Test Driven Development

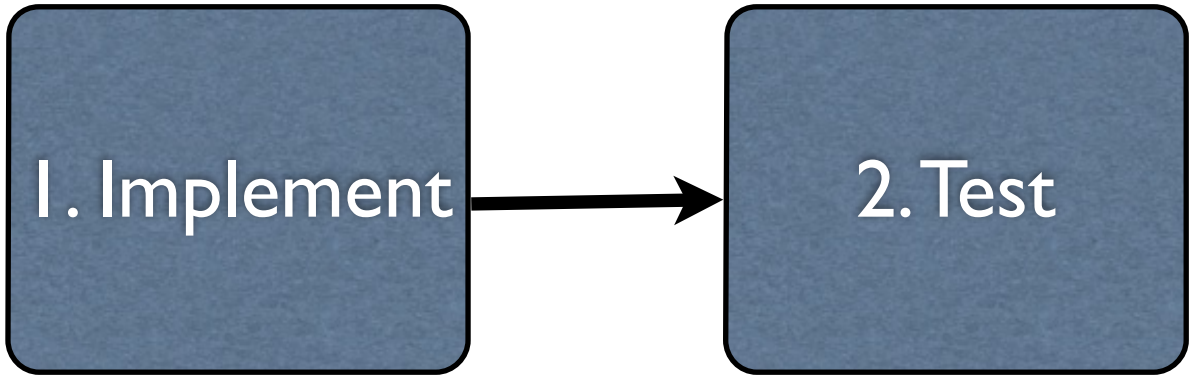
Mel Ó Cinnéide
School of Computer Science and Informatics
University College Dublin

Test Driven Development (TDD)

The notion of developing automated tests has lead to the idea of making testing more central to the development process.

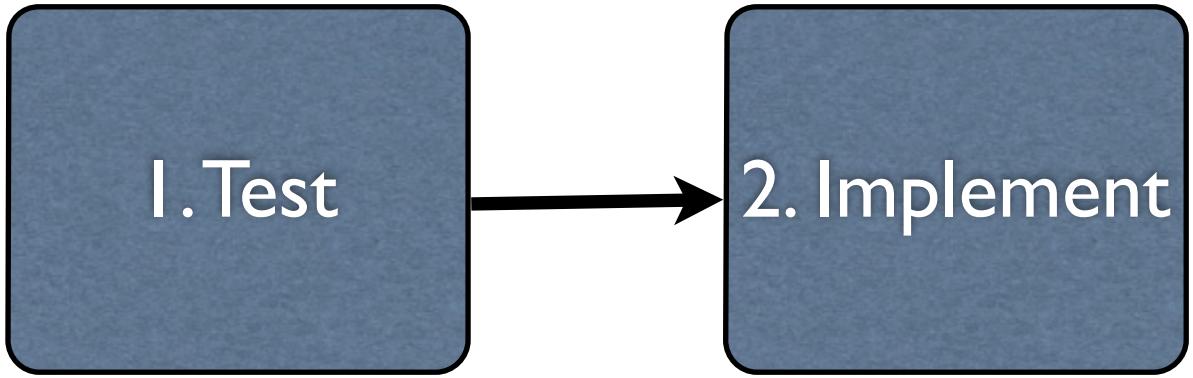
We'll look at this idea in the coming slides...

“Traditional” Order of Implementation and Testing



Test-Driven Development (TDD)

Aka **Test-First** Development



How can you test
before you implement?

Test-First isn't that unusual

Any specification implies certain tests that the program must pass, e.g.

Write a method that accepts a array of integers and returns their average value, e.g. for the input
[1,2,3,4,5]
the value 3 will be returned.

The specification defines the whole range of input/output pairs, and in this case even provides a concrete example.

The difference with test-first is that we use the unit tests to *drive* the development of the software.

Test-First in the Real World

Although test-first may seem odd in software, it's not uncommon in the real world.

E.g., mechanic doing a pre-check for an Irish National Car Test (NCT) exam, uses the tests to drive the work performed on the car.

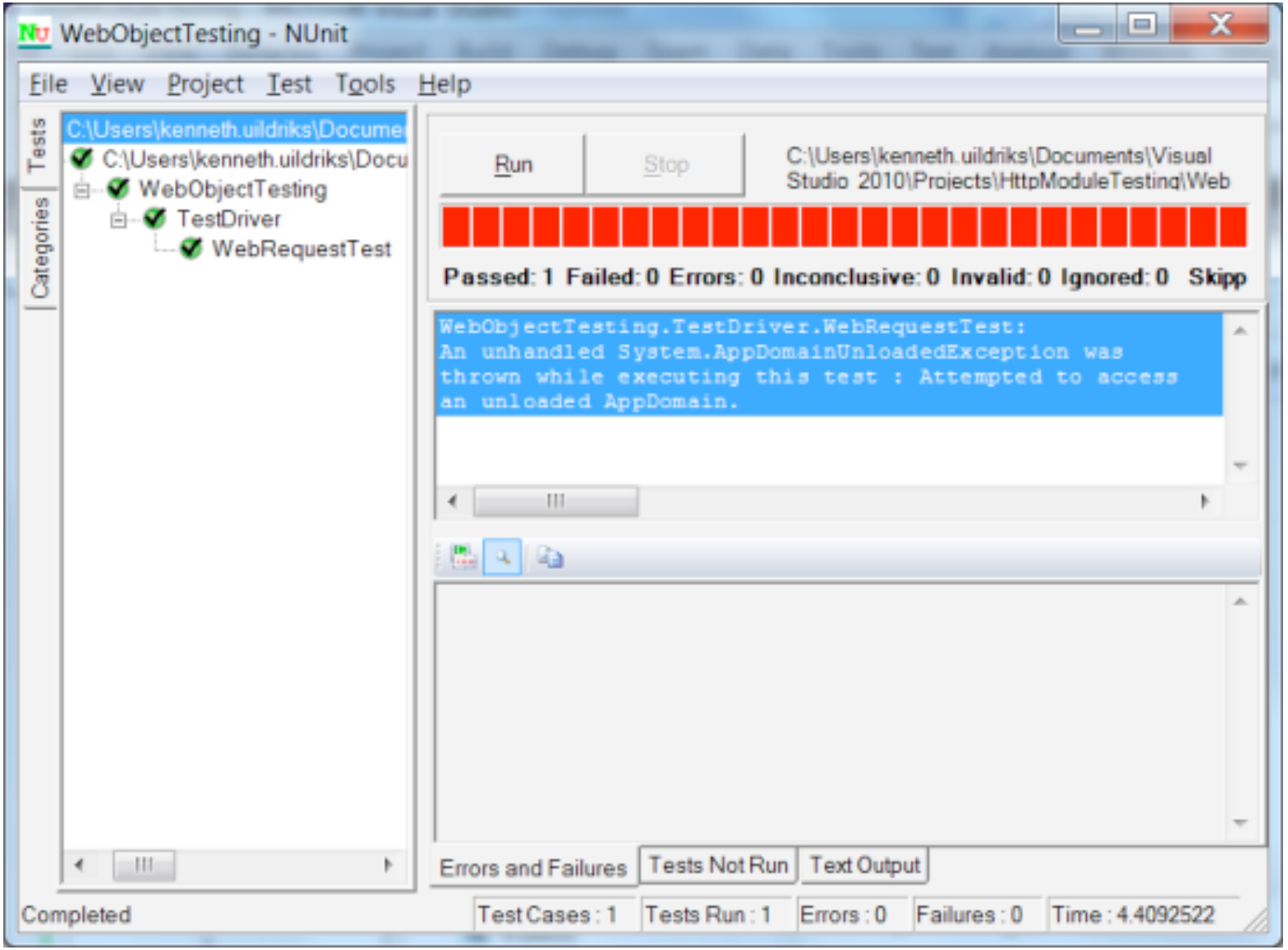
Test Readings					Limits			Results
Side-slip/Alignment Test					**+values are to the offside and - are to the nearside**			
Front Axle	0.0 m/Km				Outside	+14 to -14(m/km)	fail	PASS
Rear Axle	-6.0 m/Km				Outside	+18 to -18(m/km)	fail	PASS
Suspension Test	***Nearside***	***Offside***	***Imbalance***		Imbalance above	30 %	fail	PASS
Front Axle	33 Mm	24 Mm	27 %		Imbalance above	30 %	fail	PASS
Rear Axle	32 Mm	26 Mm	19 %					
Brake Test	***Brake Effort***		***Ovality***	***Imbalance***	Ovality above	90 %	fail	PASS
	Nearside	Offside	Nearside	Offside				
Front Axle	2.670 kN	2.400 kN	64 %	64 %	10 %	Imbalance above	30 %	fail	
Rear Axle	1.900 kN	2.190 kN	56 %	55 %	13 %	Imbalance above	30 %	fail	PASS
Parking-Brake	1.870 kN	1.810 kN			3 %	Imbalance above	50 %	fail	
Brake Performance(Car weight 1,426 Kg)					Performance less than	55 %	fail	
Brake Effort	65 %				Performance less than	16 %	fail	
Parking	26 %							
Exhaust Emissions					*Limits depend on Year of Manufacture*			
	Engine/Oil Temperature 110 °C				above	0.50 %	fail	FAIL/REFUSAL
Low Idle	CO 0.80 vol%				above	0 ppm	fail	PASS
(830 rpm)	HC 139 ppm							PASS
					between	0.97 and 1.03	fail	
High Idle	Lambda: 1.02				above	0.30 %	fail	FAIL/REFUSAL
(3,170 rpm)	CO 1.17 vol%				above	200 ppm	fail	PASS
	HC 112 ppm							
Head Light Aim	Nearside	Offside						PASS
Dip Beam	PASS	PASS						
Full Beam	N/A	N/A						
Fog Light	N/A	N/A						
Aux. Light	N/A	N/A						

Visual Defects				
ITEM	DESCRIPTION	REASON	LOCATION	
Registration Plates and Chassis Number(1)	Plates	Incorrect/Different	MiddleFront	FAIL/REFUSAL
Registration Plates and Chassis Number(1)	Plates	Incorrect/Different	MiddleRear	FAIL/REFUSAL

Test Results: FAIL/REFUSAL

Process of TDD

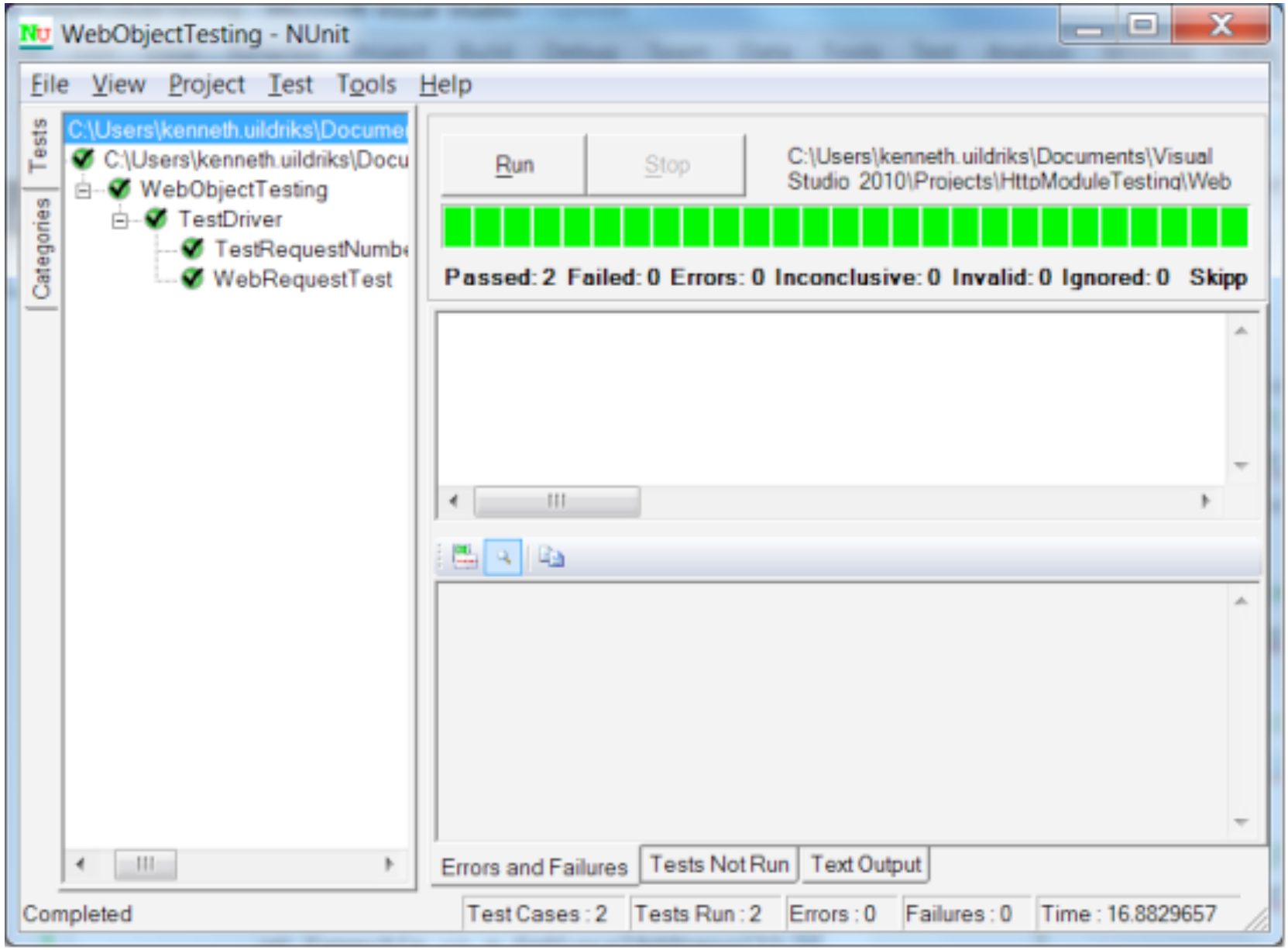
Write a Unit Tests that fails -- **RED**



Unit Tests should fail at first...

This provokes you to write code to make them pass.

Write code to make the Unit Test pass -- **GREEN**



Now tidy up your code -- **REFACTOR**

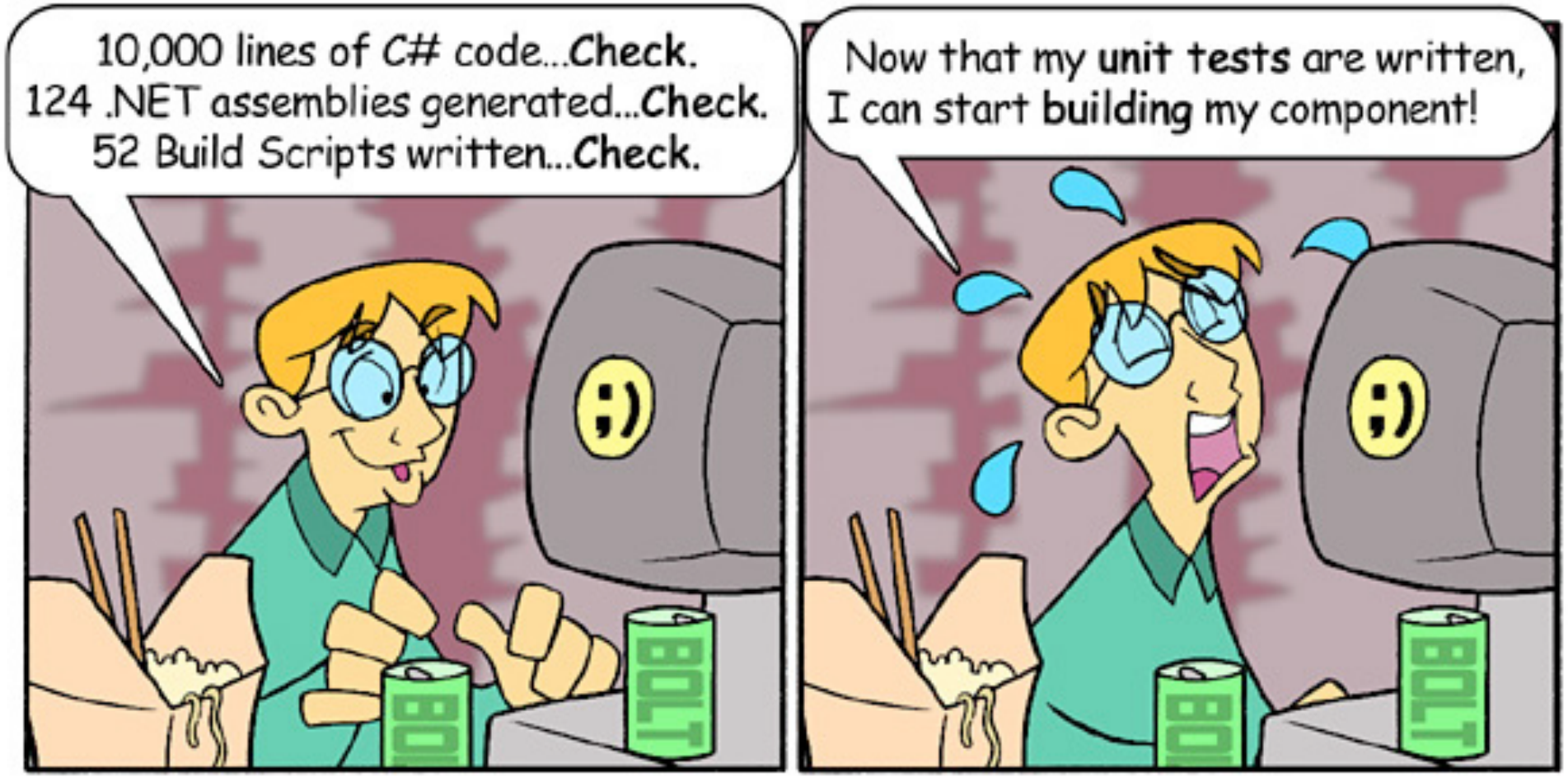
Tidy up your code, make sure there is no duplicated code or ugly bits. (This is a whole area in itself, termed **refactoring**.)

TDD Demo

Here we work through an example of using TDD to develop a simple class. Some points to note about the demo:

- The example is trivial of course; the purpose is to illustrate TDD
- We take very small steps, which is what you do when you're unsure of the problem. In a simple case like this we could have made larger steps.
- At every stage we have working code, and can confirm what it works by rerunning the unit tests

Test cases can take up a lot of code



... but this is not a bad thing.

Key Points

Testing is a vital part of software development.

Unit testing involves writing test cases for individual classes. A **unit testing framework** like JUnit can manage these test cases and run them on command.

Test-Driven Development (TDD) is where test cases are written first, as a way of driving the development of software. This is a key part of the **Agile** approach to software development.

In the lab today you'll be using TDD for a very simple example, and then for a more challenging one.

Demo

The problem: Create a simple shopping cart

Example taken from http://www.basicsbehind.com/tdd-by-example/

Demo

The problem: Create a simple shopping cart

Requirement: Create an empty shopping cart
When: An empty shopping cart created
Then: The product count should be 0

Example taken from http://www.basicsbehind.com/tdd-by-example/

Demo

The problem: Create a simple shopping cart

Requirement: Create an empty shopping cart
When: An empty shopping cart created
Then: The product count should be 0

Requirement: Add product to shopping cart
When: Add one unit of Irish Sausage, unit price €5
Then:

- product count should be 1
- the total value of cart should be €5

Example taken from http://www.basicsbehind.com/tdd-by-example/

Demo

The problem: Create a simple shopping cart

Requirement: Create an empty shopping cart
When: An empty shopping cart created
Then: The product count should be 0

Requirement: Add product to shopping cart
When: Add one unit of Irish Sausage, unit price €5
Then:

- product count should be 1
- the total value of cart should be €5

Offer: Buy 2 packs of Irish Sausage and get one free

Example taken from http://www.basicsbehind.com/tdd-by-example/

Demo

The problem: Create a simple shopping cart

Requirement: Create an empty shopping cart
When: An empty shopping cart created
Then: The product count should be 0

Requirement: Add product to shopping cart
When: Add one unit of Irish Sausage, unit price €5
Then:

- product count should be 1
- the total value of cart should be €5

Requirement: Apply offer on shopping cart
When: Add three units of Irish Sausage, unit price €5
Then:

- product count should be 3
- the total value of cart should be €10.00

Offer: Buy 2 packs of Irish Sausage and get one free

Example taken from http://www.basicsbehind.com/tdd-by-example/