

VS2010はここがすごい

中 博俊



わんくま
同盟

わんくま同盟 名古屋勉強会 #08

- C# 4.0
- VB 10.0
- 文字列の処理などが変わる
- F#搭載
- M搭載?
- C++0x一部搭載(ちょっとしょぼめ)
- Contract 契約プログラミング
- Memory Mapped Files
- SortedSet<T>

C#10 & VB9

- **Dynamic** DLRサポート, Office開発支援
- **Task & Parallel** 並行処理開発支援

```
foreach(var y in ls) { Task t =  
    Task.Factory.StartNew(x=>Console.WriteLine(x), y); }  
(from x in Enumerable.Range(1, 100).AsParallel()  
 select x).ForAll(i => Console.WriteLine(i));  
Parallel.For(1,30,i =>Console.WriteLine(i));  
using System.Threading.Tasks;  
using System.Threading;  
using System.Linq.Parallel;
```

- **BigInteger**



C#10 & VB9

- **BigInteger** 何桁でも行ける整数

```
var bi = new System.Numerics.BigInteger();
```



静的メンバー	
パブリックでないメンバー	
_Bits	{uint[2]}
_bits	{uint[2]}
[0]	2808348671
[1]	232830643
Sign	1

静的メンバー	
パブリックでないメンバー	
bits	{uint[15]}
bits	{uint[15]}
1	1
sign	1

- F#でもおなじみ Tuple

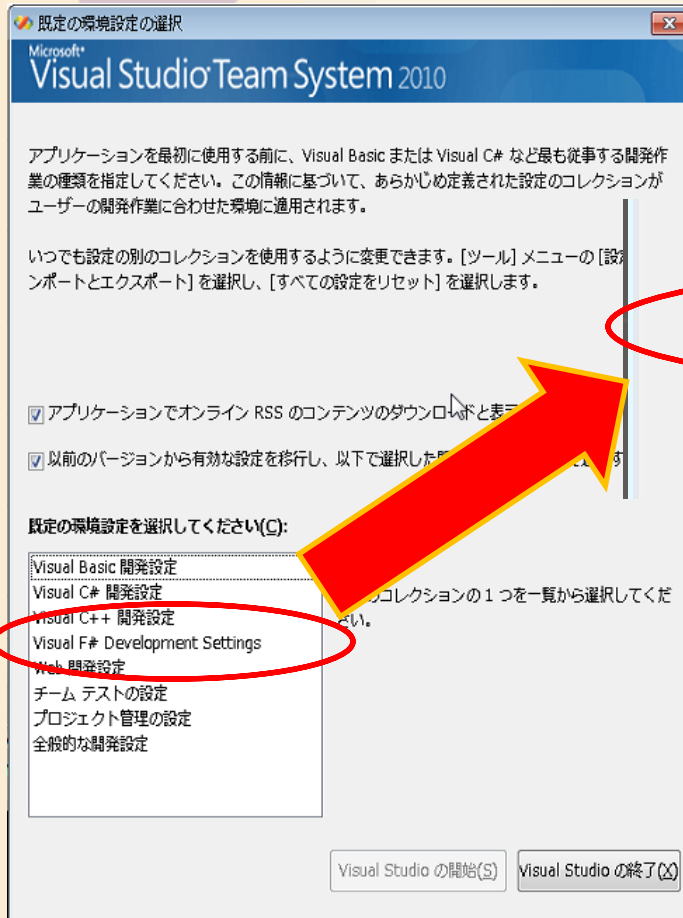
```
var t = new Tuple<bool, int>(true, 10);
```

```
if (t.Item1 == true) Console.WriteLine(t.Item2);
```

文字列の処理が変わる

- 愛々問題の解決
- `var x = "愛々,123";`
- `Console.WriteLine(x.Substring(x.IndexOf(",")));`
- CLR2.0(.NET 2.0~3.5)
- CLR4.0(.NET 4.0)
- 挙動の変更
 - <http://msdn.microsoft.com/ja-jp/netframework/dd890508.aspx>

F#



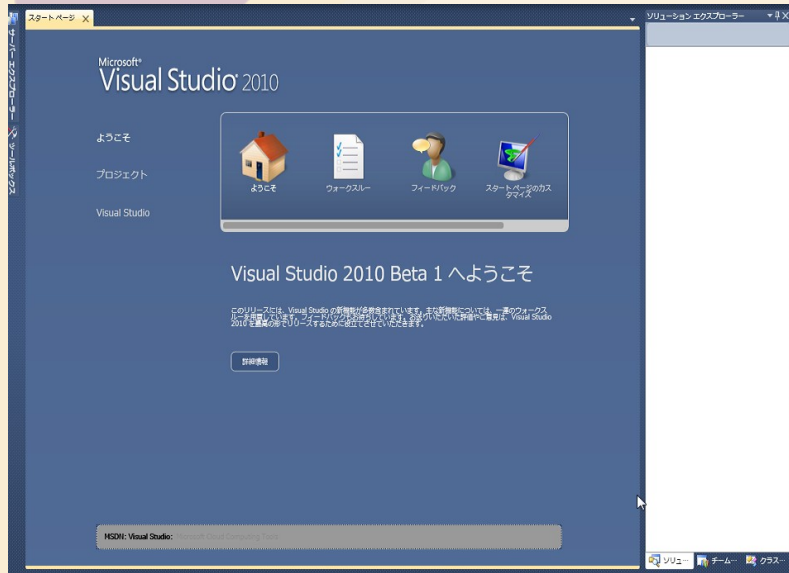
Visual C++ 開発設定

Visual F# Development Settings

Web 開発設定

IDEなど

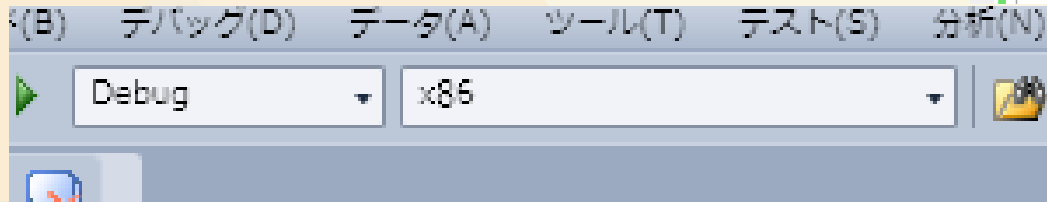
画面がWPF



Ctrl+ホイールで拡大が可能

```
Program.cs | スタートページ | オブジェクトブラウザー
@_ConsoleApplication1.Program | @Main(string[] args)
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Threading;
using System.Linq.Parallel;

namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            Parallel.For(1, 30, i => Console.WriteLine(i));
        }
    }
}
```



デフォルトプロジェクトがx86



Contract

- 事前(コンパイル時)に検証可能なことを検証してくれる。

```
using System.Diagnostics.Contracts;
```

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

- 契約プログラミング
- 事前
- `Contract.Requires(param != null);`
- 事後
- `Contract.Ensures(Contract.Result<string>() != null);`



UI Test

コード化された UI テストの作成オプション

コード化された UI テストの作成方法を選択します

テスト ケースまたは共有ステップに関連付けられた操作の記録を使用(I)
テスト ケースまたは共有ステップに関連付けられた操作の記録と同じ操作
を実行するコードを生成します。

レコーダーの使用(R)
レコーダーでは、アプリケーションをステップ スルーしながら操作をキャ
プチャして、これらの操作のコードを生成できます

UI コントロール ロケーターの使用(U)
UI コントロール ロケーターを使用して、アプリケーションの UI コント
ロールを選択し、各コントロールのコードを生成してから、各コントロー
ルの値を確認します

コードを生成しない(G)

コード化された UI テスト ビルダー - メソッドの記録

一時停止(P) | リセット(R) | 直前の操作を削除(D)

クリック '不明な名前' ペイン
'不明な名前' テキスト ボックス に 'あいうえお' を入力
'不明な名前' テキスト ボックス に '{Enter}' を入力
不明な名前 テキスト ボックス に 'かきくけお' を入力

メソッド名 RecordedMethod1 | メソッドの生成(G) | キャンセル(C)

記録中です。

```
// '不明な名前' テキスト ボックス に 'AA' を入力
WpfWindow wpfWindow = new WpfWindow();
#region 検索条件
wpfWindow.SearchProperties.AddRange(new PropertyExpression("N
#endregion
WpfEdit itemEdit = new WpfEdit(wpfWindow);
itemEdit.Text = "AA";

// 'Wpf' の 2 番目の テキスト ボックス ウィンドウ に 'BB' を入力
WpfEdit itemEdit1 = new WpfEdit(wpfWindow);
#region 検索条件
itemEdit1.SearchProperties.Add("Instance", "2");
#endregion
itemEdit1.Text = "BB";

// クリック '不明な名前' ボタン
WpfButton itemButton = new WpfButton(wpfWindow);
```



- <http://www.microsoft.com/downloads/details.aspx?displaylang=ja&FamilyID=85520793-68fc-4361-a8b6-dc2cff49c8d2>
- みなさんもVisual Studio 2010を評価しましょう！