```

```java
new *
@Test
public void testGetMascotaRiacOk() {
    Mascota mascotaRequerida = new Mascota( riac: "1234567890",  poliza: 12345,  nombre: "Firulais",  localidad: "Madrid",  descripcion: "Gato", dueno);
    listaMascotas.add(mascotaRequerida);

    Mascota mascotaReal = controladorMascota.getMascotaRiac("1234567890");

    assertEquals(mascotaReal, mascotaRequerida);
}

new *
@Test
public void testGetMascotaRiacMal() {
    Mascota mascotaReal = controladorMascota.getMascotaRiac("1234567890");

    assertNull(mascotaReal);
}
```

5.RESULTADO TESTS