Test Driven Development
In Action!

by Jon Kruger
What is Test Driven Development?

- A software development technique where you write automated unit tests *before* you write your implementation code
- A technique for ensuring good quality and good design
- Awesome!
Example of a unit test

[TestFixture]
public class When_using_the_calculator
{
    [Test]
    public void Should_add_two_numbers()
    {
        int result = new Calculator().Add(2, 3);
        result.ShouldBeEqual(5);
    }
}
Unit Tests vs. Integration Tests

- Unit tests:
  - Tests a small unit of functionality
  - Mock or “fake out” external dependencies (e.g. databases)
  - Run fast

- Integration tests:
  - Test the whole system working together
  - Can run slow
  - Can be brittle
Unit Testing Frameworks

- .NET
  - NUnit, MSTest, MBUnit, xUnit, MSpec
- Java
  - JUnit, TestNG
- Ruby
  - RSpec, Test::Unit, Shoulda
The TDD Process

1. Write a test (or tests) for something. Since you haven’t written the implementation code yet, the tests should fail.
2. Write just enough code to get the test to pass.
3. Move on or refactor

“Red – Green – Refactor”
Code!
Benefits of TDD

We know that our code is working!
Benefits of TDD

We know that our code will continue to work
Benefits of TDD

We didn’t write bugs
Benefits of TDD

We know when we are done
We incrementally translated the requirements
Behavior Driven Development

- Testing the *behavior* of the system (not just data returned by a method)
- Defining what it means for your system to work correctly (not just verifying that code works)
Benefits of TDD

Concentrate on the requirements/tests, then concentrate on implementation
Benefits of TDD

We only wrote as much code as we needed to make the tests pass
Benefits of TDD

Our tests helped us design our code
“Clean code that works is the goal of Test Driven Development.”

-- Ron Jeffries
Benefits of TDD

We had to write testable code
Benefits of TDD

Our tests are documentation of what our code does
Benefits of TDD

Our tests are documentation of what our code does

• Someday someone other than you will have to understand your code
• Someday you will wonder what your code was supposed to do
• Living, breathing documentation!
Benefits of TDD

We can quickly regression test our code
Benefits of TDD

We can quickly regression test our code

- Fewer bugs
- Able to release more often
- Less time spent doing manual testing
- Prevent our app from becoming “legacy code”
Benefits of TDD

Peace of mind!
But I don’t have time to do TDD!

- I don’t have time to fix bugs
- I don’t have time to constantly step through code to see if it’s all working
- I don’t have time to figure out what your code is supposed to do
- I don’t have time to figure out if my changes will break something in your code
- I don’t have time to rewrite code

Think long-term, not short-term!!
"If I don't need to make it work, I can go a lot faster."

-- Kent Beck
The Cost of Unit Testing

The Cost of Not Unit Testing

The Relative Cost of Fixing Defects

Resources - Cost of unit testing

• Microsoft Research – “Realizing quality improvement through test driven development: results and experiences of four industrial teams”

• Cost of Testing, by Misko Hevery (Agile Coach/Java developer at Google)

• TDD Derangement Syndrome, by Uncle Bob Martin
Resources - Tools

- .NET
  - NUnit – [http://nunit.org](http://nunit.org)
  - Should – [http://should.codeplex.com](http://should.codeplex.com)
- Java
  - JUnit – [http://junit.org](http://junit.org)
  - TestNG – [http://testng.org](http://testng.org)
- Ruby
  - RSpec – [http://rspec.info](http://rspec.info), or gem install rspec
Behavior Driven Development

So How do You Introduce TDD into an Organization or Team?, by Jeremy Miller

How to get started with TDD, by Misko Hevery (Java examples)

TDD Starter Kit – Sample Projects and Links (C# examples)

Pair Programming Bot
Resources – Books

- The Art of Unit Testing
  - by Roy Osherove

- Test Driven Development: By Example
  - by Kent Beck

- Test Driven Development: A Practical Guide
  - by David Astels

- The RSpec Book: Behaviour Driven Development with RSpec, Cucumber, and Friends
  - by David Chelimsky, Dave Astels, et. al.
Resources – Practice!

- String Calculator kata
  - [http://osherove.com/tdd-kata-1/](http://osherove.com/tdd-kata-1/)

- Bowling Game kata

- Prime Factors kata

- Greed game (part of the Ruby koans)
  - [http://github.com/edgecase/ruby_koans](http://github.com/edgecase/ruby_koans)

- Katacasts (watch screencasts of people doing various katas)
• TDD Boot Camp (.NET)
  ▪ http://tddbootcamp.com

• Pillar Technology (Java)
  ▪ http://pillartechology.com

• EdgeCase (Ruby on Rails)
  ▪ http://edgecase.com
My Info

- Email: jon@jonkruger.com
- Twitter: @JonKruger
- Blog: http://jonkruger.com/blog

- These slides: http://tinyurl.com/tdd-in-action