
Prácticas de Metodología de la Programación

Práctica 3: Pruebas unitarias

David Amor Antolin
Wu Li Lu

Contenidos

1. Explicación del desarrollo seguido	3
2. Repositorio e integrantes	3
3. Pruebas unitarias	4
4. Tabla de las ilustraciones	8

1. Explicación del desarrollo seguido

Instrucciones

Al haber revisado el mensaje de Antonio sobre la práctica 2 de que no compilaban algunas prácticas por culpa de las 'ñ' y las tildes en el código, hemos decidido crear un proyecto en NetBeans sustituyendo las 'ñ' por 'nn' y eliminando las tildes; ya que en un principio habíamos creado el proyecto en Eclipse y no nos daba problemas de compilación. Fallo nuestro por no comunicarlo.

Abriendo el proyecto en NetBeans no debería dar ningún problema de compilación.

2. Repositorio e integrantes

Datos

Los integrantes de esta práctica somos:

- | | | |
|----------------------|--|------------------------|
| ❖ David Amor Antolín | - d.amora@alumnos.urjc.es | - Github: Almendron100 |
| ❖ Wu Li Lu | - w.lil@alumnos.urjc.es | - Github: alilu8 |

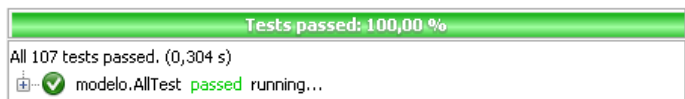
Repositorio de Github: [repositorio](#)

Aclaración: Retroalimentación de la segunda memoria, al habernos dividido hay varias carpetas en el repositorio. Hemos adjuntado en la que hemos trabajado esta tercera práctica.

3. Pruebas unitarias

Desarrollo

En esta tercera práctica se nos pedía realizar las pruebas unitarias de los métodos de las distintas clases de nuestro proyecto. Hemos conseguido realizar las pruebas de todos los métodos con satisfacción, exceptuando los métodos de tipo 'Get' y 'Set', y también aquellos métodos que solo nos devuelvan una cadena de caracteres como los toString() o los que solo tengan System.out.println().



1. Todas los test OK.

Retroalimentación

Nos hemos encontrado con algunos métodos que, o bien nos daban un error al pasar las pruebas y otros que nos pasaban la prueba bien pero no comprobaban algunos parámetros. A continuación, vamos a mostrar cuáles de ellos han sido y los cambios que han sufrido:

- Clase Comentario: las pruebas los pasaba satisfactoriamente, pero al modificar el voto no lo actualizaba bien. Tampoco añadíamos ni quien votaba ni la puntuación.

```
private HashMap<String, Usuario> votantes = new HashMap<>();
private HashMap<String, Integer> votos = new HashMap<>();
```

2. Nuevas variables de la clase Comentario.

```
public boolean votar(int puntuacion, Usuario usuario) {
    if(!this.getVotantes().containsKey(usuario.getNick())) {
        this.puntuacion += puntuacion;
        this.votantes.put(usuario.getNick(), usuario);
        this.votos.put(usuario.getNick(), puntuacion);
        return true;
    }
    else if(this.getVotos().get(usuario.getNick()) != puntuacion && this.get
        this.puntuacion = puntuacion;
        this.votos.remove(usuario.getNick());
        this.votos.put(usuario.getNick(), puntuacion);
        return true;
    }
    return false;
}
```

3. Nuevo método de votar comentario.

- Clase EntradaGenerica: igual que con la clase Comentario y el método de votar, pero esta vez la prueba daba error. La solución al problema fue la misma que le den punto anterior.



4. Error actualizar voto entrada genérica.

```
public boolean votar(int puntuacion, Usuario usuario) {
    if(this.verificada && !this.getVotantes().containsKey(usuario.getNick()))
        this.puntuacion += puntuacion;
        this.votantes.put(usuario.getNick(), usuario);
        this.votos.put(usuario.getNick(), puntuacion);
        return true;
    }
    else if(this.verificada && votos.get(usuario.getNick()) != puntuacion && vo
        this.puntuacion = puntuacion;
        this.votos.remove(usuario.getNick());
        this.votos.put(usuario.getNick(), puntuacion);
        return true;
    }
    return false;
}
```

5. Solución método de votar.

```
testVotar2 passed (0,0 s)
testVotar3 passed (0,0 s)
```

6. TestVotar3 arreglado.

- Clase SubForo: todas las pruebas pasaron satisfactoriamente, pero encontramos un fallo y es que no comprobábamos si un usuario existía tanto al añadirlo como al eliminarlo del subforo. Añadimos dicha comprobación a los métodos para no tener errores.

```
public void añadirSubscritor(Usuario usuario) {
    if(!this.suscritos.containsKey(usuario.getNick())){
        this.suscritos.put(usuario.getNick(), usuario);
        System.out.println("Te has suscrito al subforo de '" + this.getTitulo() + '
    }
    else
        System.out.println("Ya estas suscrito a en este foro");
}
```

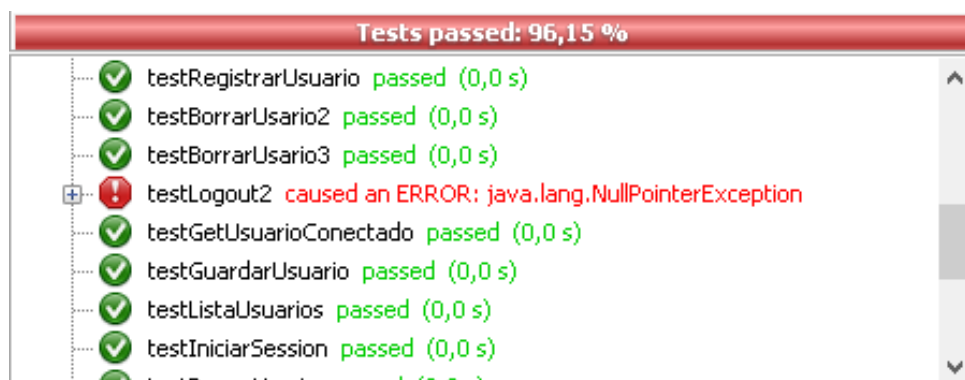
7. Nuevo método añadir suscriptor.

```
public void eliminarSubscritor(Usuario usuario) {
    if(this.suscritos.containsKey(usuario.getNick())){
        this.suscritos.remove(usuario.getNick());
        System.out.println("Te has desuscrito del subforo de '" + this.getTitulo() + '
    }
    else
        System.out.println("No estabas suscrito en este foro.");
}
```

8. Nuevo método eliminar suscriptor.

- Clase controlador: aquí encontramos dos errores, el primero fue al hacer login. No nos daba error en la prueba, pero lo que tenía que imprimir el sistema no era lo correcto, y es que faltaban dos comprobaciones en el método de iniciar sesión.

El segundo error fue al hacer el logout(), la prueba nos daba error de NullPointerException, y es que al intentar hacer login de un usuario que no se encuentra conectado, la variable sesión tenía como valor por defecto 'null'. Solo tuvimos que añadir una comprobación y lo solucionamos.




9. Error en la prueba de Logout2.

```

public boolean Logout() {
    if(this.getUsuarioConectado() != null) {
        System.out.println("Sesión de '" + sesion.getNick() + "' cerrada");
        sesion = null;
        return false;
    }
    System.out.println("No puedes cerrar una sesión sin haberte logeado");
    return false;
}

```

10. Nuevo método de cerrar sesión.

 testLogout2 passed (0,0 s)

11. Test logOut2 OK.

```

public boolean iniciarSession(Usuario usuario){
    Estudiante usuarioaux = new Estudiante();
    try{
        if(usuarios.containsKey(usuario.getNick()) && (usuarios.get(usuario.getNick()) instanceof Estudiante)) {
            if(usuario.getRol().equals("estudiante")) {
                usuarioaux = (Estudiante) usuario;
                if(usuarioaux.getPenalizacion() != null)
                    if(usuarioaux.getPenalizacion().isActiva())
                        System.out.println("No puedes iniciar sesión porque estás penalizado por " + usuarioaux.getPenalizacion().getDescripcion());
                return false;
            }
        }
        System.out.println("Acabas de iniciar sesión como: " + usuario.getNick());
        sesion = usuarios.get(usuario.getNick());
        if(sesion.getNotificaciones().size() > 0) {
            System.out.println(this.mostrarNotificaciones());
        }
        this.usuarios.get(usuario.getNick()).setNotificaciones(new ArrayList());
        return true;
    }
    else if(usuarios.containsKey(usuario.getNick()) && !(usuarios.get(usuario.getNick()) instanceof Estudiante)) {
        System.err.println("Contraseña incorrecta");
        return false;
    }
    else if(!usuarios.containsKey(usuario.getNick())) {
        System.err.println("El usuario no se encuentra registrado");
        return false;
    }
    else {
        System.err.println("Error del sistema.");
        return false;
    }
} catch (Exception e) {
    System.err.println("Error en el servidor " + e.getLocalizedMessage());
    return false;
}
}

```

12. Nuevo método de iniciar sesión.

4. Tabla de las ilustraciones

1. Todos los test OK.....	4
2. Nuevas variables de la clase Comentario.....	4
3. Nuevo método de votar comentario.....	4
4. Error actualizar voto entrada genérica.....	5
5. Solución método de votar.....	5
6. TestVotar3 arreglado.....	5
7. Nuevo método añadir suscriptor.....	6
8. Nuevo método eliminar suscriptor.....	6
9. Error en la prueba de Logout2.....	6
10. Nuevo método de cerrar sesión.....	7
11. Test logOut2 OK.....	7
12. Nuevo método de iniciar sesión.....	7