# Unit Testing for InproTK

Casey Kennington

June 19, 2012

## 1 Unit Testing Overview

## 2 Procedure

-naming convention ClassNameUnitTest -one file per class -package doesn't matter -test every public method -use assert methods -use fail methods in exception catches -add to test.AllTests SuiteClasses list, make sure to import -comment

## 3 Unit Testing in Eclipse

This follows a nice step by step guide: http://www.vogella.com/articles/JUnit/article.html

I recommend using Eclipse for unit test development. It has a very nice framework for making new classes and you can specify which class you want to test.

A nice, general exmample that tests all constructors and methods: test.unit.inpro.annotation.Label

### 3.1 Testing Private Methods

Example: test.unit.inpro.annotation.AnnotationUtilTest.java

For most cases, one simply creates an object of a class and tests the public methods, some of which give access to private methods. But, in some cases you will need to test private methods. Assuming you want to call a method that takes a String as an argument, one can call it with the following:

MyClass object = new MyClass();
Method m = object.getDeclaredMethod("myMethodName", String.class);
m.setAccessible(true);
m.invoke(object, "String");

When the method is PRIVATE STATIC, do the following:
Method m = MyClass.class.getDeclaredMethod("myMethodName", String.class);
m.setAccessible(true);
m.invoke(null, "String");

When a method takes multiple arguments, use the following instead of String.cass as the second argument:
Class[] argtypes = Class[2];
argtypes[0] = String.class;

argtypes[0] = List.class;
m.invoke(object, arg1, arg2);

# 4 Running a Test Suite

run test.AllTests as junit test