What is Jbehave?

To express stories & behavior.to explain application behavior at high level.

User stories are written using 3 terms

* Given(initial context)
* When(occurances)
* Then(expected outcome)

JBehave is a framework for [Behaviour-Driven Development](http://en.wikipedia.org/wiki/Behavior_driven_development) (BDD). BDD is an evolution of test-driven development (TDD) and acceptance-test driven design, and is intended to make these practices more accessible and intuitive to newcomers and experts alike. It shifts the vocabulary from being test-based to behaviour-based, and positions itself as a design philosophy.

* **Phrases the scenarios from story files**
* **Maps tem to java code**
* **Runs them via Junit tests**
* **Generates reports**

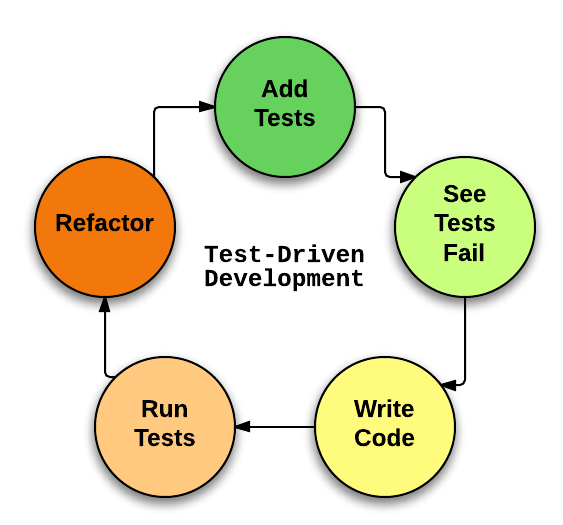
**TDD:**

TDD, the idea seemed to be pretty simple. Just by doing a little word swizzling, obviously TDD is when you have tests that drive your software development.

If we were to unpack the definition of TDD a bit more, we’d see that it is usually broken up into five different stages:

1. First the developer writes some tests.
2. The developer then runs those tests and (obviously) they fail because none of those features are actually implemented.
3. Next the developer actually implements those tests in code.
4. If the developer writes their code well, then the in next stage they will see that their tests pass.
5. The developer can then refactor their code, add comments and clean it up, as they wish because the developer knows that if the new code breaks something, then the tests will be an alert by failing.

The cycle can just continue as long as the developer has more features to add. A flowchart is given below of this cycle:

Test-driven development flowchart.

**BDD:**

In contrast to TDD, BDD is when we write behavior & specification that then drives our software development. Behavior & specification might seem awfully similar to tests but the difference is very subtle and important.

**TDD vs BDD:**

The choice between TDD and BDD is a complicated one. It depends on if there is an appropriate testing framework for your given target language, what your coworkers are comfortable with, and sometimes other factors.

Some argue that BDD is always better than TDD because it has the possibility of eliminating issues that might arise when using TDD.

The key to BDD is that it **might** prevent issues; it isn’t guaranteed to. Issues like poor code organization, bad design practices, etc. will still persist. You’ll just have a lower likely hood of writing bad tests and thus have more robust features.

Introduction to JBehave
JBehave is a framework for Behaviour-Driven Development
 How to add new test scenario
• Write new test scenario on the Gherkin like:
Scenario: “describes what should be passed”
Gi...

**VERSIONS:**

**Jbehave** – 3.94

3.9(2013)

**JDK Java**-1.6

**Junit**-4.11

4.12

**Jmeter**-3.0