

# Unit Testing Basic

Jul/13

*Dung Le Hoang*



# Outline

- Unit Testing at first sight
- What is Unit Testing?
- Unit Testing Framework
- Test First Development

# Unit Testing

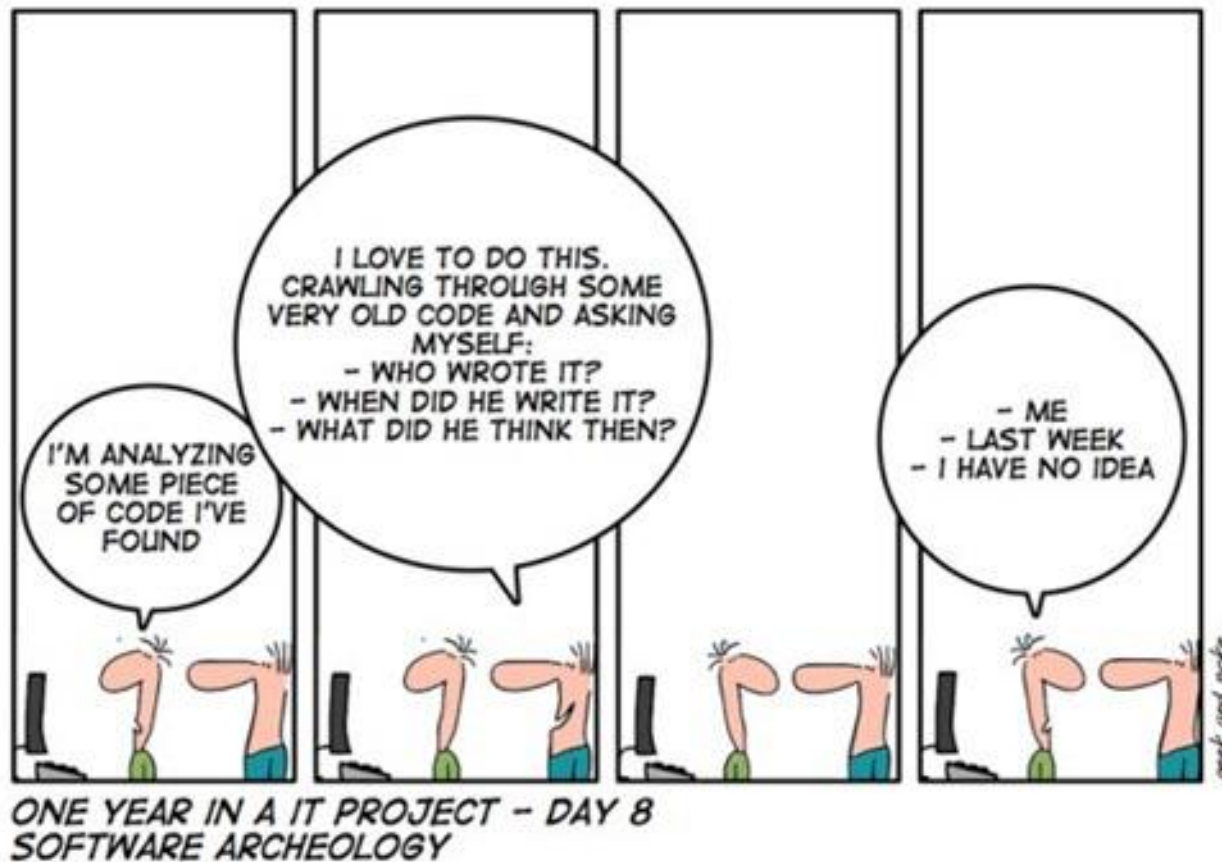


# Unit Testing at first sight



# Why Unit Testing when I have QCs?

- But you may hear the same story..



# What is unit testing?

- “**Unit testing** is a method by which individual units of [source code](#), sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine if they are fit for use.”

“unit tests are so  
important that they  
should be a first  
class language construct”

- Jeff Atwood



← you know, the  
Coding Horror guy.

# What is an unit Test

- Verifies an atomic piece of code
- Test one specific behavior
- Each Test is autonomous

```
[TestMethod]
public void CheckPassword_ValidUser_ReturnTrue(){
    bool result = CheckPassword("User", "Pass");

    Assert.IsTrue(result);
}
```



# Benefits of unit testing

- Find problem early in the development cycle
  - Higher quality and fewer defects
- Facilitates change
- Simplifies integration
- Documentation
  - Living documentations
- Design
  - Well-crafted code





# Unit Test Framework



# .NET Unit Testing Framework

## .NET programming languages [\[edit\]](#)

Name	xUnit	Source	Remarks
csUnit	Yes		includes GUI, command line, VS2005 plug-in; supports C#, VB.NET, Managed C++, J#, other .NET languages, supports .NET 3.5 and earlier versions; integrated with <a href="#">ReSharper</a>
DbUnit.NET		<a href="#">[135]</a>	A .NET 2.0 unit testing framework for database access code
EMTF	No	<a href="#">[136]</a>	open source
Foq	No	<a href="#">[137]</a>	Lightweight type-safe and thread-safe <a href="#">mock object</a> library for F# with C# support.
Gallio		<a href="#">[138]</a>	Extensible, and neutral automation platform that provides a common object model, runtime services and tools (such as test runners) that may be leveraged by any number of test frameworks.
MbUnit	Yes	<a href="#">[139]</a>	Extensible, model-based NUnit compatible framework. Part of the <a href="#">Gallio Test Automation Platform</a> .
MSTest	No		A command-line tool for executing Visual Studio created unit tests outside of the Visual Studio IDE - not really a testing framework as it is a part of the <a href="#">Visual Studio Unit Testing Framework</a> .
NaturalSpec	No	<a href="#">[84]</a>	Domain-specific language for writing specifications in a natural language. Based on NUnit.
NMate		<a href="#">[140]</a>	NUnit and PartCover Code Generation and integration Addin for Microsoft Visual Studio 2005/2008
NUnit	Yes		includes GUI, command line, integrates into Visual Studio with <a href="#">ReSharper</a>
NUnitAsp			Based on NUnit
Pex	Yes	<a href="#">[133]</a>	<a href="#">Microsoft Research</a> project providing <a href="#">White box testing</a> for .NET, using the <a href="#">Z3</a> constraint solver to generate unit test input (rather than <a href="#">Fuzzing</a> ).
Quality Gate One Studio	No	<a href="#">[141]</a>	Commercial/freeware test framework for unit and integration testing that analyses dependencies between test cases to flow data between them. Supports combinatorial testing, multithreading and time-dependencies.
QuickUnit.net	No	<a href="#">[142]</a>	Implement unit tests without coding. Minimalist approach to test driven development.
Rhino Mocks	Yes	<a href="#">[134]</a>	A dynamic mock object framework for the .NET platform.
Roaster	Yes	<a href="#">[143]</a>	NUnit-based framework and tools for the .NET Compact Framework
SpecFlow	Yes	<a href="#">[144]</a>	<a href="#">Behavior Driven Development</a> framework for .NET. Inspired by <a href="#">Cucumber</a> . Integrates with NUnit, MSTest, MbUnit, and others.
Specter	Yes	<a href="#">[145]</a>	<a href="#">Behavior-driven development</a> with an easy and readable syntax for writing specifications. Includes command line, optional integration with NUnit
TestDriven.NET		<a href="#">[146]</a>	Commercial
.TEST	Yes	<a href="#">[135]</a>	Commercial. Automated software quality solution that includes unit test generation and execution as well as reporting industry standard code coverage.
TickSpec	Yes	<a href="#">[147]</a>	<a href="#">Behavior-driven development</a> framework for .NET and Silverlight. Supports the Gherkin language as used by <a href="#">Cucumber</a> and extends it with combinatorial examples. Integrates with NUnit, xUnit, MbUnit and MSTest.
TPT	Yes	<a href="#">[12]</a>	<a href="#">Time Partition Testing</a> provides a .NET-API for the TPT-VM for testing controller software.
Typemock Isolator	Yes	<a href="#">[148]</a>	Commercial unit testing framework with simple API and test code generation features, supports C#, ASP.NET, SharePoint, Silverlight.
Visual Studio	No		The <a href="#">Visual Studio Unit Testing Framework</a> was first included in Visual Studio Team System 2005 where it integrated with the IDE, but not available in the most-used Standard Edition. From Visual Studio 2008 it is available also in Professional Edition. But it is still not included with Visual Studio Express editions.
Visual T#	Yes	<a href="#">[136]</a>	Visual T# is a unit testing framework and development environment integrated with Visual Studio. It includes T#, a programming language designed specifically to naturally express unit test intentions, and tools for compiling, running and maintaining them.
xUnit.net	Yes	<a href="#">[149]</a>	Developed by the original inventor of NUnit to be its successor. xUnit.net is currently the highest rated .NET unit testing framework <sup><a href="#">[137]</a></sup> due to it being leaner with a more refined syntax and lower friction usage than NUnit.

[http://en.wikipedia.org/wiki/List\\_of\\_unit\\_testing\\_frameworks#.NET\\_programming\\_languages](http://en.wikipedia.org/wiki/List_of_unit_testing_frameworks#.NET_programming_languages)

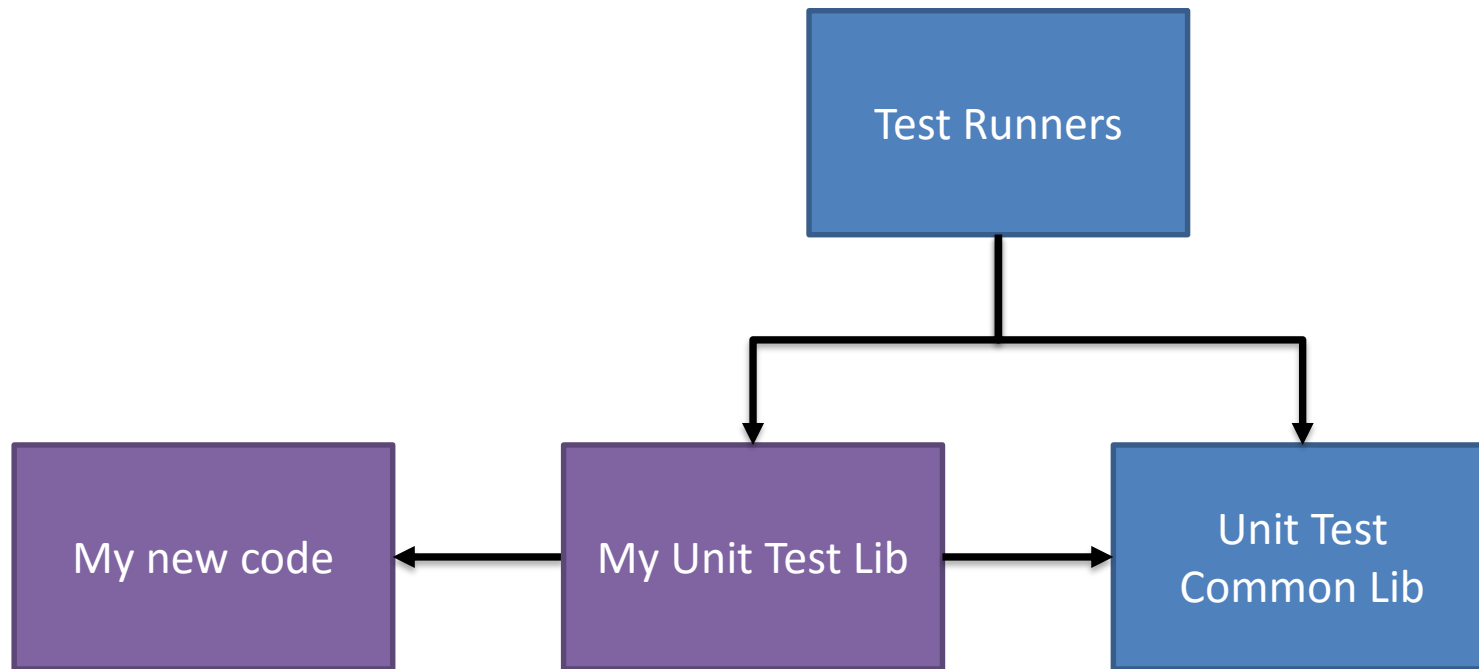
# .NET Unit Testing Framework



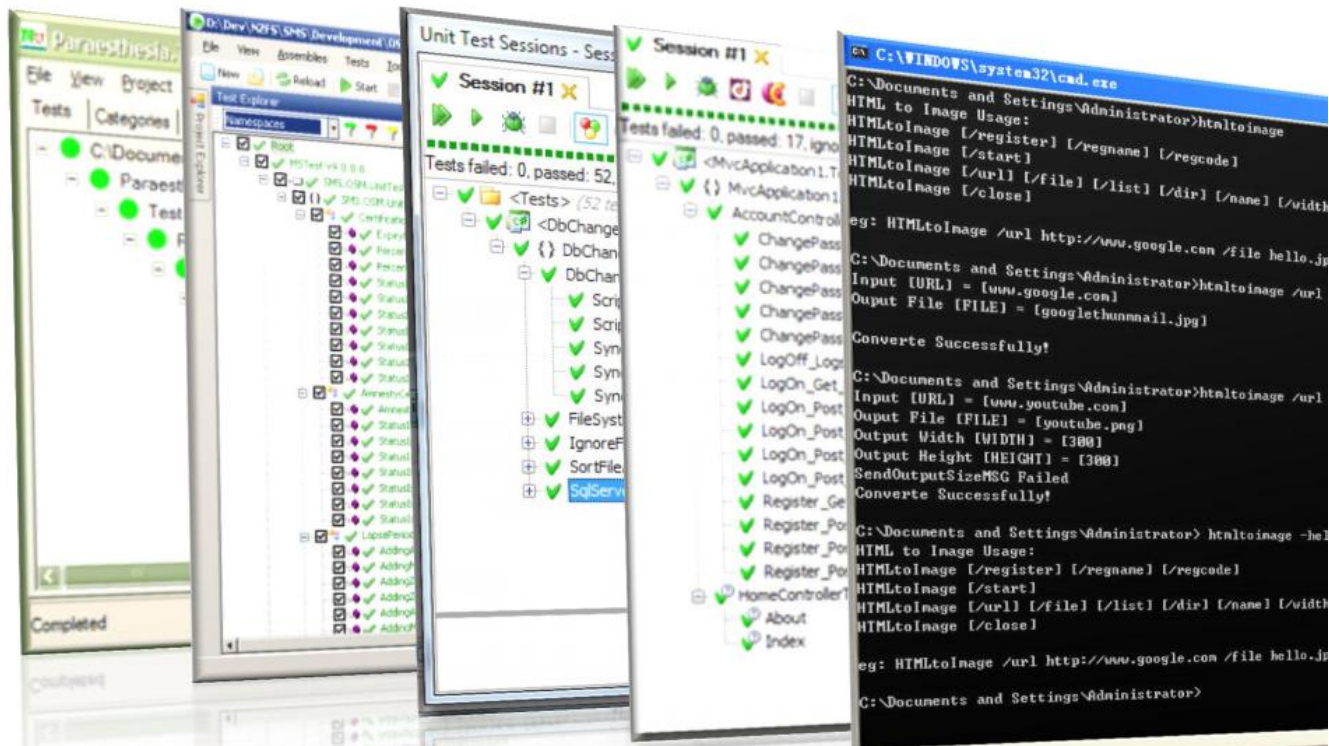
<http://www.nunit.org/>



# How Unit Test framework Work



# Unit Test Runners



# How unit-testing framework h

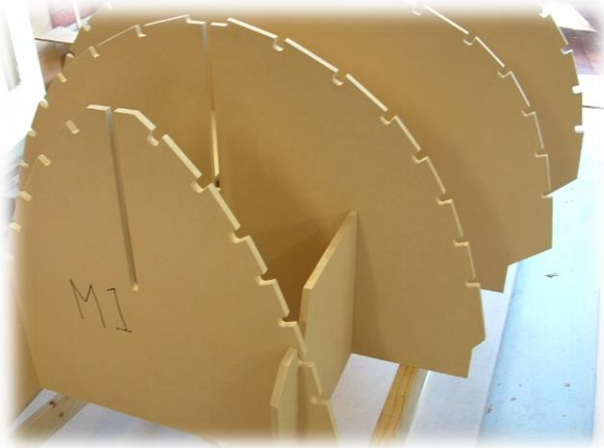
- Write tests easily and in a structured manner.
- Execute one or all of the unit tests
- Review the results of the test runs



# Test First Development

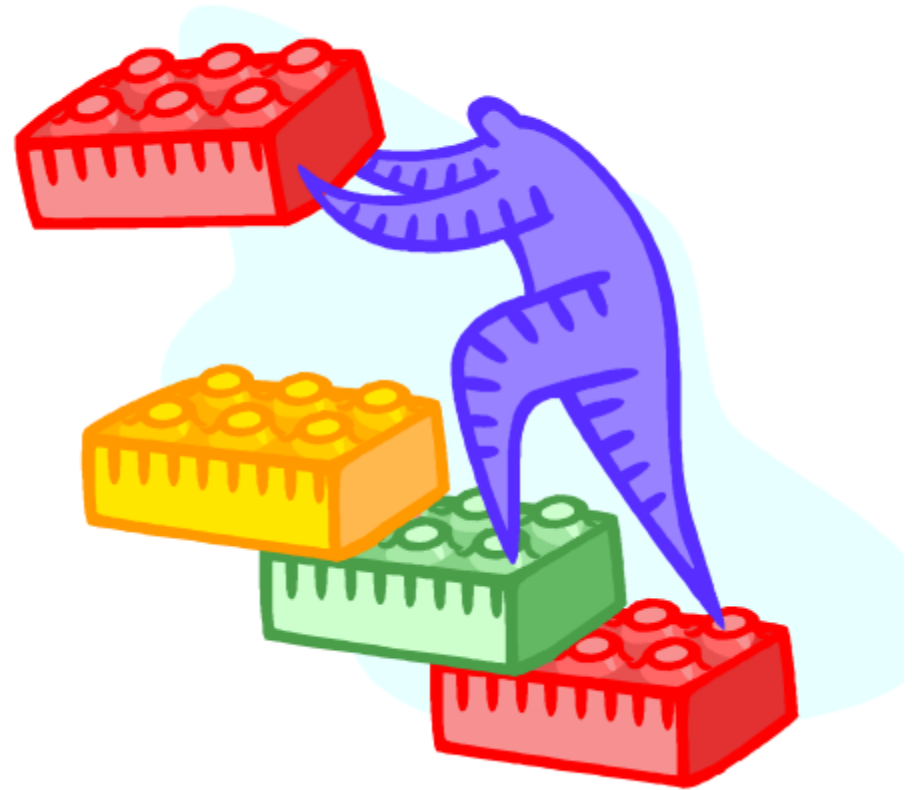


# Building Software Upon Existing Jigs

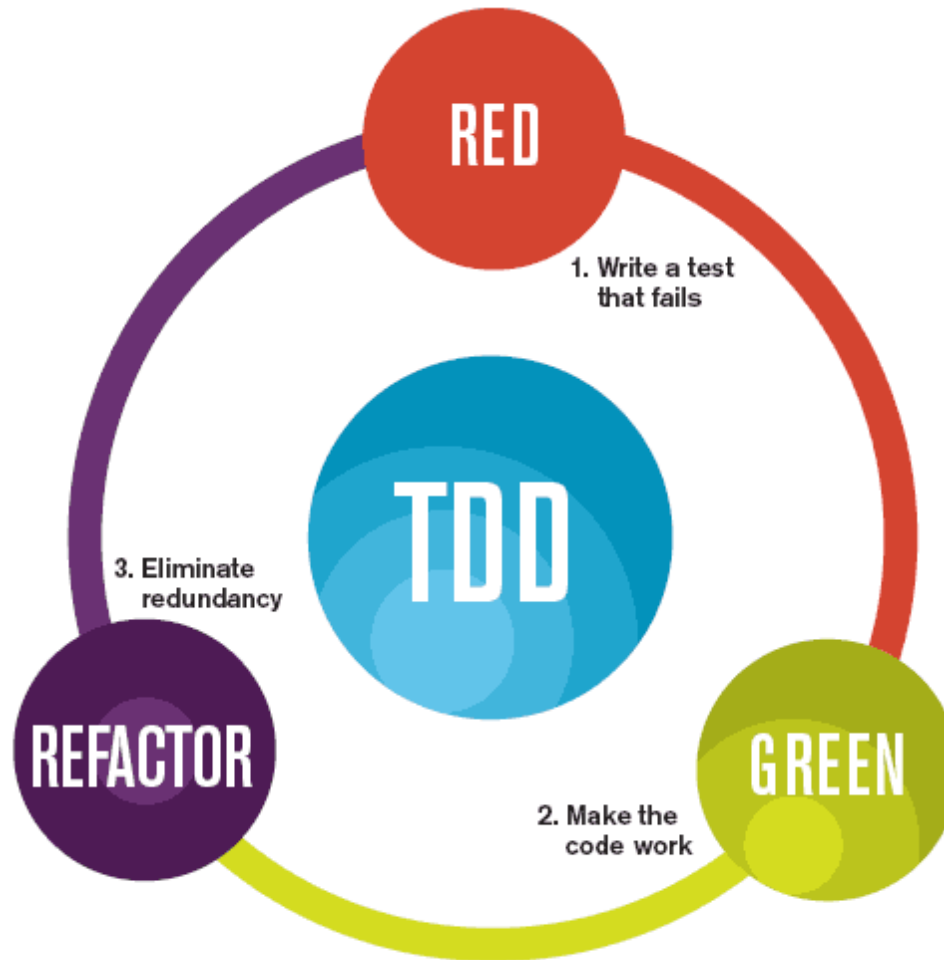


# Build the Jig only a moment a

- Software is often shaped as it is created
- The jig is often build only moments ahead of being used

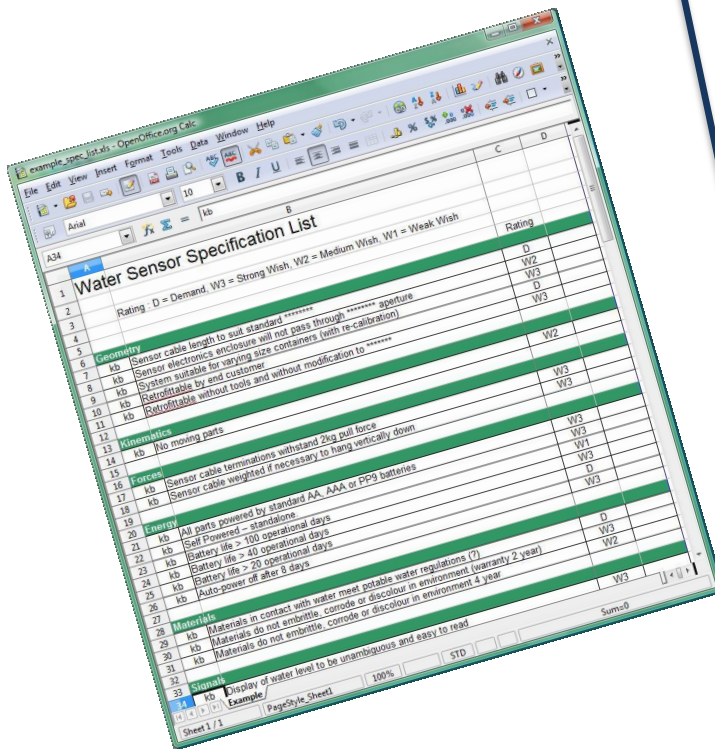


# Test Driven Development



The mantra of Test-Driven Development (TDD) is “red, green, refactor.”

# Tests are specs



The screenshot shows a spreadsheet titled "Water Sensor Specification List" with columns for specification details and a "Rating" column. The specifications are categorized into Geometry, Kinematics, Forces, Energy, and Materials. The ratings are D (Demand), W1 (Weak Wish), W2 (Medium Wish), and W3 (Strong Wish).

Spec ID	Specification	Rating
1	Rating: D = Demand, W3 = Strong Wish, W2 = Medium Wish, W1 = Weak Wish	
2		
3		
4		
5	Geometry	
6	kb Sensor cable length to suit standard *****	D
7	kb Sensor electronic's enclosure will not pass through ***** aperture	W2
8	kb System suitable for varying size containers (with re-calibration)	W3
9	kb System suitable by and customer	D
10	kb Retractable without tools and without modification to *****	W3
11	kb	
12	Kinematics	
13	kb No moving parts	W2
14	kb	
15	Forces	
16	kb Sensor cable terminations withstand 2kg pull force	W3
17	kb Sensor cable weighted if necessary to hang vertically down	W3
18	kb	W1
19	Energy	
20	kb All parts powered by standard AA, AAA or PPG batteries	W3
21	kb Self Powered - standalone	D
22	kb Battery life > 100 operational days	W3
23	kb Battery life > 40 operational days	W2
24	kb Battery life > 20 operational days	
25	kb Auto-power off after 5 days	
26	Materials	
27	kb Materials in contact with water meet potable water regulations (?)	W3
28	kb Materials do not embrittle, corrode or discolour in environment (yearly 2 year)	
29	kb Materials do not embrittle, corrode or discolour in environment 4 year	
30	kb	
31	Signal	
32	kb Display of water level to be unambiguous and easy to read	
33	Example	

```
<AnimalSpecs> (3 tests)
AnimalSpecs (3 tests)
  When_telling_a_dog_to_bark (3 tests)
    The_dog_barks_once_for_1_treat
    The_dog_barks_once_per_treat
    The_dog_doesnt_bark_for_no_treats
```