# Welcome to JUnitScript

## 1. Features at a Glance

- Use scripting languages to unit test Java.
- Unit test scripting languages.
- JUnit and Ant integration.
- Use any language supported by BSF.
- Testing extensions to javascript and beanshell.
- Pass commandline arguments to test cases.

## 2. What is JUnitScript?

*JUnitScript* is set of tools that allow you to use scripting languages to write unit tests that can be embedded inside normal JUnit tools. The goal of the project is to make writting tests as easy and as natural as possible. JUnitScript not only lets you use scripts for testing Java, but also lets you unit test your scripts which may be invaluable for complex DHTML.

Scripting languages may confer some advantages over using Java for testing. Often, scripting provides a more forgiving environment, for example, weakly typed languages tend to yeild more compact code with a greater resiliance to minor changes in interface. Not needing to compile test cases can speed up the test cycle. Scripting languages can also, often provide abstractions unavailable in Java.

Conversely, scripting poses some problems. Errors in the script may be harder to find than in compiled, well typed languages as the script might have to be entirely run to find syntax errors. Sometimes, the speed of an interpreted language can make it unsuitable for all tasks. Also, while additional abtractions may be offered that Java cannot provide, the reverse may be true and useful abstractions from Java may not be available.

The choice of whether to use scripting or not for testing will vary depending upon the situation. Fortunately, scripted tests can be intermixed with normal JUnit tests. Indeed, if a test script is written in a language that is close to Java, then the script might get translated into Java if it proves to be a good and useful test.