

Détail :

Le projet est développé en java (compatible avec JAVA 7 et plus), avec Maven et JUnit 4 pour les tests

Aussi nous avons ajouté les dépendances suivantes :

JUnitParams (1.1.0)

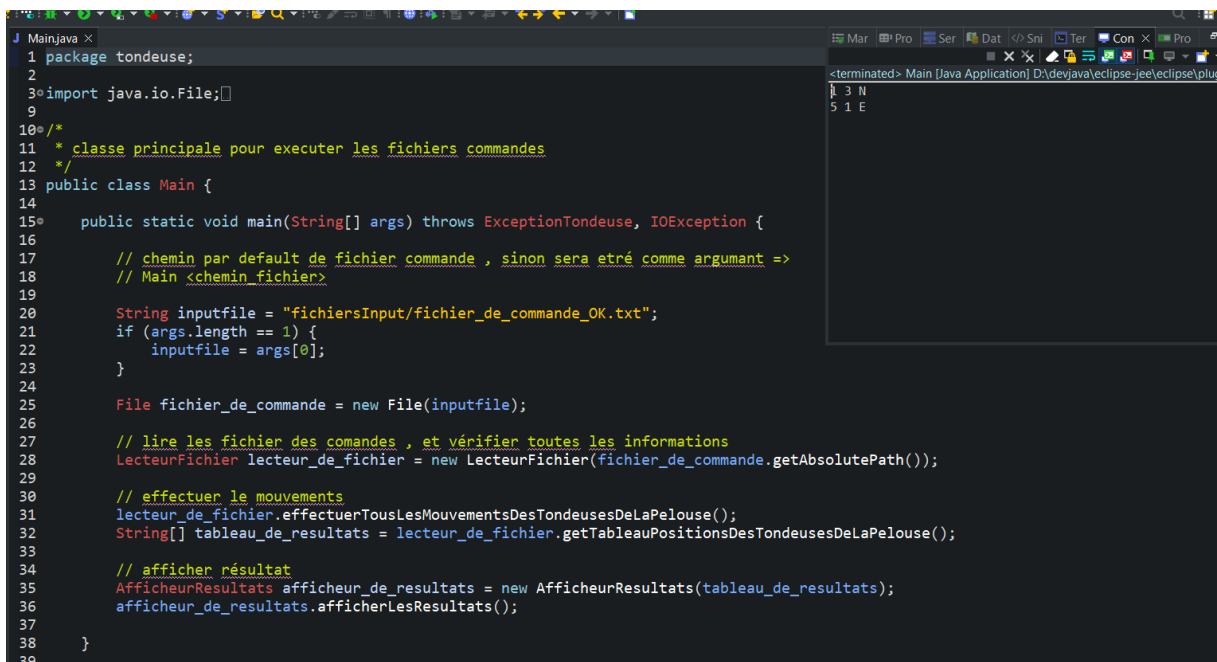
fest-assert (1.4)

Exécution de la classe Main avec le bon fichier :



```
<terminated> Main [Java Application] D:\devjava\eclipse-jee\eclipseplu
1 3 N
5 1 E
|
```

La classe Main : (Main.java)



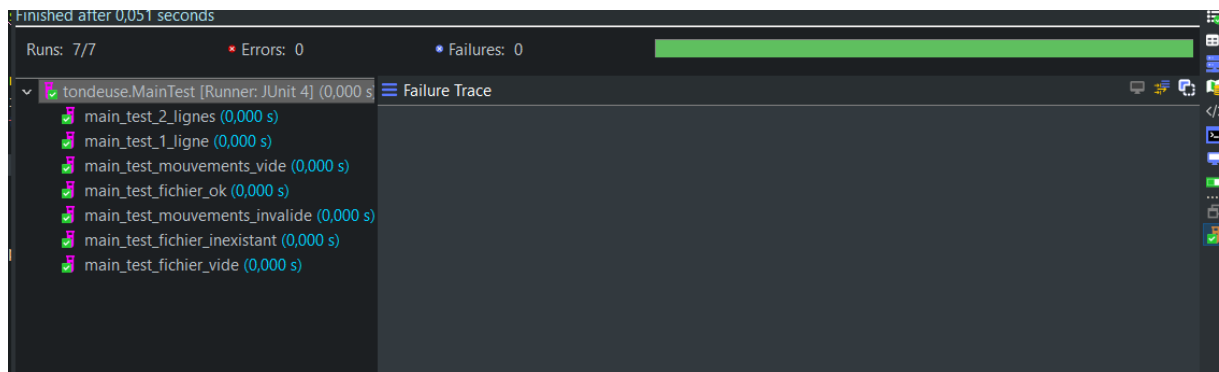
```
1 package tondeuse;
2
3 import java.io.File;
4
5
6
7
8
9
10 /*
11  * classe principale pour executer les fichiers commandes
12  */
13 public class Main {
14
15     public static void main(String[] args) throws ExceptionTondeuse, IOException {
16
17         // chemin par default de fichier commande , sinon sera etré comme argument =>
18         // Main <chemin fichier>
19
20         String inputfile = "fichiersInput/fichier_de_commande_OK.txt";
21         if (args.length == 1) {
22             inputfile = args[0];
23         }
24
25         File fichier_de_commande = new File(inputfile);
26
27         // lire les fichier des commandes , et vérifier toutes les informations
28         LecteurFichier lecteur_de_fichier = new LecteurFichier(fichier_de_commande.getAbsolutePath());
29
30         // effectuer le mouvements
31         lecteur_de_fichier.effectuerTousLesMouvementsDesTondeusesDeLaPelouse();
32         String[] tableau_de_resultats = lecteur_de_fichier.getTableauPositionsDesTondeusesDeLaPelouse();
33
34         // afficher résultat
35         AfficheurResultats afficheur_de_resultats = new AfficheurResultats(tableau_de_resultats);
36         afficheur_de_resultats.afficherLesResultats();
37
38     }
39 }
```

Les tests de la classe Main sont développés dans MainTest :

- Test d'un fichier inexistant
- Test d'un fichier avec 1 line uniquement
- Test d'un fichier avec 2 lignes
- Test d'un fichier avec des données incorrectes
- Test le bon fichier demandé

```
J Main.java J MainTest.java X
20 * test de la classe principale et les cas possibles
21 */
22 public class MainTest {
23     final String CHEMIN_FICHIER = "fichiersInput/";
24     private final static PrintStream sortie_par_defaut_vers_la_console = System.out;
25     private final static ByteArrayOutputStream output = new ByteArrayOutputStream();
26     @Rule
27     public ExpectedException expectedEx = ExpectedException.none();
28
29     @BeforeClass
30     public static void setUpBeforeClass() throws Exception {
31         System.setOut(new PrintStream(output, true, "UTF-8"));
32     }
33
34     // test si le fichier innexistant
35
36     @Test
37     public void main_test_fichier_inexistant() throws ExceptionTondeuse, IOException {
38         expectedEx.expect(ExceptionTondeuse.class);
39         expectedEx.expectMessage(Valeurs.ERREUR_FICHIER_INEXISTANT);
40         String[] arg = { "fichierinexistant" };
41         Main.main(arg);
42     }
43
44     // tester si le fichier est vide
45
46     @Test
47     public void main_test_fichier_vide() throws ExceptionTondeuse, IOException {
48         expectedEx.expect(ExceptionTondeuse.class);
49         expectedEx.expectMessage(Valeurs.ERREUR_FICHIER_VIDE);
50
51         String[] arg = { CHEMIN_FICHIER + "fichier_de_commande_vide.txt" };
52         Main.main(arg);
53     }
54
55     // tester si la line 1 est incorrecte (taille de pelouse)
56
57     @Test
58     public void main_test_1_ligne() throws ExceptionTondeuse, IOException {
59         expectedEx.expect(ExceptionTondeuse.class);
60         expectedEx.expectMessage(Valeurs.ERREUR_PELOUSE);
61
62         String[] arg = { CHEMIN_FICHIER + "fichier_de_commande_line1NOK.txt" };
63     }
64 }
```

Résultat test : toutes les tests sont bien passés



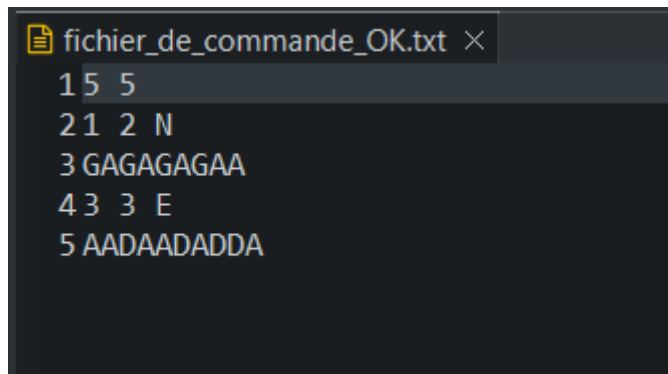
Exemple de test (fichier demandé)

```
// fichier test demande dans l'exercice ,
/*
 * 5 5 1 2 N GAGAGAGAA 3 3 E AADAADADDA => tondeuse1: 1 3 N , tondeuse2: 5 1 E
 */

@Test
public void main_test_fichier_ok() throws ExceptionTondeuse, IOException {
    String[] arg = { CHEMIN_FICHIER + "fichier_de_commande_OK.txt" };
    Main.main(arg);

    String string_de_resultats = "1 3 N" + System.lineSeparator() + "5 1 E" + System.lineSeparator();
    assertEquals(string_de_resultats, output.toString());
    assertNotNull(string_de_resultats);
    assertTrue(string_de_resultats.contains("1 3 N").contains("5 1 E"));
}
```

Fichier en input :



Autres tests : la plupart des classes sont testé avant d'implémenter la classe main, exemple ici test de la classe pelouse :

Test pelouse :

- Test de création de pelouse
- Test de crée une tondeuse dans pelouse
- Test de non création dans le cas où les positions départs dépassent la taille max

```
@Rule
public ExpectedException expectedEx = ExpectedException.none();

@Test
@Parameters({"5, 6"})
public void devraitCreerUnePelouseNonCarree(int taille_est_ouest_x, int taille_nord_sud_y) {

    Pelouse pelouse = new Pelouse(taille_est_ouest_x, taille_nord_sud_y);
    TailleDePelouse TailleDePelouse = pelouse.getTailleDePelouse();
    assertEquals(taille_est_ouest_x, TailleDePelouse.getTaille_max_x());
    assertEquals(taille_nord_sud_y, TailleDePelouse.getTaille_max_y());

}

@Test
@Parameters({"1, 2, N"})
public void devraitCreerUneTondeuse(int longitude, int latitude, char orientation){

    TailleDePelouse TailleDePelouse = new TailleDePelouse(5, 5);
    Tondeuse tondeuse = new Tondeuse(longitude, latitude, orientation, TailleDePelouse, "");
    assertEquals(longitude, tondeuse.getX());
    assertEquals(latitude, tondeuse.getY());
    assertEquals(orientation, tondeuse.getOrientation());

}

@Test
@Parameters({"1, 6, N"})
public void devraitPasCreerUneTondeuse(int longitude, int latitude, char orientation) throws ExceptionTondeuse,
expectedEx.expect(ExceptionTondeuse.class);
expectedEx.expectMessage(Valeurs.ERREUR_CREATION_TONDEUSE);
Pelouse pelouse = new Pelouse(5, 5);
pelouse.ajouterTondeuse(longitude, latitude, orientation, "");
```

Résultat :

Finished after 0,059 seconds

Runs: 4/4 ✖ Errors: 0 • Failures: 0

▼ tondeuse.base.PelouseTest [Runner: JUnit 4] ≡ Failure Trace

- > devraitPasCreerUneTondeuse (0,000 s)
- > devraitCreerUnePelouseNonCarree (0,000 s)
- > devraitCreerUneTondeuseDansUnePelouse (0,000 s)
- > devraitCreerUneTondeuse (0,000 s)