Računarstvo i informatika

III razred

dr Ana Kaplarević-Mališić
Institut za matematiku i informatiku
Prirodno-matematički fakultet
Univerzitet u Kragujevcu

Cilj

Praktično

- Naučiti osnove programskog jezika C#

Idejno

- Upoznati se sa drugačijim programskim paradigmama
- Razumeti osnovne koncepte objektno-orijetisanog programiranja
- Razumeti koncept programiranja vođenog događajima

Programske paradigme

- Programski jezici se međusobno razlikuju po načinu na koji modeliraju realne probleme (programskim paradigmama koje podržavaju).

Šta je šta

- **Programming paradigm**
  Obrazac koji služi kao doktrina/učenje koje se sledi u procesu programiranja
- **Programming technique**
  Strategija rešavanja problema koja se primenjuje u algoritmu.
  Primer: strategija 'podeli pa vladaj'
- **Programming style**
  Stril kojim je program napisan. (elegenacija ili nedostatak elegancije)
- **Programming culture**
  Sveukupan izraz jednog programera, koji je često usko povezan sa familijom programskih jezika – definišu je glavne paradigme, stilovi i programerske tehnike koje jedan programer koristi ili kojima vlada.

Podela paradigmi

Paradigma programiranja – osnovni način struktuiranja misli tokom programiranja

Jedna moguća podela: imperativno vs. deklarativno

Neke paradigme

- **(imperativno)** Strukturno, proceduralno programiranje – Pascal, Basic,C

  koncept – "prvo uradi ovo, pa ovo"

- **(deklarativno)** Paradigma funkcionalnog programiranja – program se sastoji iz definicija (matematički definisanih) funkcija, a računanje se svodi da evaluaciju vrednosti definisanih f-ja za zadate argumente (Lisp, Haskell)

  koncept – "izačunaj vrednost izraza"

- **(deklarativno)** Paradigma logičkog programiranja – (Prolog), zasnovano na aksiomama i pravilima zaključivanja,

  koncept – "odgovori na pitanje"

- **(imperativno)** Objekto-orjentisana paradigma

  koncept – "modeliranje fenomena realnih sistema definisanjem komunikacije između objekata tog sistema"

Primer n!

proceduralno **C**

```
int fakt(int n)
{
  if (n==1) return 1;
  return n*fakt(n-1);
}


…
  printf("%d",fakt(10));
```

funkcionalno **LISP**

```
(defun fakt(n)
  (cond
    ((= n 0) 1)
    (t (* n (fakt (- n 1))))
   )
)

…
fakt(10)
```

objektno-orjentisano **Java**

```
class Calc
{
   static int fakt()
   {
     if (n==1) return 1;
     return n*fakt(n-1);
   }
}

…
System.out.println(Calc.fakt(10));
```

logičko **PROLOG**

```
fakt(0,1).
fakt(X,Y):- U is X-1,fakt(U,Z),
            Y is X*Z.


…
fakt(10,120).
fakt(10,X).
```

Naš fokus

- Objektno-orijentisana paradigma
- C#
- Razvoj aplikacija sa grafičkim okruženjem
- Razvoj aplikacija za više uređaja
- XNA C# bibiloteka za razvoj igrica

C#

Hello, world!

```
namespace Pozdrav
{
    using System;

    class Pozdrav
    {
        public static void Main()
        {
            Console.WriteLine("Pozdrav!");
        }
    }
}
```

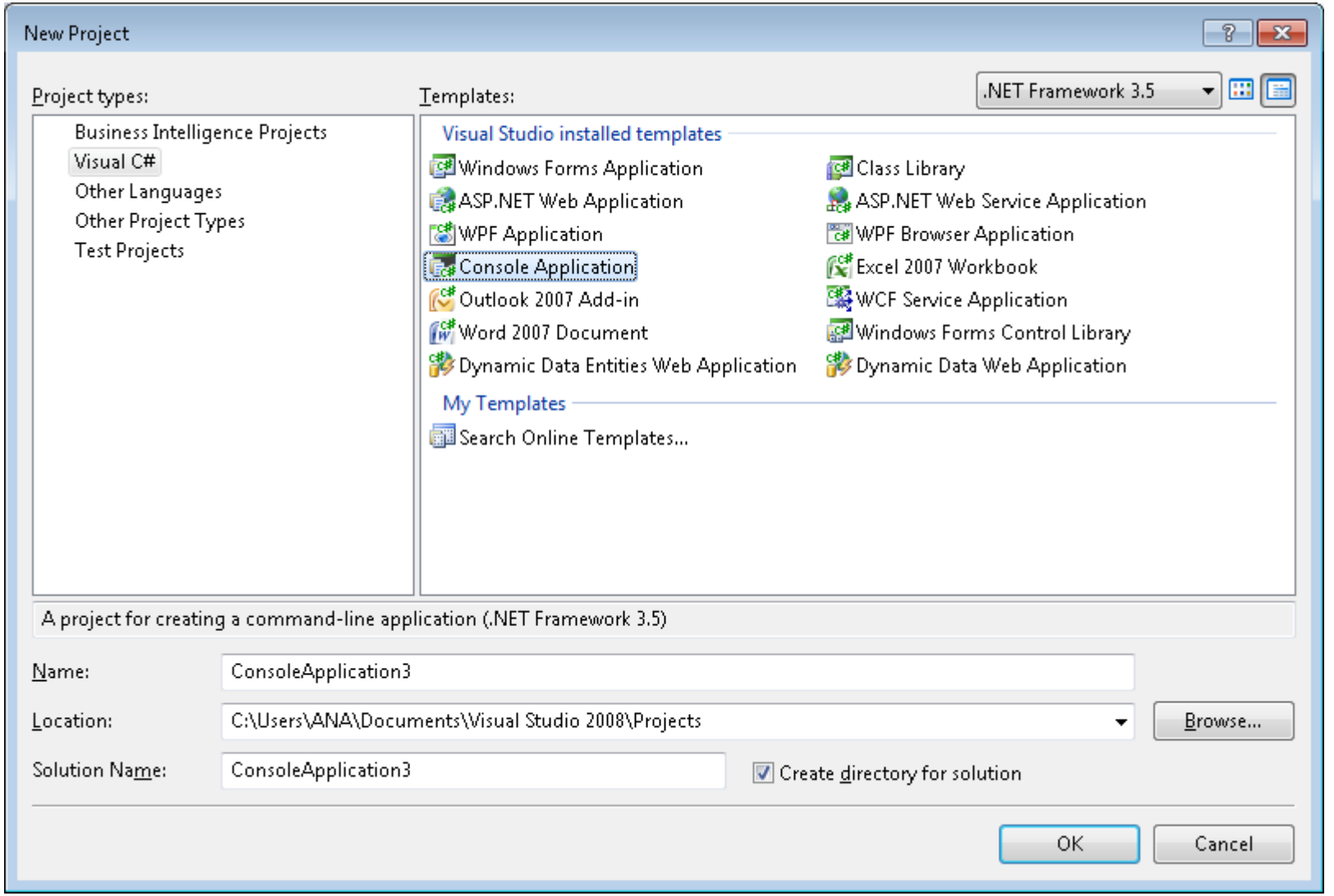
Veliko M je obavezno!

Komentari kao u C-u

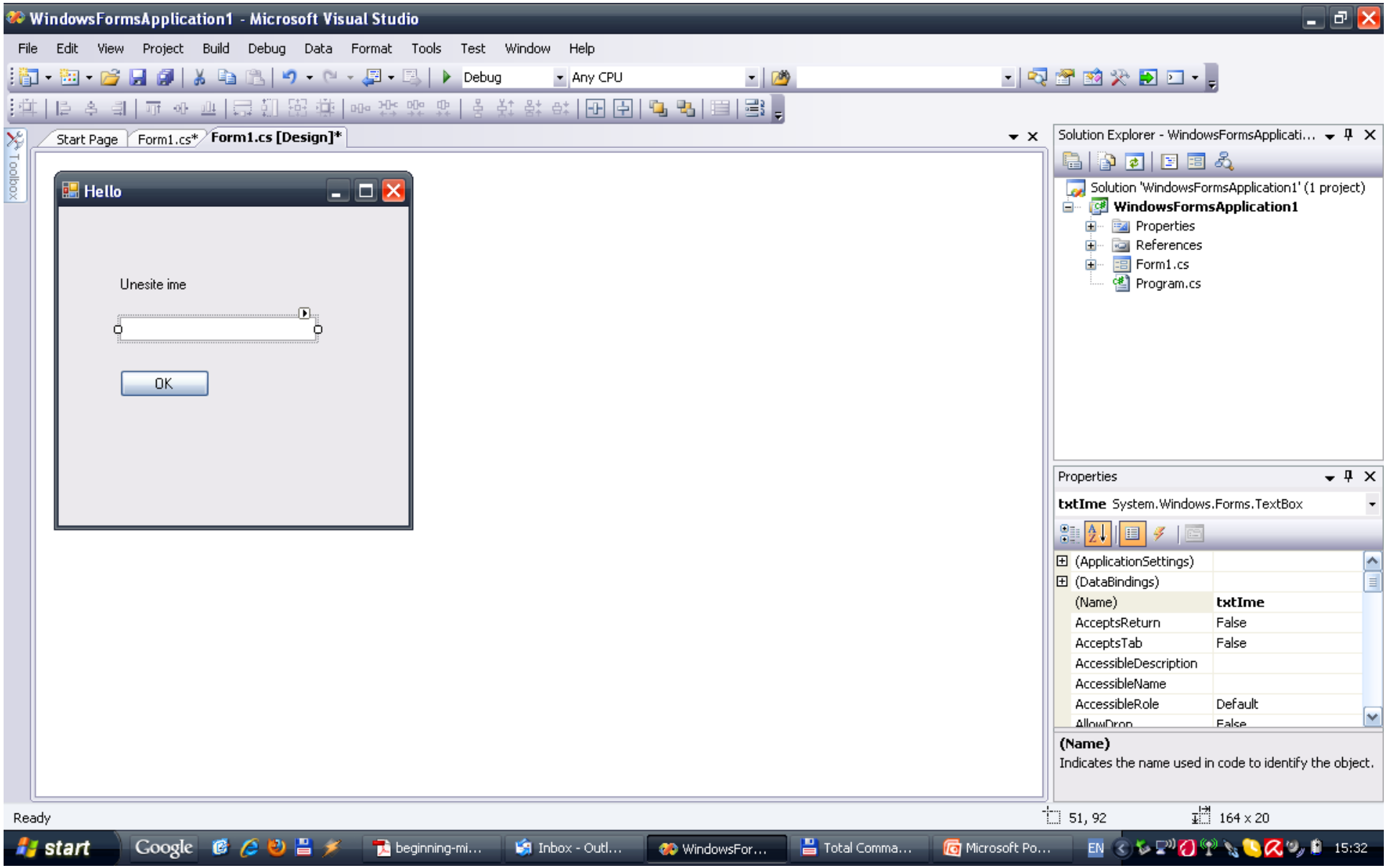
//	jedna linija
/* */	više linija

- **Main()** method je funkcija od koje kreće izvršavanje cele aplikacije.
- **Main()** i sve ostale funkcije moraju biti unutar class-e.
  - Međusobno povezane klase se organizuju u skupove koji se zovu namespaces.
  - Jedan namespace može sadržati druge namespace-ove.
  - System je namespace koji sadrži klase koje većina aplikacija koristi za interakciju sa operativnim sistemom.
  - VS 2010 po pravilu koristi naziv projekta kao top-level namespace

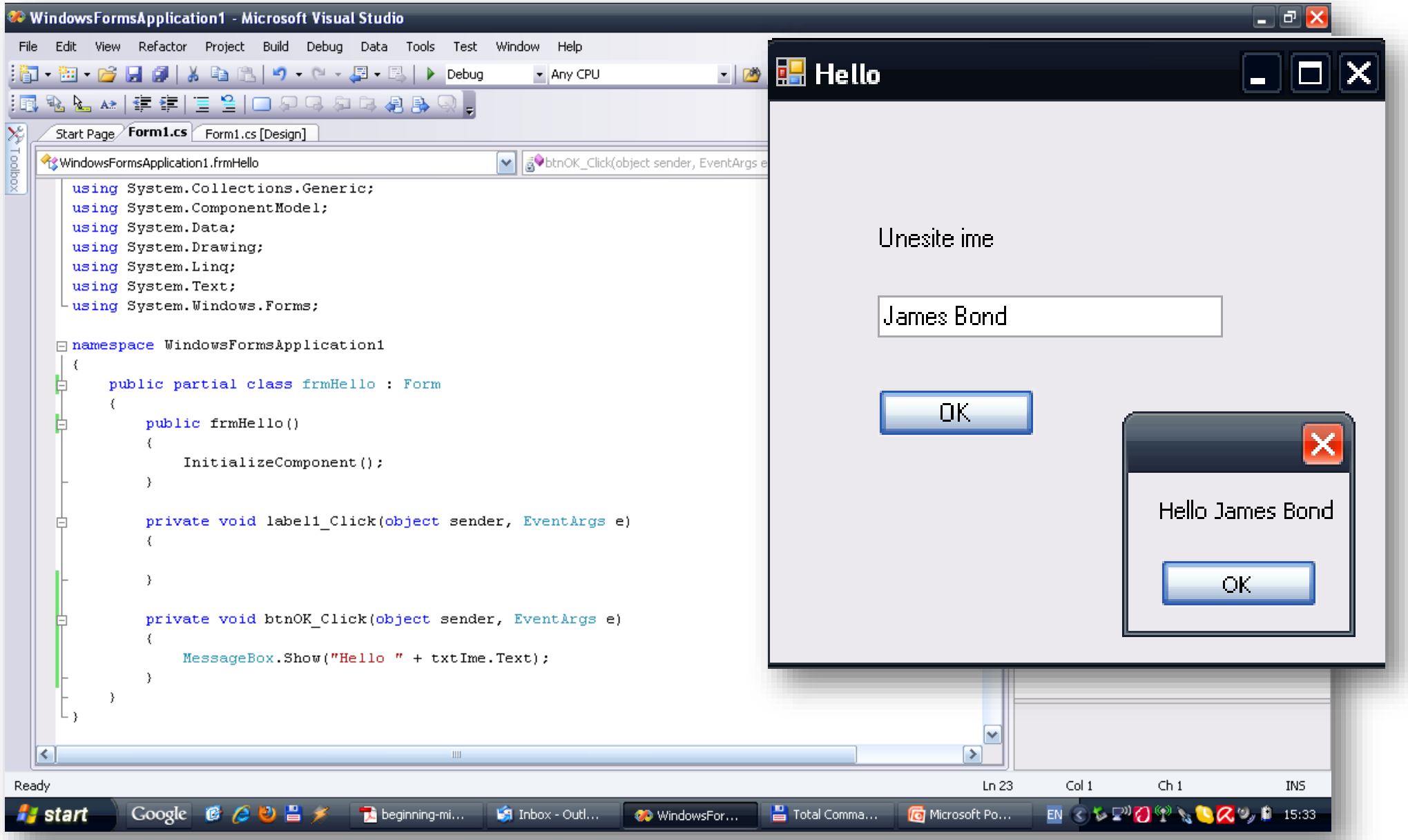
Test konzolne



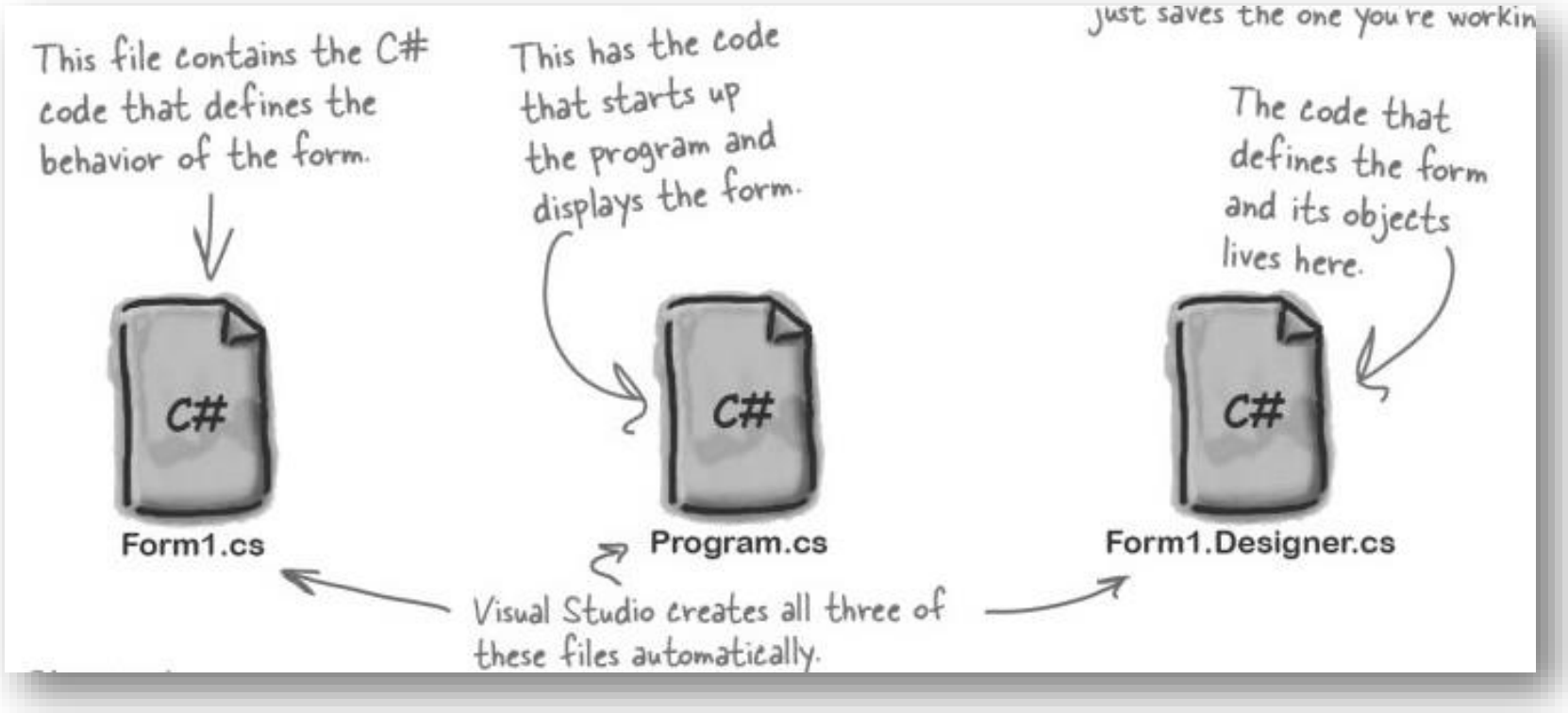
GUI aplikacija



GUI aplikacija



Gde je sve kood?



C#

Varijable, primitivni tipovi, komentari

Elementi C# jezika

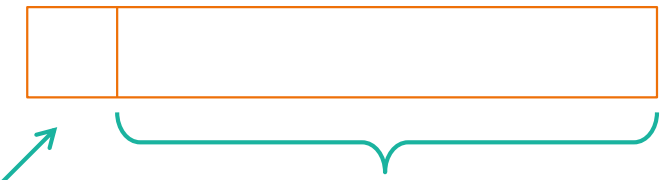
- Sintaksa jezika predstavlja skup pravila koja definišu format i konstrukciju naredbi – komandi u programskom jeziku
- Semantika predstavlja specifikaciju funkcije – onoga što naredba radi, izvršava
- Statement je iskaz, naredba, komanda koja izvršava neku akciju
- Više naredbi u određenom redosledu sačinjava metod – imenovani skup naredbi
- Sintaksa jezika C# je slična C jeziku, tako da C# zajedno sa C++, JAVA-om, JavaScript-om, PHP-om i drugim jezicima spada u C-olike jezike
- Svaka naredba se mora završiti sa ;
- Razmak, tab-ovi, novi redovi – white space služe kao separatori

Identifikatori

Identifikatori predstavljaju imena imenskih prostora, klasa, metoda, promenljivih.

Pravilo imenovanja

CaSe SeNsItIvE!
Velika i mala slova u nazivima nisu isto.



First character

Allowed	Not Allowed
a-z, A-Z, _, @	0-9

Subsequent characters

Allowed	Not Allowed
a-z, A-Z, _, 0-9	@

Identifikatori

Ispravno

```
myBigVar
VAR1
_test
```

Neispravno

```
99BottlesOfBeer
namespace
It’s-All-Over
```

camelCase konvencija imenovanja

```
age
lastName
```

Ključne reči

Ključne reči predstavljaju rezervisane identifikatore koji imaju odgovarajuće značenje i koje koristi C#
Ključne reči se ne mogu koristiti kao korisnički identifikatori

abstract	const	extern	int	out	short	typeof
as	continue	false	interface	override	sizeof	uint
base	decimal	finally	internal	params	stackalloc	ulong
bool	default	fixed	is	private	static	unchecked
break	delegate	float	lock	protected	string	unsafe
byte	do	for	long	public	struct	ushort
case	double	foreach	namespace	readonly	switch	using
catch	else	goto	new	ref	this	virtual
char	enum	if	null	return	throw	void
checked	event	implicit	object	sbyte	true	volatile
class	explicit	in	operator	sealed	try	while

Varibles - promenljive

- Promenljiva predstavlja imenovanu lokaciju u RAM memoriji računara koja čuva vrednost određenog tipa.
- Svaka promenljiva mora biti jedinstveno označena.
- Imenovanje promenljive je sasvim proizvoljno u okviru kombinacije dozvoljenih znakova. Ipak, postoje neka generalna pravila
  - Ne savetuje se imenovanje promenljivih koje se razlikuju samo prema velikim i malim slovima
  - Ne savetuje se korišćenje underscore-a
  - Savetuje se da promenljiva počne malim slovom
- Da bi C# sarađivao sa drugim .NET jezikom neophodno je da se ne koristi underscore i da se ne koristi razlikovanje promenljivih samo prema veličini slova. Ispravna, preporučena imena promenljivih:
  - score, footballTeam
  - Ispravno, ali se ne preporučuje:
    - _score, FootballTeam

Deklaracija promenljivih

- C# je strogo tipiziran jezik - svaki identifikator je neophodno deklarisati pre upotrebe
- Deklaracijom se navodi tip i ime promenljive

Primer deklaracije:

```
int age;
```

Dodeljivanje vrednosti deklarisanoj promenljivoj (definisanje):

```
age = 42;
```

Upotreba promenljive:

```
Console.WriteLine(age);
```

Primitivni tipovi

Data type	Description	Size (bits)	Range [1]	Sample usage
int	Whole numbers	32	-2^{31} through $2^{31} - 1$	`int count;` `count = 42;`
long	Whole numbers (bigger range)	64	-2^{63} through $2^{63} - 1$	`long wait;` `wait = 42L;`
float	Floating-point numbers	32	$\pm 1.5 \times 10^{45}$ through $\pm 3.4 \times 10^{38}$	`float away;` `away = 0.42F;`
double	Double-precision (more accurate) floating-point numbers	64	$\pm 5.0 \times 10^{-324}$ through $\pm 1.7 \times 10^{308}$	`double trouble;` `trouble = 0.42;`
decimal	Monetary values	128	28 significant figures	`decimal coin;` `coin = 0.42M;`
string	Sequence of characters	16 bits per character	Not applicable	`string vest;` `vest = "fortytwo";`
char	Single character	16	0 through $2^{16} - 1$	`char grill;` `grill = 'x';`
bool	Boolean	8	True or false	`bool teeth;` `teeth = false;`

Primitivni tipovi

C# Data Type	Java Data Type	Runtime Type	Size (bytes)	Range	Description
bool	boolean	Boolean	n/a	True or false	Boolean Value
byte		Byte	1	0 to 255	Unsigned Integer
char	char	Char	2	0x0000 to 0xFFFF	Unicode Character
decimal		Decimal	8	–79,228,162,514,264,337,593,543,950,335 to 79,228,162,514,264,337,593,543,950,335	Decimal Number
double	double	Double	8	–1.79769313486232e308 to 1.79769313486232e308	Double-Precision 64- Bit Number
float	float	Single	4	–3.402823e38 to 3.402823e38	Single-Precision 32-Bit Number
int	int	Int32	4	–2,147,483,648 to 2,147,483,647	Signed Integer
long	long	Int64	8	–9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	Signed Integer
sbyte	byte	SByte	1	–128 to 127	Signed Integer
short	short	Int16	2	–32768 to 32767	Signed Integer
uint		UInt32	4	0 to 4,294,967,295	Unsigned Integer
ulong		UInt64	8	0 to 184,467,440,737,095,551,615	Unsigned Integer
ushort		UInt16	2	0 to 65535	Unsigned Integer

Definite Assignment Rule

- *Definite Assignment Rule* - promenljivoj se mora dodeliti vrednost pre korišćenja

```
int age;
Console.WriteLine(age); // compile-time error
```

Literali

Integer literali

```
85          /* decimal */
0x4b        /* hexadecimal */
30          /* int */
30u         /* unsigned int */
30l         /* long */
30ul        /* unsigned long */
```

Decimal literali

```
3.14159       /* Legal */
314159E-5L    /* Legal */
510E          /* Illegal: incomplete exponent */
210f          /* Illegal: no decimal or exponent */
.e55          /* Illegal: missing integer or fraction */
```

String literali

```
"hello, dear"
"hello, \
dear"
"hello, " "d" "ear"
@"hello dear"
```

Konverzija između tipova

- Implicitna (automatska)
- Explicitna (kastovana)

- Tipovi su identični ili kompatibilni
  - int64 -- int
  - float -- int
- Ciljani ti je širi od izvornog

- Nisu sve uspešne
- Deo podataka može biti izgubljen

Konverzija između tipova

```
using System;
class ImplicitConversion
{
  public static void Main()
  {
   byte a=1;
   int b=1234;
   int c=a; //Implicit cast
   double d=b; //Implicit cast
   Console.WriteLine("{0}", c);
   Console.WriteLine("{0}", d);
  }
}
1
1234
```

```
using System;
class ExplicitConv
{
  static void Main()
  {
    double a = 5.654321;
    int b;
    b = (int) a;
    Console.WriteLine("The value
                          is {0}", b);
  }
}


The value is 5
```

Konverzija između tipova

MessageBox.Show() prima string tip parametra. Tip će biti odgovarajući zbog automatske konverzije.

```
long x = 139401930;

MessageBox.Show("The answer is " + x);

MessageBox.Show(x);
```

MessageBox.Show() prima string tip parametra. (Compile error)

rešenje

```
MessageBox.Show(x.ToString());
```

Artimetički operatori

Operator	Category	Example Expression	Result
+	Binary	`var1 = var2 + var3;`	var1 is assigned the value that is the sum of var2 and var3.
-	Binary	`var1 = var2 - var3;`	var1 is assigned the value that is the value of var3 subtracted from the value of var2.
*	Binary	`var1 = var2 * var3;`	var1 is assigned the value that is the product of var2 and var3.
/	Binary	`var1 = var2 / var3;`	var1 is assigned the value that is the result of dividing var2 by var3.
%	Binary	`var1 = var2 % var3;`	var1 is assigned the value that is the remainder when var2 is divided by var3.
+	Unary	`var1 = +var2;`	var1 is assigned the value of var2.
-	Unary	`var1 = -var2;`	var1 is assigned the value of var2 multiplied by -1.

Operator	Category	Example Expression	Result
++	Unary	`var1 = ++var2;`	var1 is assigned the value of var2 + 1. var2 is incremented by 1.
--	Unary	`var1 = --var2;`	var1 is assigned the value of var2 - 1. var2 is decremented by 1.
++	Unary	`var1 = var2++;`	var1 is assigned the value of var2. var2 is incremented by 1.
--	Unary	`var1 = var2--;`	var1 is assigned the value of var2. var2 is decremented by 1.

Operatori dodele

Operator	Category	Example Expression	Result
=	Binary	`var1 = var2;`	var1 is assigned the value of var2.
+=	Binary	`var1 += var2;`	var1 is assigned the value that is the sum of var1 and var2.
-=	Binary	`var1 -= var2;`	var1 is assigned the value that is the value of var2 subtracted from the value of var1.
*=	Binary	`var1 *= var2;`	var1 is assigned the value that is the product of var1 and var2.
/=	Binary	`var1 /= var2;`	var1 is assigned the value that is the result of dividing var1 by var2.
%=	Binary	`var1 %= var2;`	var1 is assigned the value that is the remainder when var1 is divided by var2.

Prioriteti operatora

Precedence	Operators
Highest	++, -- (used as prefixes); +, - (unary)
	*, /, %
	+, -
	=, *=, /=, %=, +=, -=
Lowest	++, -- (used as suffixes)

Booelan, Logički i bitwise operatori

Boolean

```
== , !=, <, >, <=, >=
```

Logički

```
&&  -   AND
||  -   OR
!   -   NOT
```

Bitovski

```
&   - AND
|   - OR
~   - KOMPLEMENT
^   - XOR
>>  - pomeranje udesno
>>  - pomeranje ulevo
```

Booelan, Logički, bitwise operatori Uslovni operator

Boolean

```
== , !=, <, >, <=, >=
```

Logički

```
&&  -   AND
||  -   OR
!   -   NOT
```

Bitovski

```
&   - AND
|   - OR
~   - KOMPLEMENT
^   - XOR
>>  - pomeranje udesno
>>  - pomeranje ulevo
```

Uslovni

```
string result = (myInteger < 10)
                ? “Less than 10”
                : “Greater than or equal to 10”;
```

Unos podataka

`Readline()` vraća string vrednost

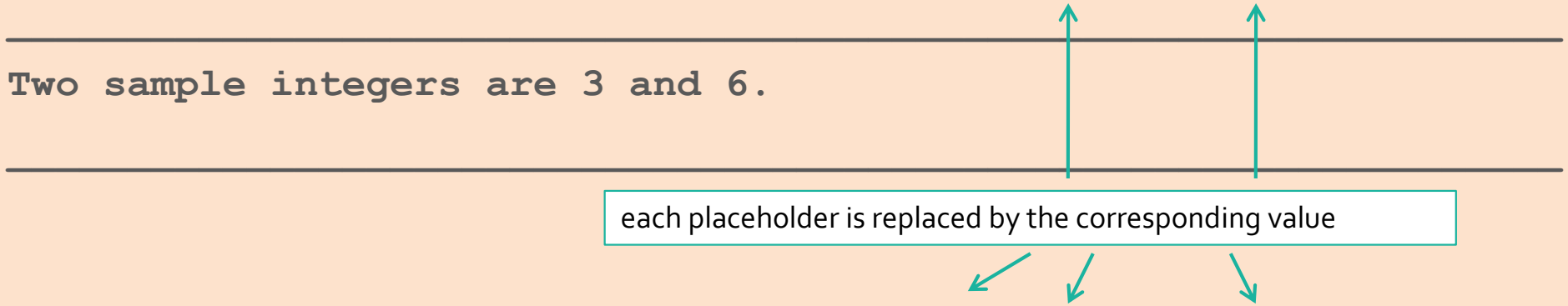
```
string userName = Console.ReadLine();
```

Za učitavanje brojeva koristiti `Convert`

```
double firstNumber = Convert.ToDouble(Console.ReadLine());
```

Ispis

```
Console.WriteLine("Two sample integers are {0} and {1}.", 3, 6);
```




```
Two sample integers are 3 and 6.
```

```
Console.WriteLine("Three integers are {1}, {0} and {1}.", 3, 6);
```

```
Three integers are 6, 3 and 6.
```

```
Console.WriteLine("Two integers are {0} and {2}.", 3, 6); // Error!
```