Pt6 - Optimització de codi

Fet per: Alex Caballé Arasa
Professor: Daniel Jesús Garcia Martínez
Cicle: DAM 1A
Mòdul: ENTORNS DE DESENVOLUPAMENT
Data: 6 febrer 2024

Index

a. L’execució dels tests abans del refactoring

a. L’execució dels tests abans del refactoring (s’haurien de veure els noms dels mètodes de tests i el check de l’execució satisfactòria).

```
private TaulerService taulerService;

@BeforeEach
void setUp() {
    taulerService = new TaulerService();
}

@Test
void testComprovarGuanyadorX() {
    taulerService.tractarClicCasella(fila:0, columna:0);
    taulerService.tractarClicCasella(fila:0, columna:1);
    taulerService.tractarClicCasella(fila:0, columna:2);
    taulerService.tractarClicCasella(fila:1, columna:1);
    taulerService.tractarClicCasella(fila:1, columna:0);
    taulerService.tractarClicCasella(fila:1, columna:2);
    taulerService.tractarClicCasella(fila:2, columna:0);

    assertTrue(taulerService.isGameOver());
    assertEquals(expected:"X", taulerService.getGuanyador());
}

@Test
void testComprovarGuanyadorO() {
    taulerService.tractarClicCasella(fila:0, columna:0);
    taulerService.tractarClicCasella(fila:1, columna:0);
    taulerService.tractarClicCasella(fila:2, columna:0);
    taulerService.tractarClicCasella(fila:1, columna:1);
    taulerService.tractarClicCasella(fila:0, columna:1);
    taulerService.tractarClicCasella(fila:2, columna:1);
    taulerService.tractarClicCasella(fila:0, columna:2);

    assertTrue(taulerService.isGameOver());
    assertEquals(expected:"O", taulerService.getGuanyador());
}
```

b. Exemples del “refactoring” aplicat explicant què corregiu.

a. Els mètodes no tinguin més d’una estructura for (seqüencials o imbricats).
He separat la verificació del guanyador i la comprovació de si el tauler està ple en dos mètodes diferents

```java
private String verificarGuanyador() {
    for (int i = 0; i < MIDA_TAULER; i++) {
        if (caselles[i][0].equals(caselles[i][1]) && caselles[i][1].equals(caselles[i][2]) && !caselles[i][0].isEmpty()) {
            return caselles[i][0];
        }
        if (caselles[0][i].equals(caselles[1][i]) && caselles[1][i].equals(caselles[2][i]) && !caselles[0][i].isEmpty()) {
            return caselles[0][i];
        }
    }

    if (caselles[0][0].equals(caselles[1][1]) && caselles[1][1].equals(caselles[2][2]) && !caselles[0][0].isEmpty()) {
        return caselles[0][0];
    }
    if (caselles[0][2].equals(caselles[1][1]) && caselles[1][1].equals(caselles[2][0]) && !caselles[0][2].isEmpty()) {
        return caselles[0][2];
    }

    return "";
}
```

```java
private boolean esTaulerPle() {
    for (int i = 0; i < MIDA_TAULER; i++) {
        for (int j = 0; j < MIDA_TAULER; j++) {
            if (caselles[i][j].isEmpty()) {
                return false;
            }
        }
    }
    return true;
}
```

b. Els mètodes no tinguin condicions de diferents nivells.

c. Que el mètode tingui una responsabilitat i no faci “moltes” coses.

Abans: El mètode `tractarClicCasella` feia diverses tasques

Després: He separat en mètodes més petits: actualitzarCasella, verificarGuanyador i esTaulerPle

Aixo tambe fa que que els mètodes no siguin massa llargs (d)

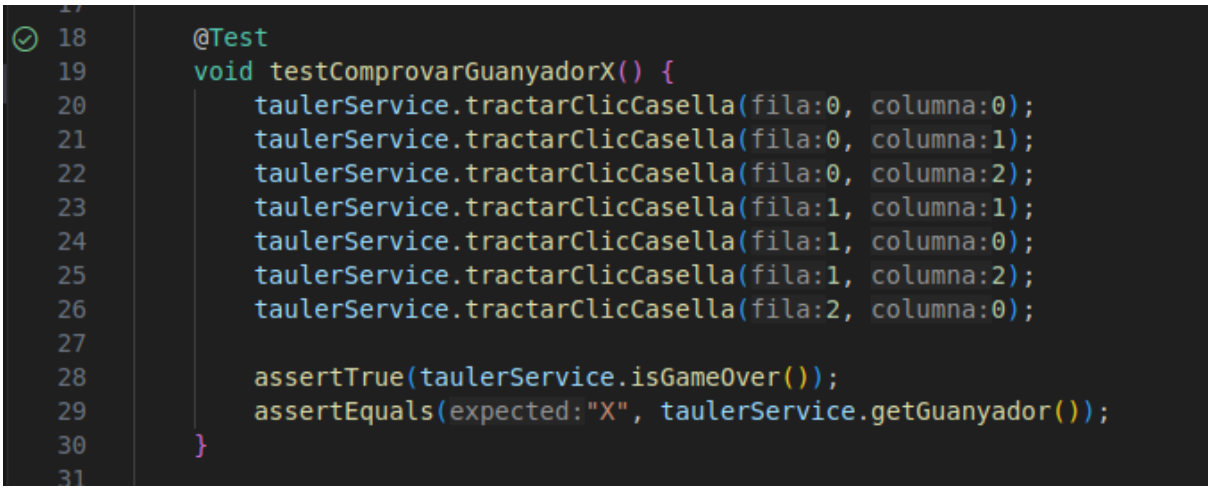
e. Que no hi hagi números màgics o valors literals a l’interior dels mètodes.
E creat el valor mida tauler per a millor claredat i futurs canvis

```java
private void inicialitzarTauler() {
    for (int i = 0; i < MIDA_TAULER; i++) {
        for (int j = 0; j < MIDA_TAULER; j++) {
            caselles[i][j] = "";
        }
    }
}
```

f. Que els noms dels mètodes, paràmetres i variables siguin descriptius.
Alguns noms com fila1 o columna1 els he canviat

c. L'execució dels tests després del refactoring.

Segueix funcionant



```java
@Test
void testComprovarGuanyadorX() {
    taulerService.tractarClicCasella(fila:0, columna:0);
    taulerService.tractarClicCasella(fila:0, columna:1);
    taulerService.tractarClicCasella(fila:0, columna:2);
    taulerService.tractarClicCasella(fila:1, columna:1);
    taulerService.tractarClicCasella(fila:1, columna:0);
    taulerService.tractarClicCasella(fila:1, columna:2);
    taulerService.tractarClicCasella(fila:2, columna:0);

    assertTrue(taulerService.isGameOver());
    assertEquals(expected:"X", taulerService.getGuanyador());
}
```

d. El detall del resultat de l'informe PMD després del "refactoring"

(en format xml és suficient, però podeu generar el site i mostrar l'HTML).

e. Una captura de l'execució del workflow en Github Actions.

f. L'enllaç

Enllaç: https://github.com/acaballee/Pt6---Optimitzaci-de-codi/settings/access?

al repositori on es trobarà el codi desenvolupat (tests, refactoring, configuració PMD i Github Actions)