MISE EN PLACE DES TESTS UNITAIRES
(avec Junit dans JBuilder)

But
Tester le code et vérifier qu'il fait ce pourquoi il a été créé dans toutes les configurations possibles, et s'assurer que tous les cas particuliers soient bien gérés.
Il existe plusieurs intérêts :
- expliquer les différents cas limites de chacune des méthodes développées
- avoir des tests de validité sous la main
- faciliter la modification du code en assurant automatiquement du bon fonctionnement

Pour cela, les tests se doivent d'être exhaustifs.

Mode d'emploi (sous JBuilder)
En parallèle du développement d'un code, on met en place des classes qui seront utilisées uniquement pour vérifier que le code fonctionne comme prévu.
Après avoir créé votre classe et ses méthodes : (le fichier de la classe concernée doit être ouvert)
- Menu File | New
- Onglet Test | Test Case
- Sélectionnez toutes les méthodes de la classe que vous voulez mettre en test
- Choisissez le package associé ; donnez un nom à cette classe de test. Utilisez si possible les valeurs proposées par défaut, c'est à dire : le même package que celui de la classe et le nom « `Test+nom+de+la+classe` »
- Vous pouvez ajouter des objets de classes existantes qui seront ajoutés au test (fixtures).

Quand vous cliquez sur « Finish » JBuilder créé alors une nouvelle classe avec le minimum requis :
- un objet de la classe à tester
- un constructeur
- protected void setUp() throws Exception : pour l'initialisation, la création des objets
- protected void tearDown() throws Exception : pour le nettoyage

Cette nouvelle classe de test est stockée dans un répertoire séparé nommé « test ».
Ensuite il vous revient de compléter avec de nouvelles méthodes de tests suivant les différentes configurations.
Les méthodes setUp et tearDown sont appelées avant et après chacune des méthodes de tests.
On peut indifféremment tester des méthodes ou des scenarii.
Les méthodes de tests ne peuvent pas prendre de paramètres en entrée.
Il peut aussi être important de vérifier qu'une méthode renvoie une exception. Dans ce cas, la méthode à tester se doit de laisser remonter les exceptions afin qu'elles atteignent la méthode de test.

Junit
Pour déterminer si un test passe correctement ou non, il existe des méthodes prédéfinies dans le package junit.framework.Assert.
- `assertEquals`(« commentaires », obj1, obj2) : vérifie que les arguments sont égaux (pour autant que == existe entre les 2 types d'objet)
- `assertTrue`(« commentaires », exp) : vérifie que l'expression booléenne donnée est *true*
- `assertNotNull`(« commentaires », obj) : vérifie que l'objet est non *null*.

Ces fonctions vont générer des avertissements au moment de lancer les tests à travers un «test runner ». Il existe un test runner intégré à JBuilder. Pour l'utiliser il suffit de cliquer dans le menu Run | Run test « Testnom.java » using defaults. Bien sûr cette fonctionnalité n'existe que pour des classes du type « TestCase ».

Test Suite

Il sera utile de pouvoir choisir de lancer un ensemble de tests pour une classe, ou pour un (sous)package. Pour cela, il existe les « Test Suite ».
Elles se créent de la même manière qu'une classe de test, à la différence de choisir « Test Suite » et non « Test Case » dans l'interface. JBuilder génère alors la classe permettant de lancer tous les tests des classes que vous avez choisis. Cette classe est souvent appelée « AllTests.java » et contient une méthode « suite() » qui sera à complèter à chaque nouvelle classe du même package que vous souhaitez ajouter dans la suite de tests.
Exemple :
Vous avez une classe cds.util.Database*, vous créez une classe* cds.util.TestDatabase *puis une TestSuite* cds.util.AllTests *qui contient dans sa méthode « suite() » les lignes suivantes :*

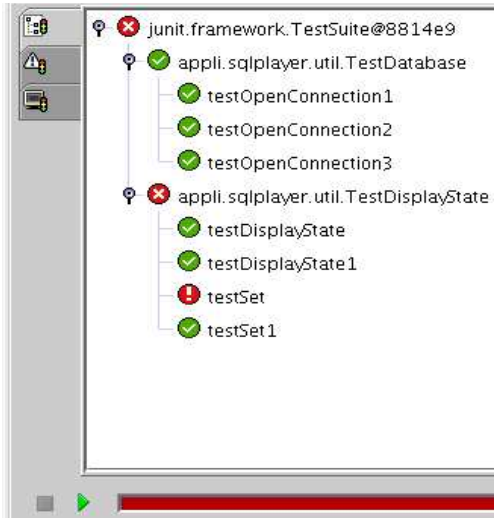
```
TestSuite suite = new TestSuite();
suite.addTestSuite(TestDatabase.class);
```

Au moment d'executer les tests de ce package, AllTests lancera toutes les méthodes qui commencent par « test » appartenant aux classes contenues dans la suite grâce à `addTestSuite`.

Il peut être utile de créer soi-même une méthode : `public TestSuite suite()` dans une classe testCase pour préciser manuellement les méthodes que vous souhaitez voir testées sans la contrainte qu'elles commencent par « test ». Dans ce cas, le constructeur doit prendre en entrée un `String` et son utilisation est :

```
suite.addTest(new TestDatabase(« testOpenConnection3 »));
```

Vous pouvez voir visuellement les résultats des tests (ici dans JBuilder):



Les différents tests d'ouverture de connection de la classe Database ont réussis.

Le test d'affectation de l'état de la connection : testSet a échoué.

Références :

- Site de Junit : www.junit.org
- Aide et tutorial de JBuilder : aussi accessible directement en ligne : http://info.borland.com/techpubs/jbuilder/jbuilder9/bajb/unittest.html
- Junit HowTo : http://admc.com/blaine/howtos/junit/
- « Tests unitaires en Java », Joannes Link, Dunod, 2003
- News : http://groups.yahoo.com/group/junit/
- Exemples et conseils en français :
  - http://smeric.developpez.com/java/astuces/tests/
  - http://www.design-up.com/methodes/testsunitaires/index.html