

# Visual Studio Toolbox



Sergey Teplyakov  
Vlad Zubkis  
Mike Rybnikov

# Agenda

- **А зачем мне это?**
- Базовые возможности & Редактирование
- Поиск & Навигация
- Unit testing with Resharper
- Refactorings & Code Generation
- Coding Style & Tools
- **Дополнительные возможности ReSharper**



# А зачем мне это?

*Инструменты – средство усиления вашего таланта.  
Чем они лучше и чем лучше вы ими владеете, тем  
больше вы сможете сделать.*

Энди Хант и Дейв Томас. Программист-прагматик. Путь  
от подмастерья к мастеру



# А зачем мне это?

- А что если владение инструментом сэкономит 10 минут в день?
- $(1/6 \text{ ч} * 40\text{ч}) * 52 = ???$
- ~350ч/год



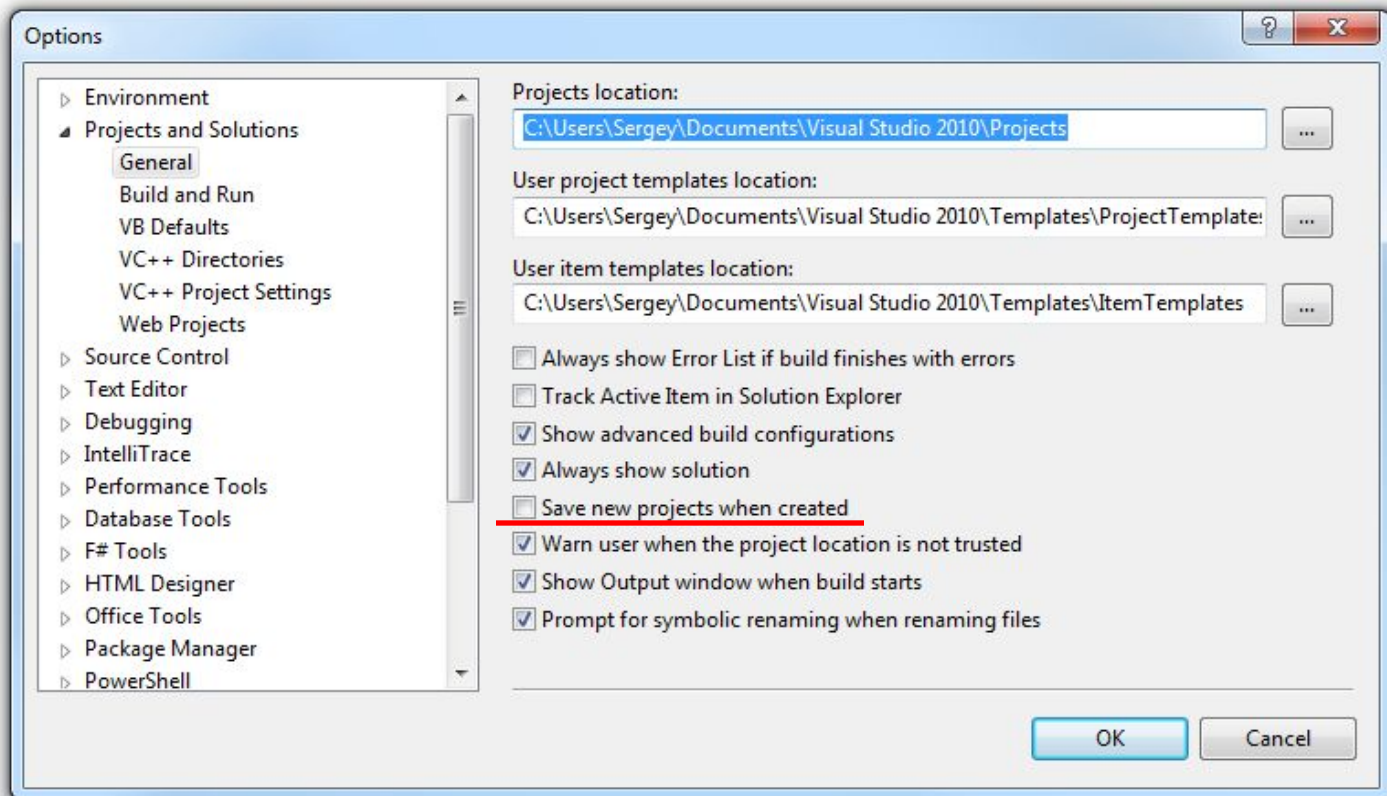
# Agenda

- А зачем мне это?
- **Базовые возможности & Редактирование**
- Поиск & Навигация
- Unit testing with Resharper
- Refactorings & Code Generation
- Coding Style & Tools
- **Дополнительные возможности  
ReSharper**



# [VS] Сохранение проектов

- Tools -> Options -> General -> Save new projects when created



# [VS] Вертикальное редактирование

- Выделяем блок текста с помощью ALT + клавиши курсора (или + мышь):

```
class TestClass
{
    public string s1;
    public string s2;
    public string s3;
    public string s4;
}
```

- Набираем “private” и меняем область видимости:

```
class TestClass
{
    priv string s1;
    priv string s2;
    priv string s3;
    priv string s4;
}
```

```
class TestClass
{
    private string s1;
    private string s2;
    private string s3;
    private string s4;
}
```

# [R#] Редактирование

- Import symbol completion (Shift + Alt + Space)
- Move code up, down, left, right (Ctrl + Shift + Alt + Up (Down, Left, Right))
- Quick documentation (Ctrl + Shift + F1)
- Duplicate a line or selection (Ctrl + D)
- [VS] Remove current line to Clipboard (Ctrl + L)



# [R#] Редактирование

- Import symbol completion работает по всем типам, а не только по импортированным!

```
var sb = new StringBui
```

 **StringBulder** (in System.Text)

Class System.Text.StringBuilder

Represents a mutable string of characters. This class cannot be inherited.

- Перемещение кода

```
Use Up/Down to move method ToString()
public override string ToString()
{
    return string.Format("LoginCommand: User name = {0}", UserInfo.Name);
}

public UserInfo UserInfo { get; private set; }
}
```

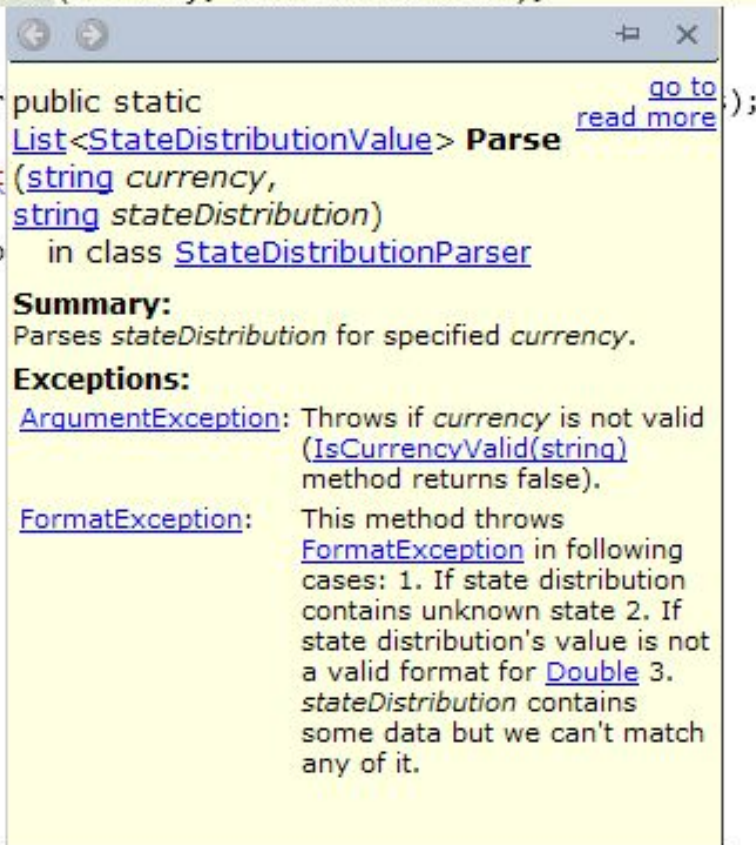
# [R#] Quick documentation

- Генерируется на основе Xml-документации

```
List<StateDistributionValue> parsedDistributions =  
    StateDistributionParser.Parse(currency, stateDistribution);
```

```
var parsedStringDistributions  
    StateDistributionParser.Cr
```

```
Console.WriteLine("Parsed dist  
  
return parsedStringDistributio
```



public static  
[List<StateDistributionValue>](#) **Parse** [go to](#)  
[\(string currency,](#) [read more](#)  
[string stateDistribution\)](#)  
in class [StateDistributionParser](#)

**Summary:**  
Parses *stateDistribution* for specified *currency*.

**Exceptions:**

[ArgumentException](#): Throws if *currency* is not valid ([IsCurrencyValid\(string\)](#) method returns false).

[FormatException](#): This method throws [FormatException](#) in following cases: 1. If state distribution contains unknown state 2. If state distribution's value is not a valid format for [Double](#) 3. *stateDistribution* contains some data but we can't match any of it.

# Agenda

- А зачем мне это?
- Базовые возможности & Редактирование
- **Поиск & Навигация**
- Unit testing with Resharper
- Refactorings & Code Generation
- Coding Style & Tools
- **Дополнительные возможности  
ReSharper**



# Поиск и навигация

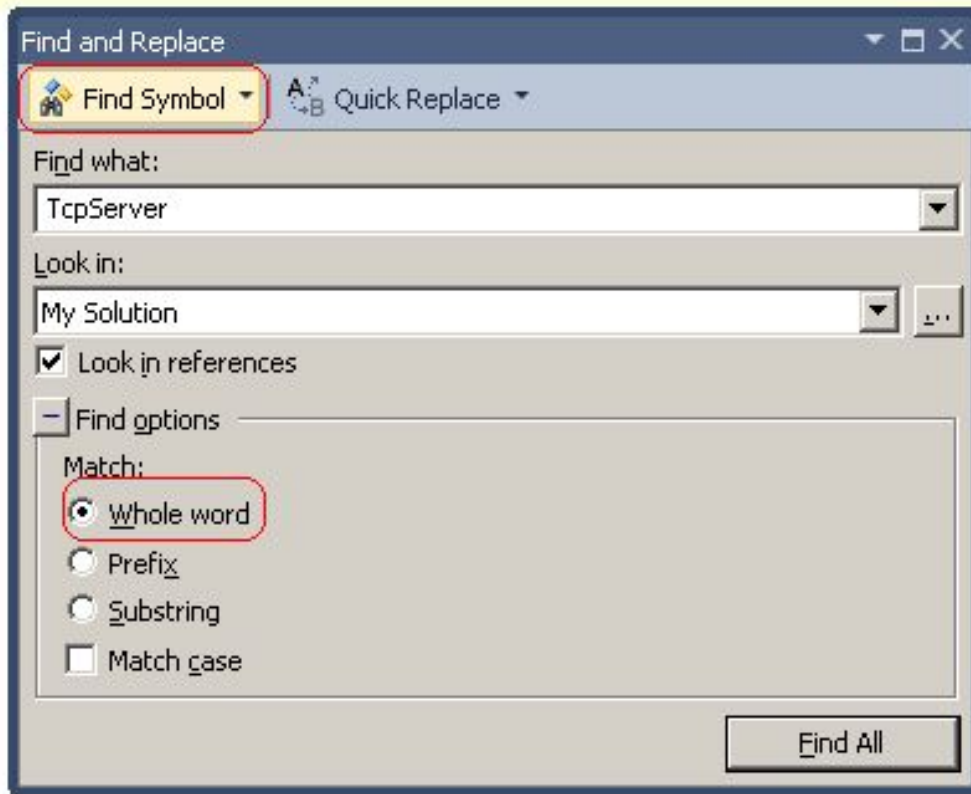
- **Встроенные возможности Visual Studio**
- Поиск файлов и типов в ReSharper
- Навигация в ReSharper



# [VS] Поиск и навигация

- Find Symbol в Find in Files (Ctrl + Shift + F)
- Navigate To (Ctrl + ,)
- Go To Definition (F12)
- Find All References (Shift + F12)
- Navigate backward/forward (!!) (Ctrl + “-”/Ctrl + Shift + “-”)

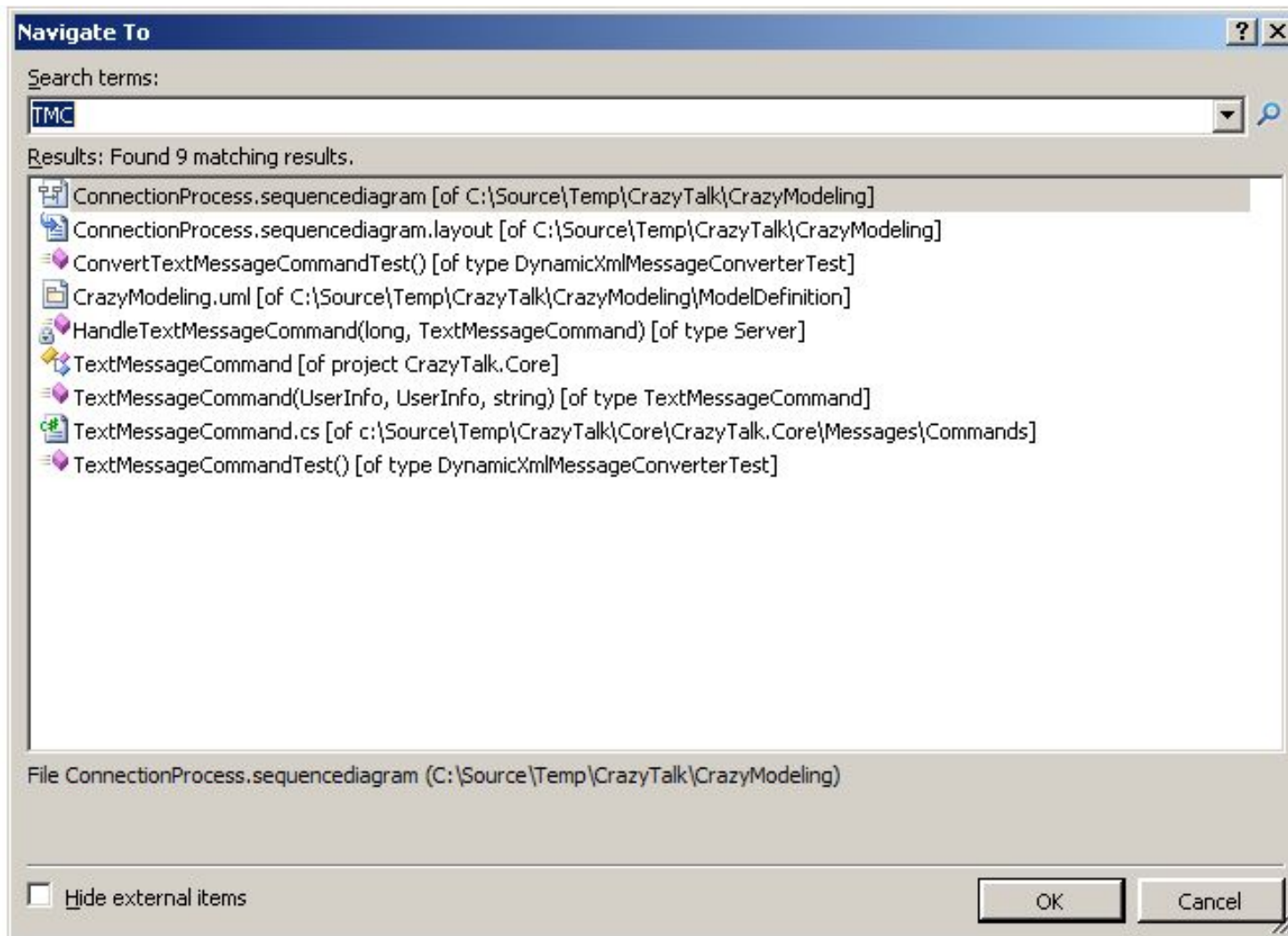
# [VS] Find Symbol vs Find in Files



# [VS] Navigate To

- Возможности
  - Одновременный поиск типов/полей/файлов
  - Поиск подстроки
  - Fuzzy Search (поиск по заглавным буквам)
    - Вбиваем TMC, найдем `TextMessageCommand`
- Недостатки
  - Универсальность (ищет все)
  - Высокий уровень «шумов»
- Подробнее: [Scott Guthrie “Searching and Navigating Code in VS 2010”](#)

# [VS] Navigate To





# Поиск и навигация

- Встроенные возможности Visual Studio
- **Поиск файлов и типов в ReSharper**
- Навигация в ReSharper

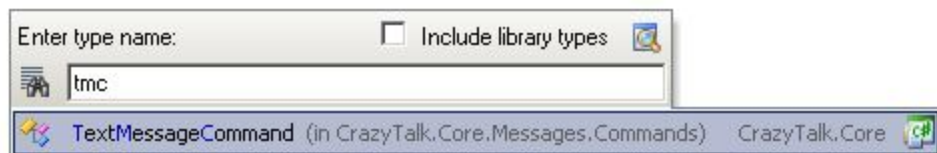


# [R#] Поиск файлов и типов

- Go to Type (Ctrl + T)
- Go to File (Ctrl + Shift + T)
- Go to File Member (Alt + \)
- Go to Symbol (Shift + Alt + T)

# [R#] Go To XXX примеры

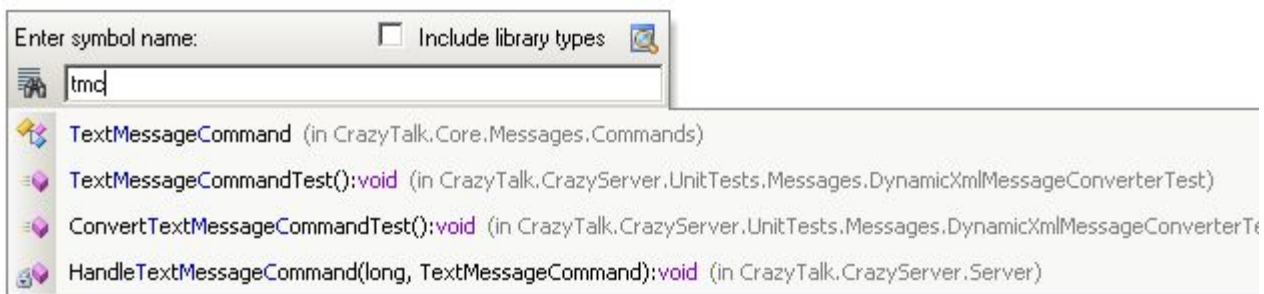
- Go To Type



- Go To File

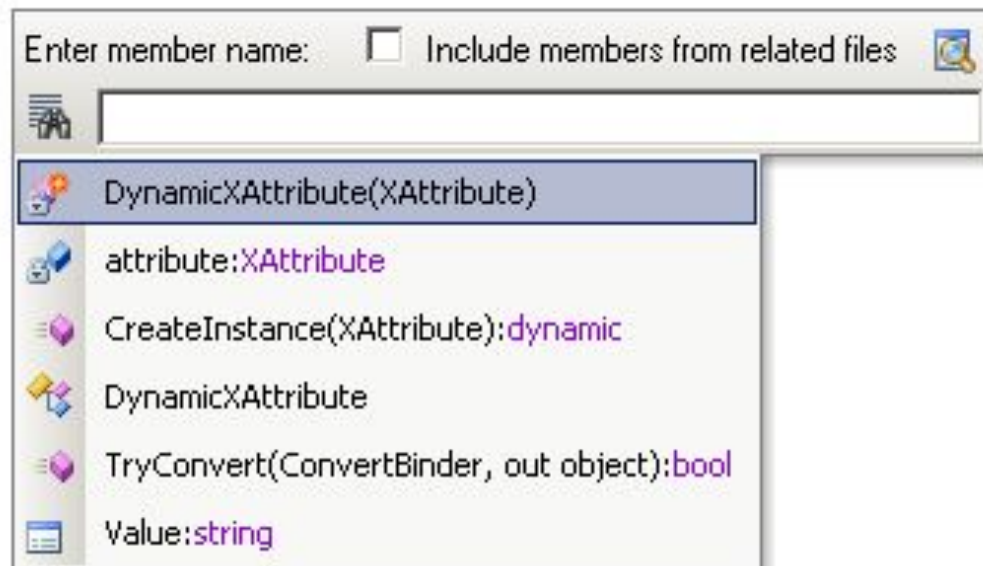


- Go To Symbol



# [R#] Go To File Member

- Универсальный способ доступа к конструктору:
  - **Alt + \** затем **Enter**



# [R#] Go To XXX

- Возможности
  - Возможность переключения из одного режима поиска в другой
  - Fuzzy Search
- Преимущества
  - Fuzzy Search работает не только с заглавными буквами
  - Специализация (как следствие, уменьшение «шумов»)

# Поиск и навигация

- Встроенные возможности Visual Studio
- Поиск файлов и типов в ReSharper
- **Навигация в ReSharper**



# [R#] Навигация

- Recent Files (Ctrl + “,”)
- Navigate To (Alt + `)
- Go to declaration (F12)
- Go to Implementation (Ctrl + F12)
- Go to Related Files (Ctrl + Alt + F7)
- Locate File in Solution Explorer (Alt + Shift + L)
- Go to next/previous member (Alt + Down/Up)

# [R#] Recent Files

- Список последних открытых файлов
- Поддерживается поиск и фильтрация





# [R#] Navigate To (Alt + `)

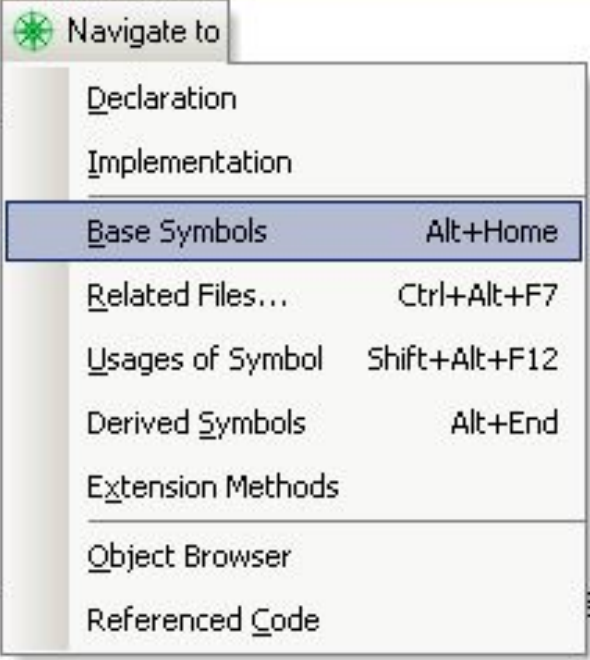
- Одна горячая для всех нужд

```
private Message CreateMessage(Command command)
{
    Contract.Requires(command != null);
    Contract.Ensures(Contract.Result<Message>() != null);

    var message = new Message(1, Interlocked.Increment(ref messageCount));
    return message;
}

private void SendMessage(Message message)
{
    Contract.Requires(message != null);

    string xml = messageConverter.Convert(message);
    connection.SendString(xml);
}
```



Option	Shortcut
Navigate to	
Declaration	
Implementation	
<b>Base Symbols</b>	<b>Alt+Home</b>
Related Files...	Ctrl+Alt+F7
Usages of Symbol	Shift+Alt+F12
Derived Symbols	Alt+End
Extension Methods	
Object Browser	
Referenced Code	

# [R#] Go to Implementation

- Борьба со «слабосвязной» архитектурой





```
/// <summary>  
/// Abstract interface for converting message from xml data.  
/// </summary>
```

```
[ContractClass(typeof(MessageConverterContracts))]
```

```
public interface IMessageConverter
```

```
{  
    Message Converter  
    XElement Conve  
}
```

Implementations of 'IMessageConverter'

 <b>DynamicXmlMessageConverter</b> (in CrazyTalk.CrazyServer.Messages)	CrazyTalk.CrazyServer	
 <b>MessageConverterContracts</b> (in CrazyTalk.Core.Messages)	CrazyTalk.Core	

# Demo

- Locate File in Solution Explorer (Alt + Shift + L)
- Go to next/previous member (Alt + Down/Up)

# Unit Testing with ReSharper

# Agenda

- А зачем мне это?
- Базовые возможности & Редактирование
- Поиск & Навигация
- Unit testing with Resharper
- **Refactorings & Code Generation**
- Coding Style & Tools
- **Дополнительные возможности  
ReSharper**



# Refactorings

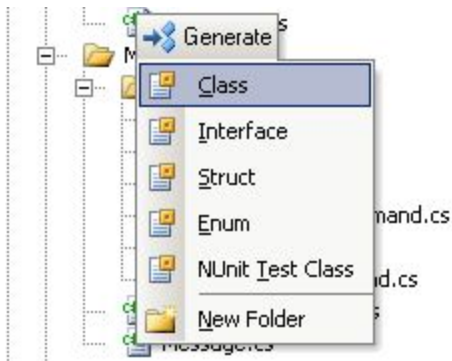
- Rename
- Change Signature
- Extract Method
- Adjust Namespaces
- Extract Class from Parameters
- Move Type to Another File or Namespace
- Move Types into Matching Files

# [R#] Code Generation

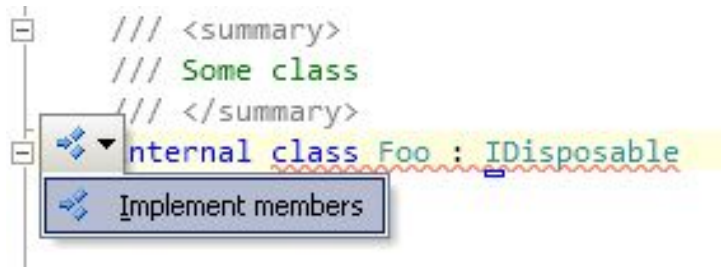
- Generate Files (Alt + Ins в Solution Explorer)
- Create from Usage (OK для TDD)
- Implement Interface (Alt + Enter на объявлении класса)
- Generate Type Members
  - Generate Constructor
  - Generate Missing Members
  - Generate Overriding Members
  - Equality Members

# [R#] Generate Files & Implement Interface

- Generate Files (Alt + Ins B Solution Explorer)



- Implement Interface





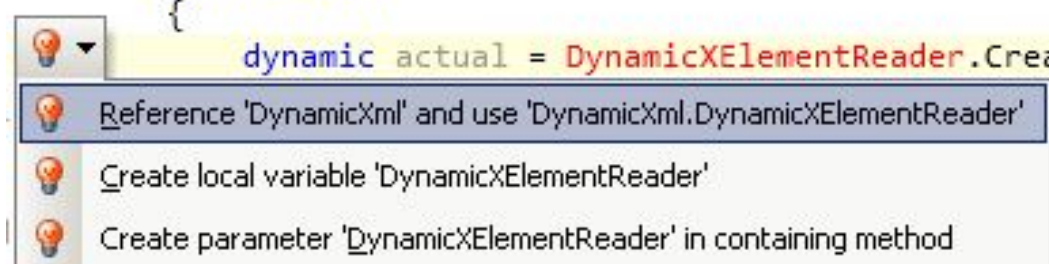
# [R#] Quick Fixes (Alt + Enter)

- Remove Unused Usings
- Move Type to Separate File
- Автоматическое добавление сборок

# [R#] Автоматическое добавление сборок

- Автоматически могут добавляться сборки:
  - из текущего проекта
  - некоторые известные сборки (например, System.Xml, System.Xml.Linq)

```
[TestCase(ExpectedExceptionName = "System.Diagnostics.Contracts._  
public void CreateInstanceFailureTest()  
{  
    dynamic actual = DynamicXElementReader.CreateInstance(null);
```



# Agenda

- А зачем мне это?
- Базовые возможности & Редактирование
- Поиск & Навигация
- Refactorings & Code Generation
- Unit testing with Resharper
- Coding Style & Tools
- **Дополнительные возможности  
ReSharper**

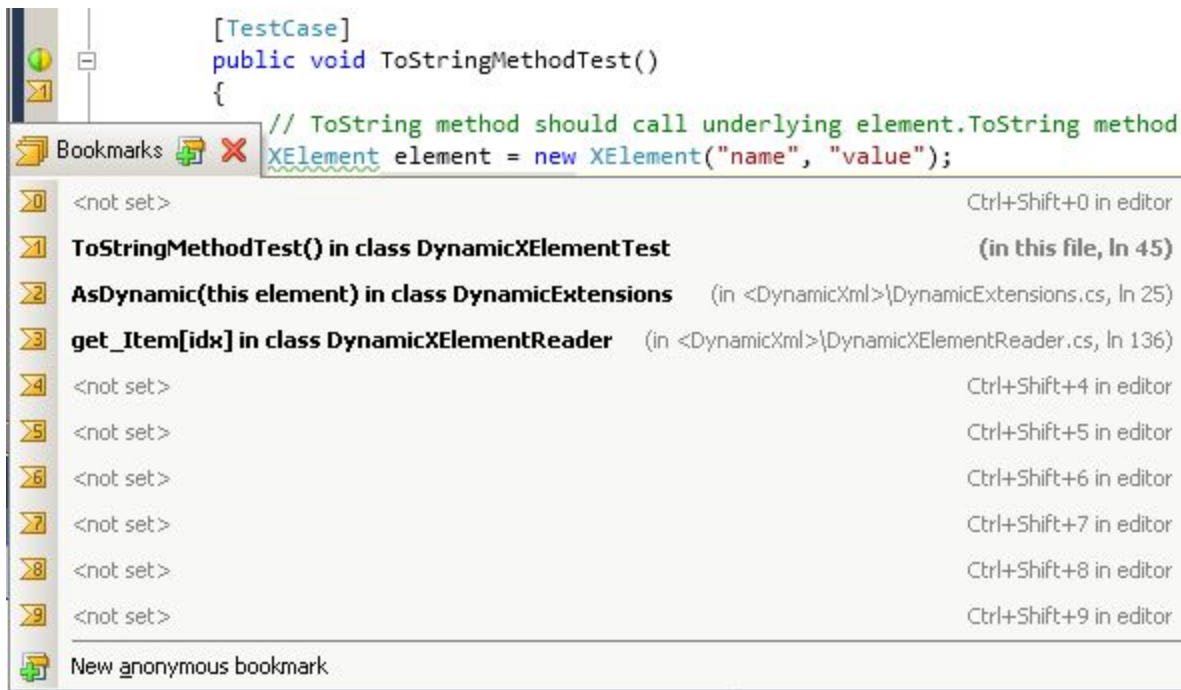


# Дополнительные возможности ReSharper

- Закладки (Bookmarks)
- Find Code Dependent on Module
- Stack Trace Explorer (Ctrl + E, T)
- To-do Items (Ctrl + Alt + D)

# [R#] Закладки

- View bookmarks (Ctrl + `)
- Go to bookmark (Ctrl + num)
- Set/remove bookmark (Ctrl + Shift + num)



# [R#] Find Code Dependent on Module

- Solution Explorer -> References -> System.Xml.Linq -> Find Code Dependent on Module

The screenshot shows the 'Find Results' window in Visual Studio. The title bar reads 'Find Results - Code of DynamicXml.UnitTests Dependent On Referenced Modules'. The main area is titled 'Code of DynamicXml.UnitTests Dependent On Referenced Modules'. Below the title bar is a toolbar with various icons and a 'Group by:' dropdown menu set to 'Modules, Namespaces and Types'. The 'Search target' section shows 'System.Xml.Linq in project <DynamicXml.UnitTests>'. The 'Found 140 usages in System.Xml.Linq referenced module' section is expanded to show a tree view of the module's contents:

- System.Xml.Linq (62 items)
  - System.Xml.Linq (62 items)
    - XAttribute (7 items)
    - XContainer (6 items)
    - XElement (48 items)
    - XNode (1 item)

# [R#] Out of scope

- Templates
- Code analysis
- Search With Patterns
- многое другое...

# Дополнительные материалы

## 1. [Coding Faster: Getting More Productive with Microsoft Visual Studio by Zain Naboulsi, Sara Ford](#)

Отличная и, видимо, лучшая книга о разных трюках в Visual Studio. Главным достоинством, как и главным недостатком, является ее объем (700+ страниц) и потенциально большой объем «шума» на единицу полезной информации. В любом случае Must See.

## 2. [Channel 9. Visual Studio ToolBox](#)

Набор видео материалов по разным «фишкам» как стандартной поставки Visual Studio, так и по некоторым сторонним «дополнениям», типа ReSharper, Code Rush, Productivity Power Tools и т.п.



# Дополнительные материалы

## 3. [Лучшие посты Скота Гаттри \(Лучшие посты Скота Гаттри \(Scott Guttrie\) за 2010 год\)](#)

Содержит весьма достойный набор ссылок, большая часть которых посвящена новым возможностям Visual Studio 2010

## 4. [ReSharper Features](#)

Да, я знаю, что никто не любит читать официальную 😊  
Но иногда в ней можно найти много чего полезного!

## 5. [ReSharper Default Keymap](#)

- [Visual Studio scheme pdf](#)
- [ReSharper 2.x / IDEA scheme pdf](#)

Распечатать в 3-х экземплярах и повесить перед рабочим столом!

Вопросы?



# Вертикальное редактирование

