TEST-DRIVEN DEVELOPMENT

THE T IN TDD IS WRONG

Example would be better, but its too late

xUnit.net

- Replaced [Test]
- [Fact] – a fact is an assertion that should be true for the specified data
- [Theory] - A theory is an assertion that should be true for any human-chosen data.

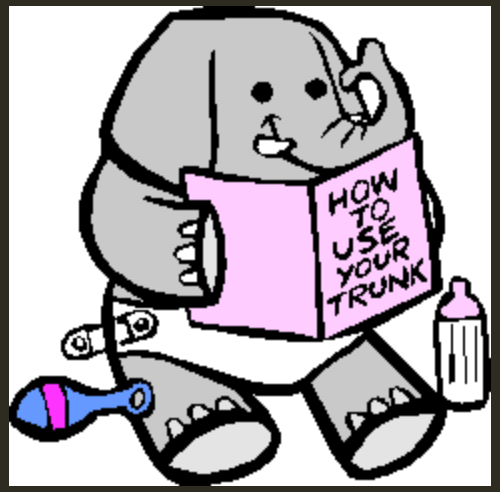
A WHIRLWIND TOUR THROUGH TEST DRIVEN DEVELOPMENT (TDD)

People learn by example

- Solving math problems
  - 2 + 2 = 4
  - 4 + 4 = ?

So do elephants, apparently…

Baby elephants have absolutely no idea what to do with their short trunks. It takes them many months of learning by example before they can even use the trunk to squirt water into their mouths for a drink.



SOFTWARE CAN LEARN BY EXAMPLE, TOO

Simple Stack class

- Is empty at creation
- Is not empty after single push
- Same value popped as pushed…

```
[Fact]
public void SameValuePoppedAsPushed()
{
      Stack stack = new Stack();
      stack.Push(17);
      Assert.Equal(17, stack.Pop());
}
```

RED-GREEN-REFACTOR

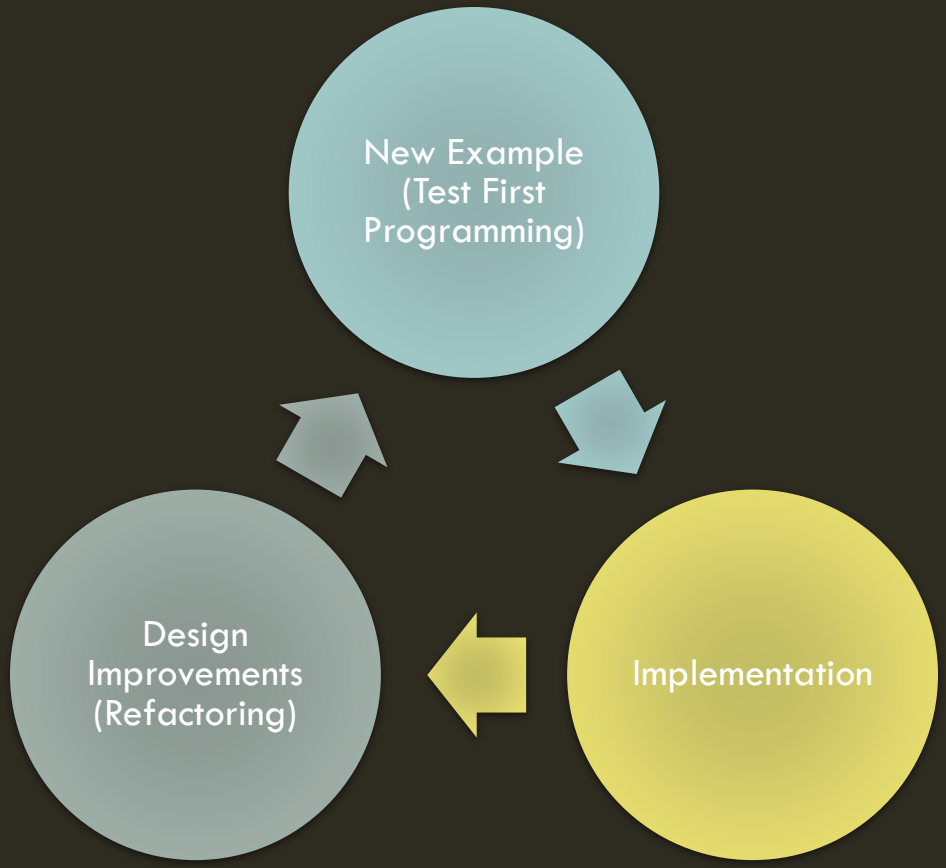
Add example to define functionality

- Test First Programming

Implement functionality

Remove duplication

- Refactoring



LESSONS AND OBSERVATIONS

Rhythm of development (Red-Green-Refactor)

- Write an example that fails to compile
- Make it compile but fail when run
- Implement the feature as simply as possible
- Watch it pass when run

Smallest size steps possible

REMINDERS

Start simply

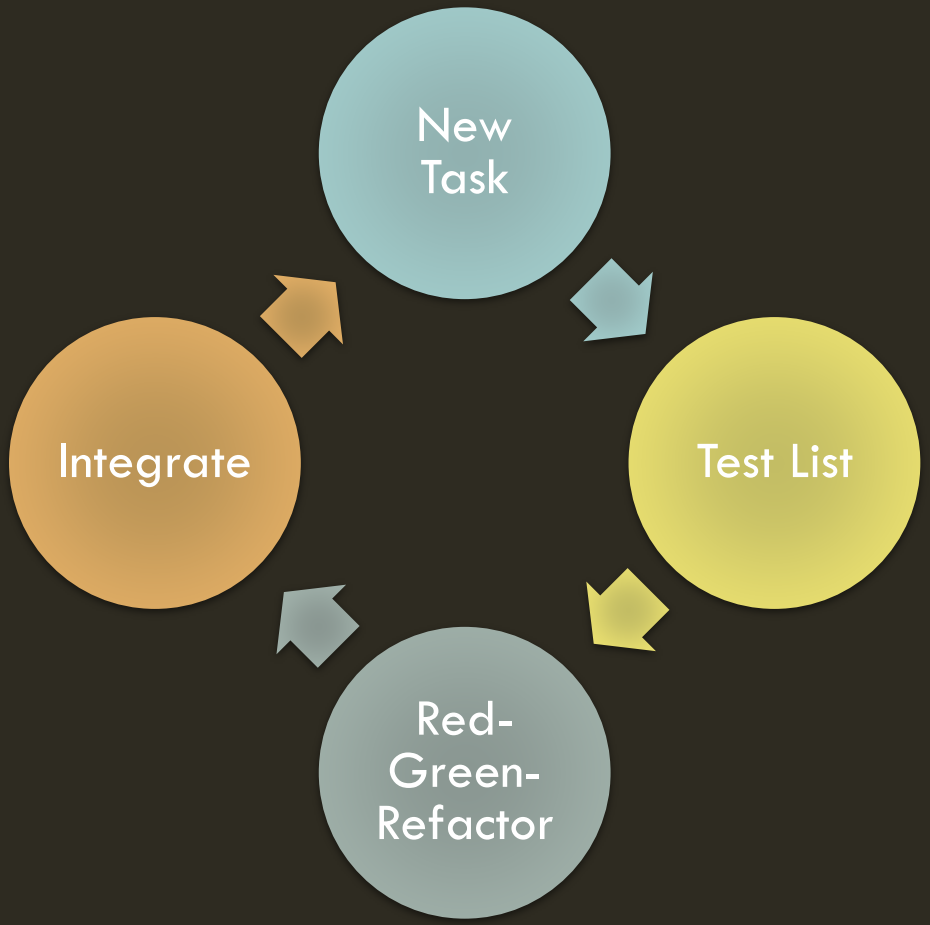
- Base cases
- Build towards ability to Add

Remember the rhythm

- Fails to compile
- Compiles but fails when run
- Simple implementation
- Passes when run

Small steps!

MACRO DEVELOPMENT CYCLE



TEST LIST

Brainstorm a list of tests for the task

Write them down!

List details the scope of the requirements

Evolves during the task as more tests reveal themselves

Describes completion requirements

TEST LIST EXERCISE

Brainstorm a list of tests for the following method

Class: List<T>

- public void CopyTo(T[] array)

Work with someone else

TDD EXERCISE

Build a simple last-in-first-out (LIFO) generic collection of objects, aka Stack<T>.

Primary Functionality

- Push, Pop, Peek

Ancillary Functionality

- CopyTo, Contains, Clear, ToArray, Count