
Silicon Valley
Code Camp

## Contract-First Development with

# Microsoft Code Contracts and Microsoft Pex

Foothill College, October 8nd 2011

# Theo Jungeblut

- Senior Software Developer at Omnicell Inc. in Mountain View
- Has been designing and implementing .NET based applications , components and frameworks for more than 8 years
- Previously worked in factory automation with focus on component based software and framework development for 3 ½ years
- Degree in Software Engineering and Network Communications

theo@csharp-lighthouse.com

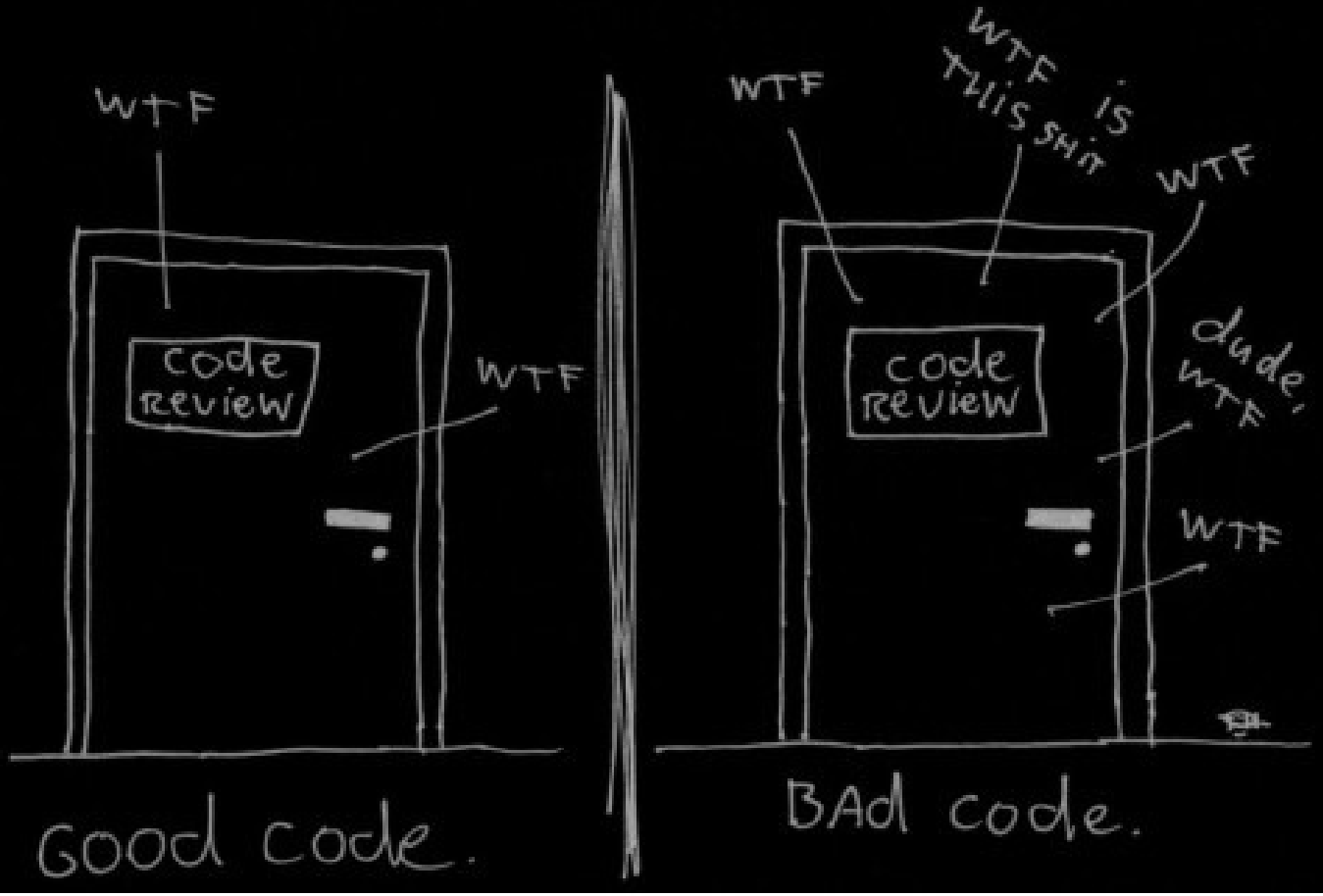
www.csharp-lighthouse.com

# Overview

- How to measure Code Quality
- Principles and Practices
- Microsoft Code Contracts
- How is your Code Coverage?
- Microsoft Pex & Moles
- Time for Source Code
- Summary
- Q&A
- References

Does writing Clean Code make us more efficient?

# The ONLY VALID measurement of code QUALITY: WTFs/minute



Clean Code???

We are here for
**Code Contracts, Pex and Mole!!**

# Clean Code is maintainable

**Source code must be:**

- readable & well structured
- extensible
- testable

# Clean Code is maintainable

**Source code must be:**

- **readable & well structured**
- ~~extensible~~
- **testable**

# Code Maintainability *

| | | |
|---|---|---|
| Principles | Patterns | Containers |
| Why? | How? | What? |
| Extensibility | Clean Code | Tool reuse |

* from: Mark Seemann's "Dependency Injection in .NET" presentation Bay.NET 05/2011

# Don’t repeat yourself (DRY)

# Don’t repeat yourself (DRY)

by Andy Hunt and Dave Thomas in their book “The Pragmatic Programmer”

```
// Code Copy and Paste Method
public Class Person
 {
   public string FirstName { get;  set;}
   public string LastName { get; set;}

   public Person(Person person)
   {
      this.FirstName = string.IsNullOrEmpty(person.FirstName)
                  ? string.Empty : (string) person.FirstName.Clone();

      this.LastName = string.IsNullOrEmpty(person.LastName)
                  ? string.Empty : (string) person.LastName.Clone();
   }

   public object Clone()
   {
      return new Person(this);
   }
}
```

```
// DRY Method
public Class Person
 {
   public string FirstName { get;  set;}
   public string LastName { get; set;}

   public Person(Person person)
   {
      this.FirstName = person.FirstName.CloneSecured();
      this.LastName = person.LastName.CloneSecured();
   }

   public object Clone()
   {
      return new Person(this);
   }
}
```

```
public static class StringExtension
{
   public static string CloneSecured(this string original)
   {
     return string.IsNullOrEmpty(original) ? string.Empty : (string)original.Clone();
   }
}
```

# Separation of Concerns (SoC)

# Single Responsibility Principle (SRP)

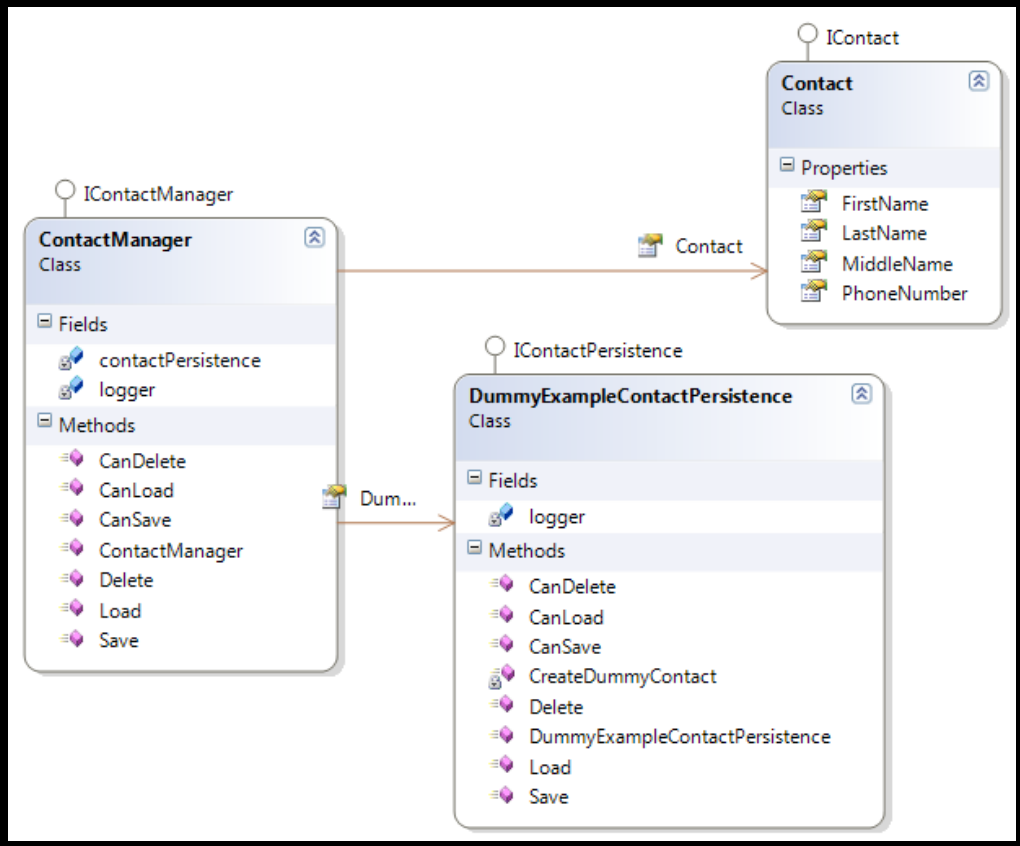
# Separation of Concerns (SoC)

probably by Edsger W. Dijkstra in 1974

• “In computer science, separation of concerns (SoC) is the process of separating a computer program into distinct features that overlap in functionality as little as possible.

•A concern is any piece of interest or focus in a program. Typically, concerns are synonymous with features or behaviors. “

http://en.wikipedia.org/wiki/Separation_of_Concerns



# Single Responsibility Principle (SRP)

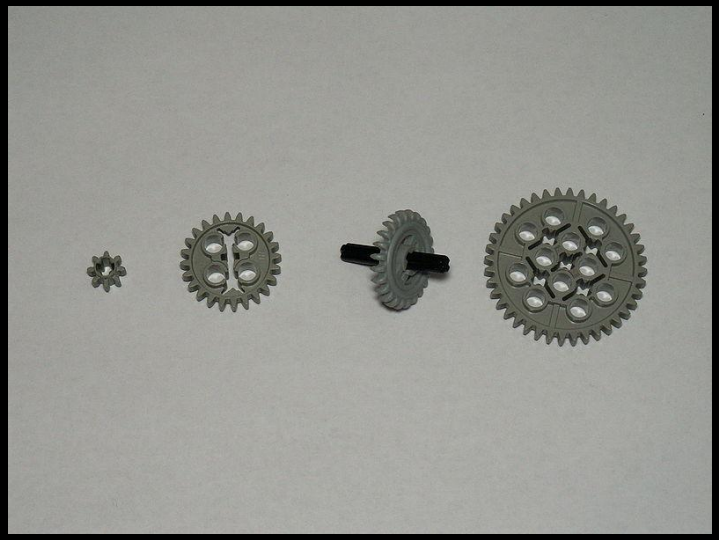
by Robert C Martin

"Every object should have a single responsibility, and that responsibility should be entirely encapsulated by the class."

http://en.wikipedia.org/wiki/Single_responsibility_principle

```
public class Logger : ILogger
{
    public Logger(ILoggingSink loggingSink)
    {}

    public void Log(string message)
    {}
}
```



http://www.ericalbrecht.com

# .NET Tools and their Impact

| Tool name | Positive Impact | Negative Impact |
|---|---|---|
| Resharper | compiling ++++ | VS responsiveness -- |
| FxCop | code quality ++ | compiling time - |
| StyleCop | code consistency +++ | compiling time - |
| StyleCop plugin for Resharper | compiling time +++ | VS responsiveness -- |
| Ghost Doc | automated docs | potentially worse doc |
| Spell Checker | fewer spelling errors ++ | performance -- |
| Code Contracts | testability, quality ++ | compiling time -- |
| Pex & Moles | automated test ++ | compiling time -- |


- Design-by-Contract programming
- Improved testability
- Static verification
- API documentation integration with Sandcastle

http://msdn.microsoft.com/en-us/devlabs/dd491992


# The Basic Idea

```
public class Logger : ILogger{
    public void Log(string message)    {
```

**Input Validation**

1.) Parameter Assignment
2.) Execute method logics

**Optional State Validation/Invariants**

3.) Possible Return Statement

**Result Validation**

```
    }}
```


# Method Overview

Method contracts should be written with the different elements ordered as follows:

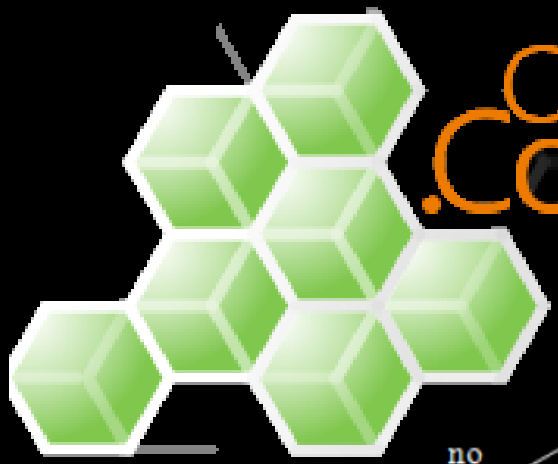
| | |
|---|---|
| If-then-throw | Backward-compatible public preconditions |
| Requires, Requires⟨E⟩ | All public preconditions |
| Ensures | All public (normal) postconditions |
| EnsuresOnThrow | All public exceptional postconditions |
| Ensures | All private/internal (normal) postconditions |
| EnsuresOnThrow | All private/internal exceptional postconditions |
| EndContractBlock | If using if-then-throw-style preconditions without any other contracts, place a call to EndContractBlock to indicate all previous if checks are preconditions. |

Code Contracts User Manual, 5.3 Contract Ordering, page 22

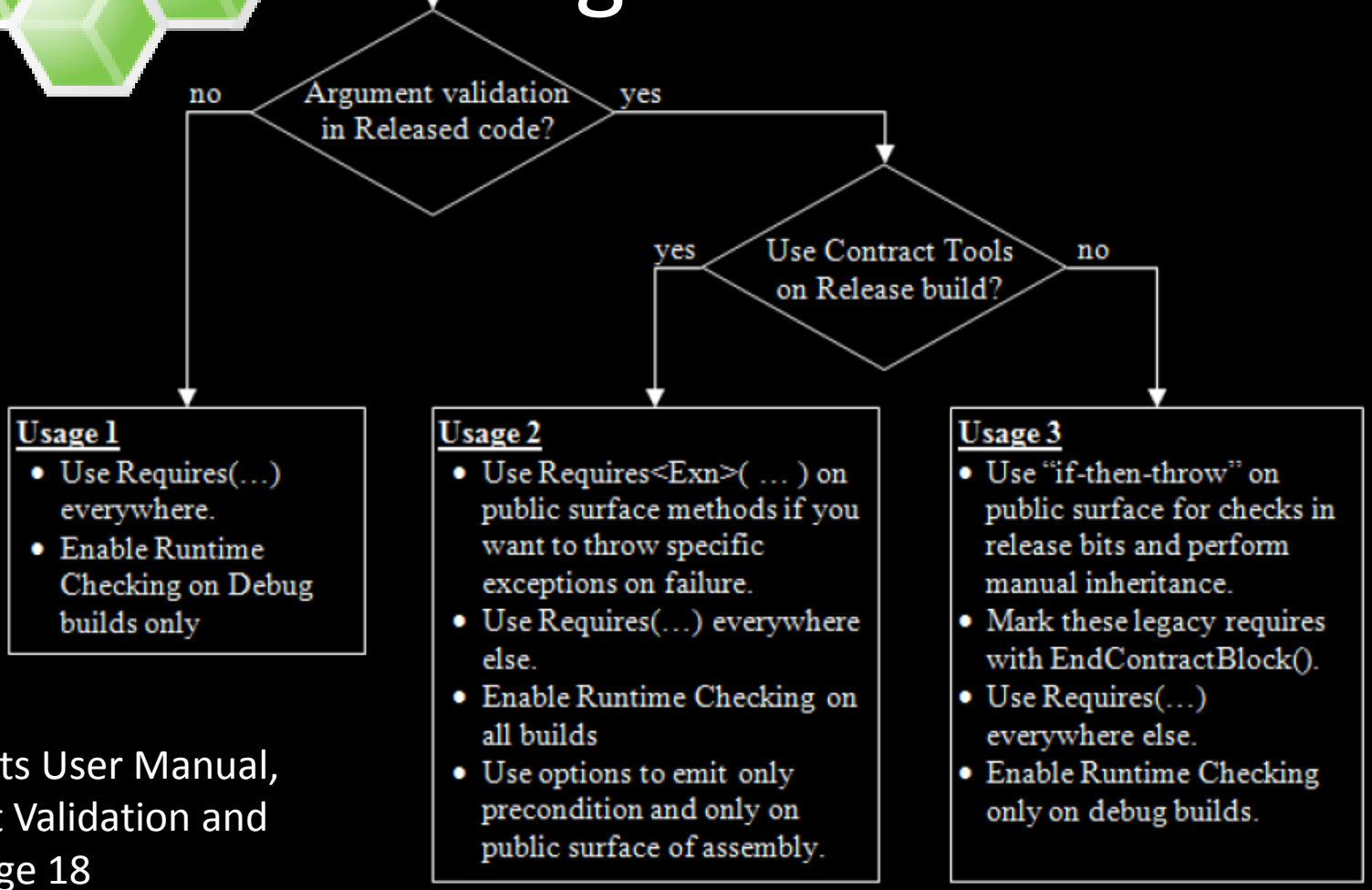

# Runtime Checking Levels

| Checking Level | Enabled Runtime Checks | | | | | | |
|---|---|---|---|---|---|---|---|
| | Legacy | Requires⟨E⟩ | Requires | Ensures | Invariants | Asserts | Assumes |
| Full | X | X | X | X | X | X | X |
| Pre and Post | X | X | X | X | | | |
| Preconditions | X | X | X | | | | |
| ReleaseRequires | X | X | | | | | |
| None | | | | | | | |

Code Contracts User Manual, 6.2.1 Runtime Checking Level, page 24




# Argument Validation



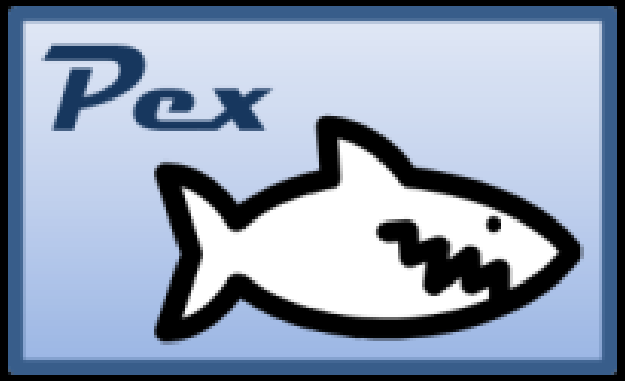

Code Contracts User Manual, 5.1 Argument Validation and Contracts, page 18

| Shortcut | Contract Snippet |
|---|---|
| cr | Contract.Requires(...); |
| ce | Contract.Ensures(...); |
| ci | Contract.Invariant(...); |
| crr | Contract.Result⟨...⟩() |
| co | Contract.OldValue(...) |
| cim | [ContractInvariantMethod]<br>private ObjectInvariant() {<br>  Contract.Invariant(...);<br>} |
| crn | Contract.Requires(... != null); |
| cen | Contract.Ensures(Contracts.Result⟨...⟩() != null); |
| crsn | Contract.Requires( !String.IsNullOrEmpty(...) ); |
| cesn | Contract.Ensures( !String.IsNullOrEmpty(Contracts.Result⟨string⟩()) ); |
| cca | Contract.Assert(...); |
| cam | Contract.Assume(...); |
| cre | Contract.Requires⟨E⟩(...); |
| cren | Contract.Requires⟨ArgumentNullException⟩(... != null); |
| cresn | Contract.Requires⟨ArgumentException⟩( !String.IsNullOrEmpty(...) ); |
| cintf | expands to an interface template and associated contract class |

# How is your Code Coverage?

# How is your Code Coverage?

- Writing enough Unit Tests takes Time
- Writing basic Unit Tests is not Fun




# Microsoft Pex & Moles

- Pex automatically generates test suites with high code coverage.
- Moles allows to replace any .NET method with a delegate.

http://research.microsoft.com/en-us/projects/pex/

# Microsoft Sandcastle

“Sandcastle produces accurate, MSDN style, comprehensive documentation by reflecting over the source assemblies and optionally integrating XML Documentation Comments. Sandcastle has the following key features:

- Works with or without authored comments
- Supports Generics and .NET”

http://sandcastle.codeplex.com/

Time for Visual Studio
as **well all love code**,
right !?

# Summary Code Contracts

- Code Contracts:
  - Better readable source code (SoC, SRP, DRY)
  - Static Analysis for Code Contracts
  - Impacts the compile time
- Pex & Mole:
  - Generated paramized Unit tests
  - Moles allows mocking also of static
- Conclusion ??

# Q & A

## Downloads, Feedback & Comments:

theo@csharp-lighthouse.com

www.csharp-lightouse.com

www.speakerrate.com/theoj

Graphic by Nathan Sawaya courtesy of brickartist.com

# References

Code Contracts

- See Dino Esposito's MSM Magazine series on Code Contracts
- http://msdn.microsoft.com/en-us/devlabs/dd491992.aspx
- http://research.microsoft.com/en-us/projects/contracts/
- http://www.codeproject.com/KB/cs/CodeContracts.aspx
- http://sachabarber.net/?p=811
- http://blogs.msdn.com/b/somasegar/archive/2009/02/23/devlabs-code-contracts-for-net.aspx
- http://designcoderelease.blogspot.com/2011/01/pex-moles.html
- http://www.agilitylux.com/practices/code-contracts-by-example/

Pex & Mole

- http://research.microsoft.com/en-us/projects/pex/

Sandcastle

- http://sandcastle.codeplex.com/
- http://stackoverflow.com/questions/3579655/does-sandcastle-support-code-contracts


Sunday 1:15 PM
Room: **1401**

# Clean Code

## Why Clean Code matters

Foothill College, October 9nd 2011

Please fill out the
online feedback, and…

# … thanks for you attention!


And visit and support the
Bay.net User Group
http://baynetug.org