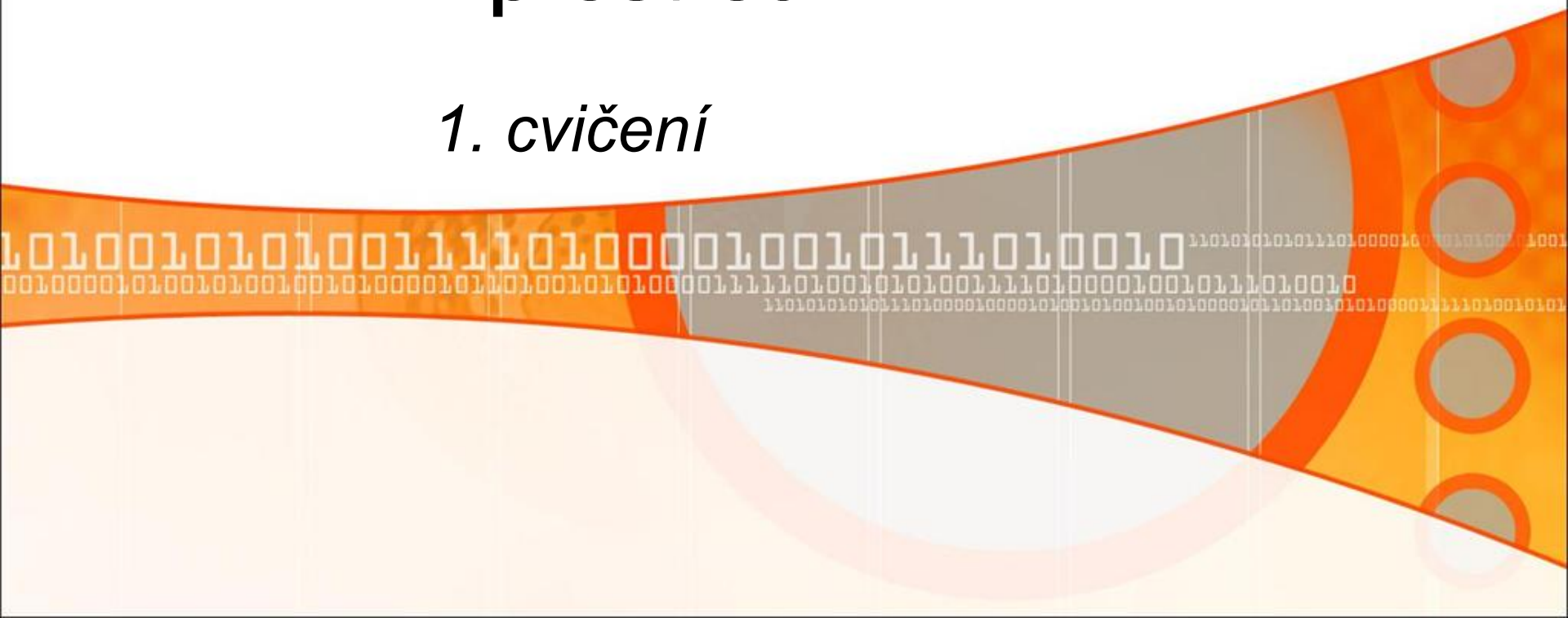
Vývoj aplikací v prostředí .NET

1. cvičení



Organizace cvičení

semestrální práce	40
aktivita v hodině	20
celkem za semestr	**60 bodů**

Na zápočet je třeba získat minimálně 18 bodů.

Aktivita v hodině

- lze získat 0-4 body
- 1 bod za splnění úkolu (odevzdaného nejpozději příští hodinu)
- 1-3 body za dokončení úkolu ještě v hodině

semestrální práce

- není povinná
- téma je nutné si zvolit do konce 4. týdne

BODY	KATEGORIE	POPIS
0 – 10	Obtížnost	*zadána nebo konzultována předem*
0 – 10	Provedení	*vhodné komponenty, algoritmy…*
0 – 10	Srozumitelnost	*členění aplikace, komentáře*
0 – 5	Použitelnost	*uživatelské rozhraní*
-15 – 5	Stabilita	*ošetření vstupů, výjimky…*
-40 - 0	Znalost kódu	–

- vždy bude hodnocen pouze vlastní přínos k aplikaci (plagiátorství)
- zadání budou na webu předmětu do 3. týdne
- téma je nutné si zvolit do konce 4. týdne

MSDN Academic Alliance

Popis: http://dce.felk.cvut.cz/msdnaa/

Doporučené produkty pro Y35VAN

- Visual Studio 2010 Ultimate (x86 and x64 WoW) - DVD
- případně i Expression Studio 4

Doporučená instalace IIS

Všichni studenti zapsaní na předmět Y35VAN byli již zaregistrovaní a měli by dostat email. Může se však stát, že zprávy nedojdou, zpravidla pro špatné přesměrování účtů na fel. V těchto případech napište na pjr@susta.cz. Mezitím si můžete půjčit DVD od spolužáků

přehled 1. cvičení

1. Java versus C#
2. Aplikace typu console
3. Visual Studio (prostředí, tvorba projektu, help)
4. Vstup a výstup uživatele (konzolová aplikace)
5. Klávesové zkratky
6. Domácí úkol

Java versus C#

Jednoduché změny, *pouhé změny formy zápisu*

```
String s1 = "hello";
String s2 = s1;
String path = "C:\\My Documents\\";
```

```
string s1 = "hello";
string s2 = s1;
string path = "C:\\My Documents\\";
// ale také výhodněji jako verbatim
string path = @"C:\My Documents\";
```

Java versus C#

Složitější změny, *aneb existuje vzdálená podobnost*

```
public static void main(String[] args)
  { for (int i = 0; i < args.length; i++)
      System.out.println( args[i] );
   }
```

```
public static void Main(string[] args)
{ for (int i = 0; i < args.Length; i++)
      System.Console.WriteLine( args[i] );
 }

// v C# též
   foreach (string s in args)
      System.Console.WriteLine(s);
```

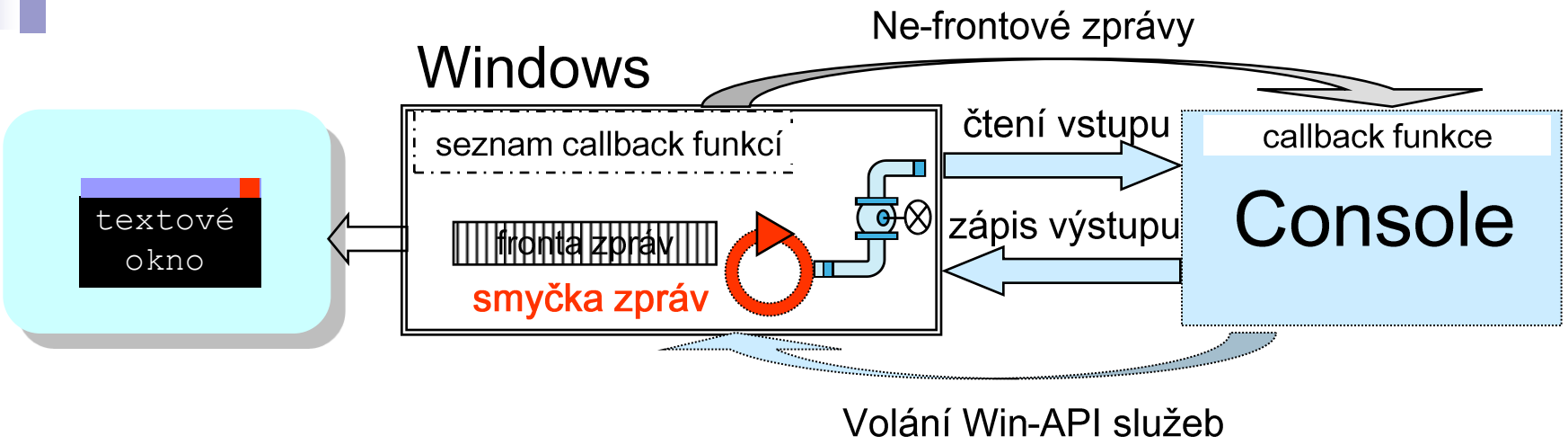
Java versus C#

Náročné změny, *aneb vše skoro jinak*

```
try  { BufferedReader infile = new BufferedReader(new FileReader("text.txt") );
       String str;
       while ((str = in.readLine()) != null)   { System.out.println(str);  }
       infile.close();
     }
      catch (IOException e)    { }
```

```
using (System.IO.StreamReader sr = System.IO.File.OpenText("text.txt"))
    {  string str = "";
       while ((str = sr.ReadLine()) != null)
         { System.Console.WriteLine(str); }
    }
       // i v C# lze také použít try ... catch(,,,), ale using snažší
```

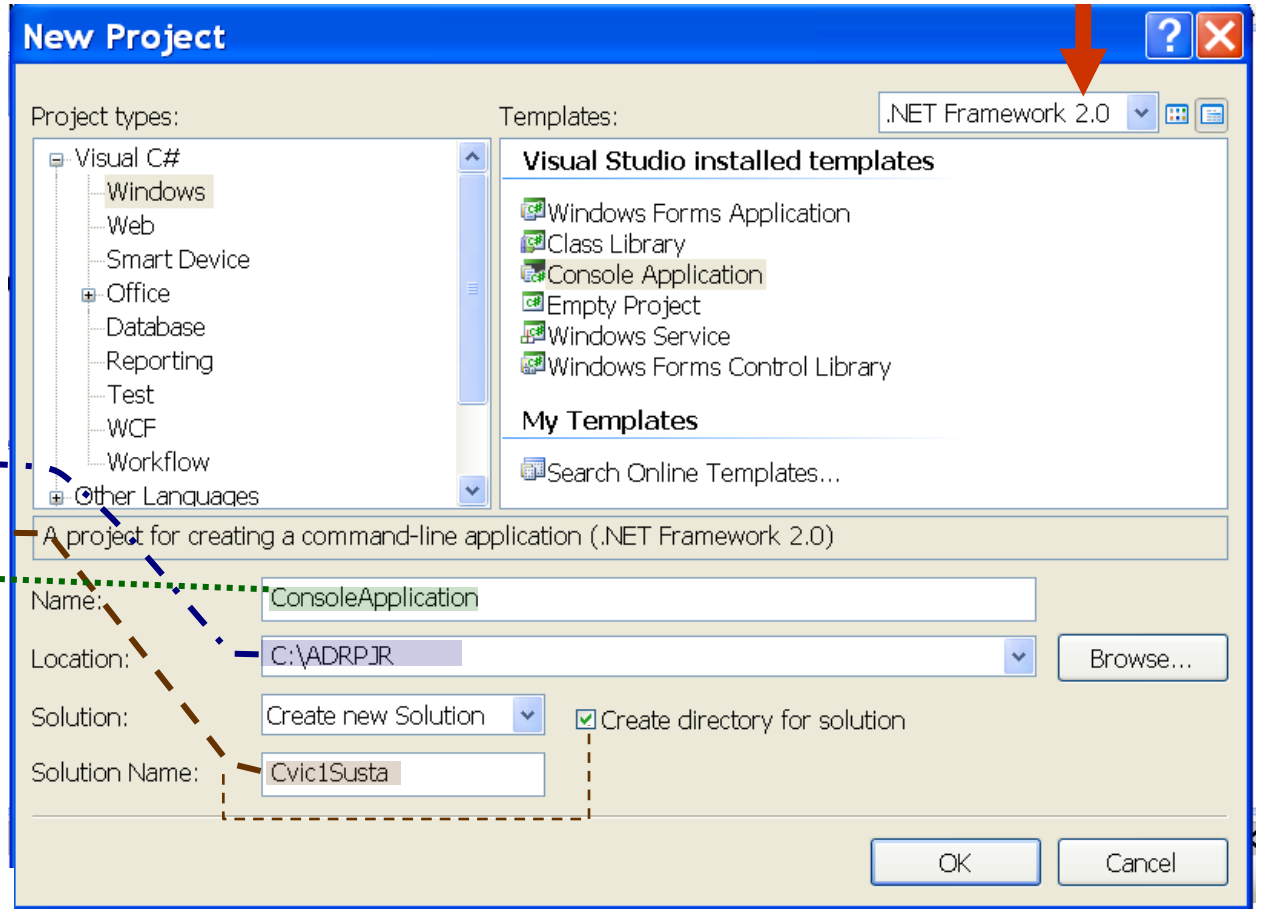
Opak. z předn.: Aplikace typu Console



- OS zpracovává smyčku zpráv, emuluje pro aplikaci textové okno spolu se vstupním a výstupním proudem (*pomocí Win API služby pipe*)
- Console sama nepřijímá frontové zprávy, tedy např. od zprávy myši nebo pokyny k překreslení okna, a proto nemůže vytvářet okna.
- Console smí využívat všechny ostatní služby OS a přijímat nefrontové zprávy, pokud se na ně zaregistruje. Může mít i TIMER, protože ten existuje ve dvou variantách: jako frontová zpráva WM_TIMER, ale i jako nefrontová s callback. (*Více o službách OS bude později na přednáškách.*)

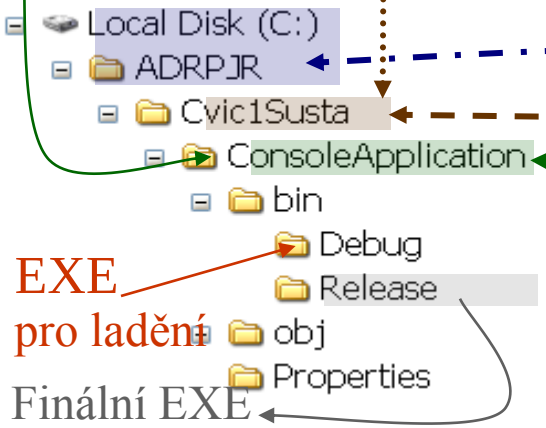
Console ve Visual Studio 2010

- File → New →Project...



Solution
(skupina programů)

Zdrojový kód



EXE pro ladění

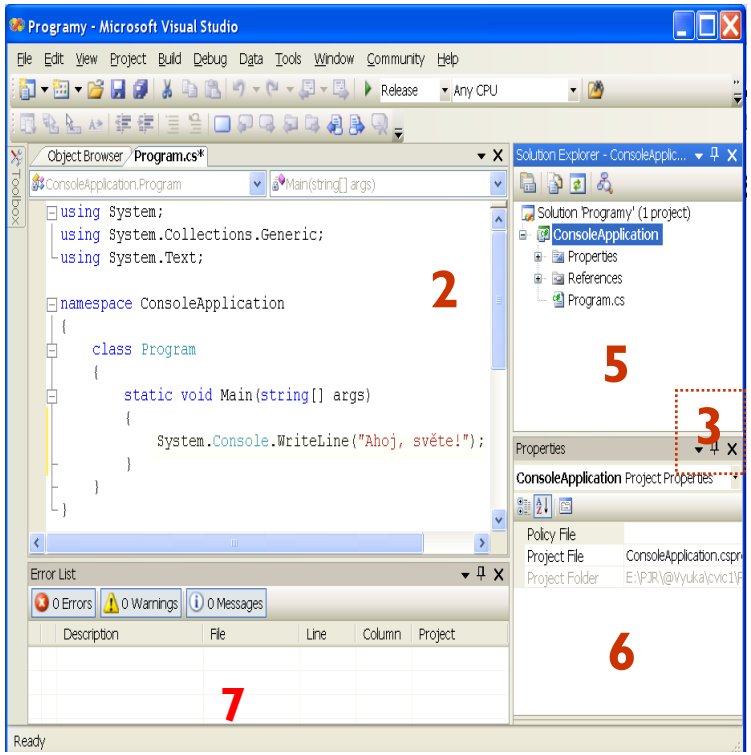
Finální EXE

– *adresář* Release *se vytvoří až při prvním „release“ překladu*

Základní vývojové okno

1. Menu a nástrojové lišty
2. Textový editor
3. Pozicování okna ▼
   špenlík - *auto-hide function*
   x zavření okna (opětovné otevření se provede z menu)

4. Toolbox – odšpendlené okno, podle typu aplikace obsahuje seznam ovládacích prvků, v Console nic neobsahuje
5. Solution Explorer – seznam programů v rámci řešení (Solution)
6. Properties - kontextově zobrazované vlastnosti jednotlivých částí zdrojového kódu
7. Error List - poznámky k řešení



Vstup a výstup uživatele

- Třída **Console**
- *Console.Write(),*
- *Console.WriteLine()*
- *Console.ReadLine()*
- *Console.ReadKey(), Console.ReadKey(true/false)*

Srovnání formátování C a C#

C++

```
int i=7; double r=3.14;
printf(
"Výsledek \t%4d * %3g = %4g (%e)\n-ok-",i,r,i*r,i*r);
```

C#

```
int i = 7; double r = 3.14;
Console.Write(
"Výsledek \t{0,4:d} * {1,3:g} = {2,4:g} ({2:e})\n-ok-", i,r,i*r);
```

Oba kódy dávají totožný výstup

```
Výsledek          7 * 3,14 = 21,98 (2,198000e+001)
-ok-
```

- *Formátovací skupiny* % aligment typ *nahrazeny v C# přehlednějším zápisem* {index, aligment : typ}, *který dovoluje více příkazů*
- *Běžné kódy typů formátu se shodují, např.* d, g, e, x
- *Stejné jsou i* " \t \r \n (VS2008 Help: escape characters)
- *V C# lze ale vícekrát odkazovat na tentýž parametr*

Kód pro Ahoj, světe

```
using System;
using System.Collections.Generic;
using System.Text;
```

Vkládané knihovny, analogie direktiv
C++ #include či Java import

```
namespace ConsoleApplication
{
  class Program
  {
    static void Main(string[] args)
    {
      Console.WriteLine("Ahoj, světe!");
    }
  }
}
```

```
// Java kód pro srovnání (v NetBean 5.0)
package javaconsole;

public class Main
{
 public static void main(String[] args)
    {
       System.out.println("Ahoj, světe!");
    }
}
```

Pozn. Zavření okna zabráníme třeba použitím break-pointu za WriteLine(),
nebo Console.ReadKey(true); // čti znak bez výpisu na displej

Komentáře

// jednořádkový komentář

/* víceřádkový komentář */

/// dokumentační komentář

Direktivy using zjednodušují zápis

```
// using System;
// using System.Collections.Generic;
// using System.Text;

namespace ConsoleApplication
{
    class Program
    {
        static void Main(string[ ] args)
        {
            System.Console.WriteLine("Ahoj, světe!");
        }
    }
}
```

Při vynechání using musíme vždy uvádět celá jména včetně namespace.

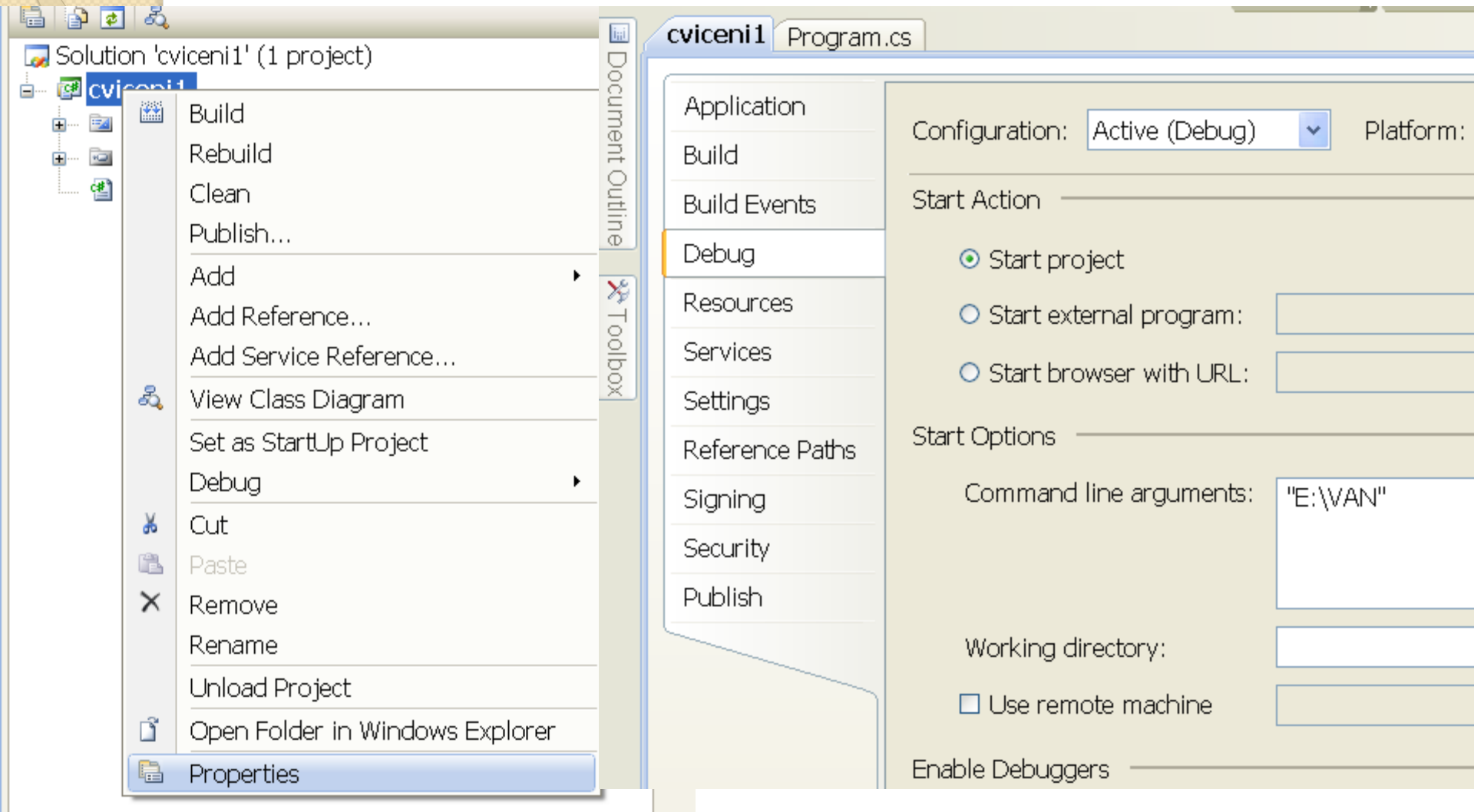
Pozn. Náš program využíval zatím jenom namespace System;

Pozor: Nelze však vložit *using System.Console*; a psát jen WriteLine(), protože using specifikuje pouze namespace. Console určuje jméno třídy v namespace System.

Výpis adresáře

- Použití: prog.exe directory
- Odezva: *výpis souborů v adresáři a všech podadresářích*

Nápověda: řádkové parametry



Příklad: výpis adresáře

```
class Program
{ static int cntr=0;
  static void Main(string[ ] args)
  {
    if (args.Length > 0)  ListDir(args[0]);
    else Console.WriteLine("prog.exe directory");
    Console.WriteLine(); Console.WriteLine("*** Done***");
    Console.ReadKey(true);
   }
   static void ListDir(string directory)
   {
      foreach (string file in Directory.GetFiles(directory))
           Console.WriteLine("{0}.\t{1}",cntr++,file);
      foreach (string dir in Directory.GetDirectories(directory))    ListDir(dir);
     }
 }
```

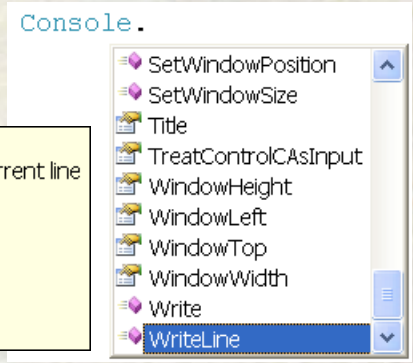
Vyzkoušejte si důležité klávesové zkratky

Kláv. zkratka	Popis	Explicitní volání
F1	*Nápověda k prvku*	
Ctrl+**F1, D**	*Ukaž okno dynamické nápovědy*	Help→Dynamic Help
Ctrl+**F**	*Hledej v textu*	Edit→FindReplace→Find
F3	*Další hledání v textu*	
Ctrl+Shift+**F**	*Hledej v souborech*	Edit→FindRepl.→Find in Files
F8	*Ukaž další řetězec (další úkol, chybu)*	*(PraváMyš→Next task)*
Shift+**F8**	*Ukaž předchozí řetězec (resp. úkol)*	*(PraváMyš→Prev. task)*
Ctrl+ **]**	*Jdi na párovou závorku či uvozovky*	
Ctrl+Shift+ **]**	*Označ text až k párovému ohraničení*	
Ctrl+ **E**, **F**	*Přeformátování vybraného textu (nebo také i Ctrl+K, Ctrl+F)*	Edit→Advanced→Format Selection
Ctrl+**E, D**	*Přeformátuj vše*	
Ctrl+ **B**, Ctrl+ **T**	*Vlož/zruš záložku*	Edit→Bookmarks→Toggle
Ctrl+ **B**, Ctrl+ **N**	*Další záložka*	Edit→Bookmarks→Next
Ctrl+ **B**, Ctrl+ **P**	*Předchozí záložka*	Edit→Bookmarks→Previous
F5	*Překlad a spuštění psané aplikace*	Debug→Start
F6	*Překlad všech projektů v řešení*	Build→Build Solution

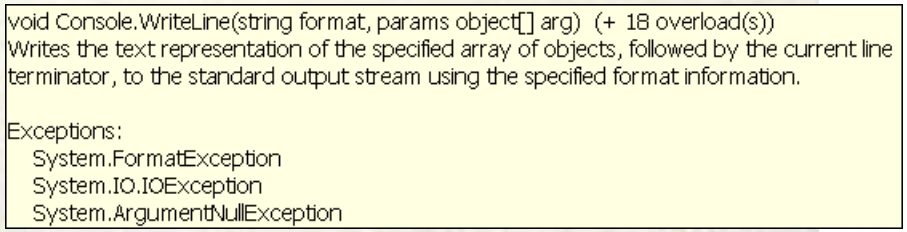
Vyzkoušejte IntelliSence technologie

Crtl+space
nebo
Ctrl+J

Automatické dokončování textu = Seznam dostupných členů dané třídy nebo instance spolu s nápovědou. Samo se vyvolává i po zapsání tečky za jménem třídy (resp. „namespace“ – o něm bude více na přednáškách)

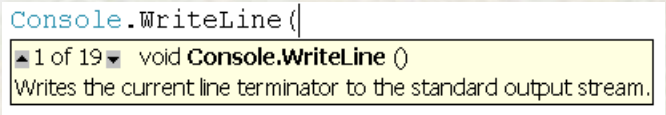


Nápověda se objeví také při najetí myší na text



Ctrl+Shift+space

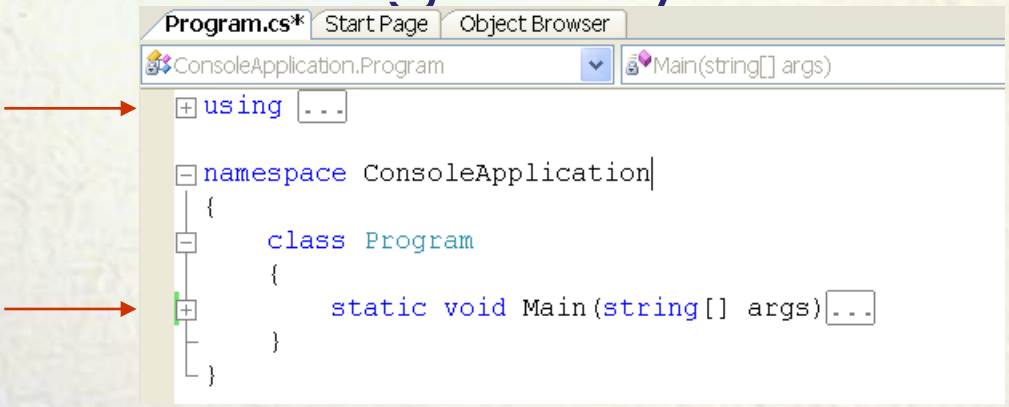
Seznam formálních parametrů dané metody. Automaticky se objeví i po zapsání otevírací závorky za jménem metody.



Edit → IntelliSense	→ Parameter Info	Ctrl+K, P
	→ Quick Info	Ctrl+K, Q
	→ List Members	Ctrl+K, L

Outlining

- Outlining = skrytí části textu



- Outlining se vytvoří i makropříkazy #region #endregion

```
        static void Main(string[] args)
        {
            #region prikazy
            Console.WriteLine("Ahoj, světe!");
            #endregion
        }
```

```
        static void Main(string[] args)
        {
            prikazy
        }
```

Příprava na další cvičení

Domácí úkol

Připravte *si třídu, fragment kódu umožňující* snadné zpracování vstupních argumentů konzolové aplikace.

Vstup – parametry příkazové řádky ve formě

--jmeno hodnota //(plné označení),

-j hodnota //(zkratka)

Příklad zadání progr.exe -s "C:\data.txt" -b 8 --log verbose

Požadovaná funkcionalita

- nezáleží na pořadí zadání parametrů,
- parametr může obsahovat hodnotu nebo být pouze přepínačem,
- jeden parametr může být označen za „výchozí“ (je možno uvést pouze hodnotu bez přepínače),
- při chybě zpracování parametrů je tato chyba oznámena uživateli a je vypsána nápověda pro použití.

Možná struktura a použití kódu

```
class Program
 { static void Main(string[] args)
   {    Parameters par = new Parameters(args);
        if(par.Empty) { /* write help */ }
        string jmenoSouboru = par.Read("--source");
        if(!String.IsNullOrEmpty(jmenoSouboru ))
             /*....*/
   }
 }
 class Parameters
 { /* tabulka prijimanych parametru */
   /* datove cleny pro nacteni parametru */
   public Parameters(string[] args)
   { /*...*/ }
   public string Read(string parName)
   { /*...*/ return "entered-string or default-string or null"; }
   public bool Empty()
   { /*...*/ return true /*false*/; }
 }
```