
МАРШРУТНА КАРТА
Дестинация
Дамаск, Сирия
Пейджър
Теми по пътя за Дамаск:
Дамаск
Оператор за присвояване
Типове
Прости типове
Какво представляват променливите?
Деклариране на променливи
Инициализиране на променливи
Използване на променливи
Други типове
Пристигане в Дамаск
1. Копенхаген, Д
2. Милано, Италия
3. Барселона, Испани
4. Казабланка, Мароко
5. Кайро, Египет
6. Дамаск, Сирия
7. София, България
8. Киев, Украйна
9. Катманду, Непал
10. Куала Лумпур, Малайзия
11. Пекин, Китай
12. Сеул, Корея

Дамаск

Потегляме към Дамаск, Сирия. Дамаск е разположен в югоизточната част на Сирия от двете страни на реката Барада. По-голямата част от града, в която се намира антично правоъгълно селище, е на южния бряг на реката.

По пътя за Дамаск, ще ви науча за типовете данни, като низове от символи и цели числа. Когато задавате свойствата на контролите посредством код, трябва да се сигурно, че работите с правилния тип данни. През по-голямата част от пътуването ни, ще говорим за променливите и начина, по който те се използват в кода. Променливите се използват за съхраняване на данни от различен тип. Те са основната градивна единица на кода. Знам, че схващате бързо и ще започнете да ползвате променливите в кода си толкова много, колкото аз и Ники.

Оператор за присвояване

Ники ни подкани да започнем с писане на реален и полезен Visual Basic код. Знаете как да четете и записвате стойности за свойствата на контролите. Знаете ли, че когато задавате стойност на свойство, използвате оператора за присвояване? Вече говорихме за това. Операторите за присвояване установяват стойността на една променлива на друга променлива. Кодът, илюстриращ това, изглежда като математическо равенство.

```
TextBox1.Text = "Code Rules"
```

Едно от нещата, които трябва да знаете за оператора за присвояване е, че дясната страна на равенството (частта от дясно на знака равно) се изчислява първа. След това лявата страна на равенството (частта от ляво на знака равно) приема изчислената стойност. Изглежда обратно, но това е начина, по който програмата работи. В кода по-долу първо се определя низът "Code Rules". След това получената стойност се записва в текстовото поле.

```
TextBox1.Text = "Code Rules"
```

Типове

Друго нещо, което трябва да запомните, че лявата и дясната страна на равенството трябва да са от един и същ тип (текст, число, цвят и т.н.). Ще ви покажа какво имам предвид като използвам кода от следния пример:

```
TextBox1.Text = "Code Rules Rocks"
```

В лявата страна на равенството е свойството Text на контрола TextBox1, което винаги трябва да приема низ от символи, така че дясната страна също трябва да бъде низ от символи. Ако дясната страна е от различен тип, при изграждането на програмата ще се появи грешка. Тъй като "Code Rules Rocks" е низ от символи, а свойството Text изисква точно такъв тип стойност, то грешка няма да възникне.
Следният код няма да се компилира, защото се опитваме да присвоим на свойството Text, което очаква низ от символи, стойността на свойството ForeColor, което е от тип цвят.

```
TextBox1.Text = TextBox1.ForeColor 'грешен резултат
```

Следният код е правилен, защото дясната страна на равенството е от целочислен тип, а свойството Height може да приема само целочислени стойности.

```
TextBox1.Height = 200
```

Кодът, който следва е грешен, защото дясната страна е дробно число..

```
TextBox1.Height =
200.5
```

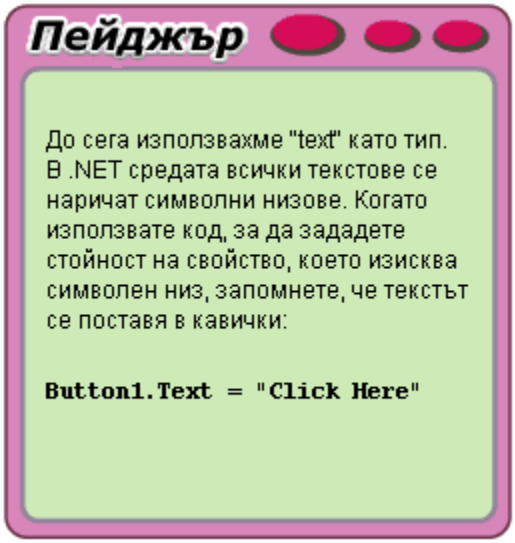
Запомнете, че винаги когато използвате оператор за присвояване, лявата и дясната страна на равенството трябва да бъдат от един и същ тип.

Полезен съвет

Visual Basic често успешно преобразува несъответстващите типове данни. Това обаче не винаги е гарантирано, така че трябва да се задават правилни типове за данните. C# допуска още по-малко несъответствия в типовете.

Прости типове

Средата за програмиране поддържа набор от типове наречени прости или примитивни типове. Вече използвахте типа string, за да зададете стойност на свойството Text на бутона, и типа integer, за да зададете стойност на свойството Height на формата. Дори използвахте типа Boolean (Истина или Лъжа) за да зададете стойност на свойството Visible на текстовото поле.



Някои от най-често използваните прости типове са: String (текст), Integer (цели числа), Single (десетични числа) и Boolean (Истина или Лъжа). Те са основни за всички .NET езици за програмиране като Visual Basic.NET и C#.

Ето някои примери за позволените от тези примитивни типове стойности:

Тип	Пример	Описание
String	"Hello"	Текстът трябва да е в кавички
Integer	123	Цели числа без десетична точка
Single	55.12	Числа с десетична точка
Boolean	True	Използват се само стойности True или False

Полезен съвет

Повечето ученици се затрудняват с правилото, че число като например 1, не е еквивалентно на символа “1”. Двоичното представяне на числото е 00000001. Двоичното представяне на символа е 00110001. Определено те не са еднакви.

Visual Basic скрива това, защото опитва да направи преобразуване, ако е необходимо. Например,
```
Dim A as Integer = 1
Dim B as String = “1”
If (A = B) Then
```
Кодът по-горе връща стойност True въпреки факта, че вътрешното представяне е различно.

Напомнете на учениците да не използват директно свойството Text на текстовото поле в аритметични операции.

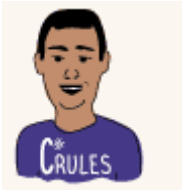
Разгледайте примерите по-долу, които използват оператора за присвояване, за задаване стойности на свойства от различни типове:

```
TextBox1.Text = "Smile"
TextBox1.Visible = True
TextBox1.Width = 1000
```

Пейджър

КОгато избирате свойство или метод от списъка на програмата IntelliSense, се визуализира прозорец с помощна информация. Той показва типа на стойността, която може да се присвои на свойството или типа на аргументите, които могат да се подадат на метода.

```
Button1.Text = "Click Here"
```



Ето и нещо хубаво! Тъй като C# поддържа същите прости типове както Visual Basic - String, Integer, Single, Boolean и всички останали.

Какво представляват променливите?

Променливите се използват в кода ви, за да пазят информация, която може да бъде необходима по-късно. Те са като променливите в математиката. Използват се за задаване и съхраняване на стойности. Тяхната стойност може да бъде променена по всяко време.

В кода променливите се използват за "запомняне" на информация. Ако кодът ви не работи с променливи, програмата ще спира да пита потребителя за конкретна информация веки път, когато й е необходима такава. Представете си, че на програмата й трябва информация за възрастта на потребителя три пъти по време на изпълнението й. Няма ли да бъде досадно да спира и да пита потребителя за възрастта му три пъти?

Въведи възрастта си.
Въведи възрастта си.
Въведи възрастта си.

Използвайки променливи, програмата ще попита потребителя за възрастта му еднократно. Полученият отговор ще се съхрани в променлива, която ще се използва от кода всеки път, когато е необходимо.

Защо използваме променливи?

Както видяхте, променливите съхраняват данни при изпълнението на програмата. Те намаляват информацията, която потребителят трябва да въвежда. Те също така помагат за редуциране на грешките. Ако трябва да въвеждате едно дълго число много пъти, съществува голяма вероятност да сбъркате някоя от цифрите.

Променливите също така се използват да съхраняват резултати от изчисления. Те се използват и за определяне на пътя, по който ще продължи изпълнението на кода, и за преброяване като например, колко пъти ще се направи дадено изчисление.

Изучаването на начина, по който могат да се използват променливите, е с фундаментално значение. Те са абсолютно необходими при изпълняването на определени изчисления и действия в кода.

Деклариране на променливи

Преди програмата да използва дадена променлива, вие трябва да я декларирате. За да направите това, е необходимо да напишете ред код, с който да зададете име и тип на променливата. Всички променливи във Visual Basic трябва да се декларират. Това е добър навик, защото ви кара да се замислите върху предназначението на променливата и нейния тип. Декларирането на променливите ускорява времето за компилация и прави изпълнението на програмата по ефективно. То също така предотвратява неправилното използване на променливата в кода ви.

За да декларирате променлива във Visual Basic, използвайте ключовата дума "Dim".

Ключовата дума Dim има следния синтаксис:

```
Dim VariableName As VariableType
```

Ето някои примери за деклариране на променливи от различни типове.

```
Dim MyName As String
Dim MyWeight As Integer
Dim NoBrainer As Boolean
Dim DVDPrice As Single
```

Първата променлива MyName е декларирана от тип String. Това означава, че тя може да приема само стойности от тип String. Втората променлива MyWeight е декларирана от тип Integer. Тя може да приема само целочислени стойности. Променливата NoBrainer може да приема само стойности от тип Boolean, а променливата DVDPrice – само стойности от тип Single.

Пейджър

Имената, които избирате за променливите, са важни. Те трябва да са описателни, но и къси. Използвайте абревиатура, ако е необходимо. Много програмисти използват стандартна конвенция за именуване, при която всяка дума от името на променливата започва с главна буква. Например: CamelCase, MyName, WinnebagoTopSpeed, LocalSpeedLimit. Visual Basic преобразува в главна буква първия символ. C# не прави това.

Button1.Text = "Click Here"

Полезен съвет

Учениците имат навик да именуват променливите с безсмислени имена или отделни букви. По-късно те забравят какво означават те, което води до допускане на грешки. След това, когато учениците потърсят помощ, човекът, който им помага не знае кои променливи за какво са използвани.

Можете да откажете помощ на ученик, който не е използвал смислени имена на променливи.

В C# декларирането на променливи става по малко по-различен начин. Първо трябва да напишете типа на променливата, след което нейното име. Имената на типовете също са по-различни. Например, вместо Integer се използва int, а вместо Single се използва decimal.

```
int val4;
string myName;
decimal myBattingAverage;
```

Къде декларирате променливите?

Можете да декларирате толкова променливи, колкото са ви необходими. Те могат да са от произволен тип, поддържан от Visual Studio средата. Аз и Ники спазваме стандартни правила за писане на код. Декларираме променливите в началото на кода. Така знаем къде да ги търсим и лесно можем да видим типа им. Това прави кода ни подреден и организиран. Ако трябва да създадем друга променлива, се връщаме към началото на програмата и я декларираме заедно с останалите променливи. Не декларираме променливите в средата на кода.

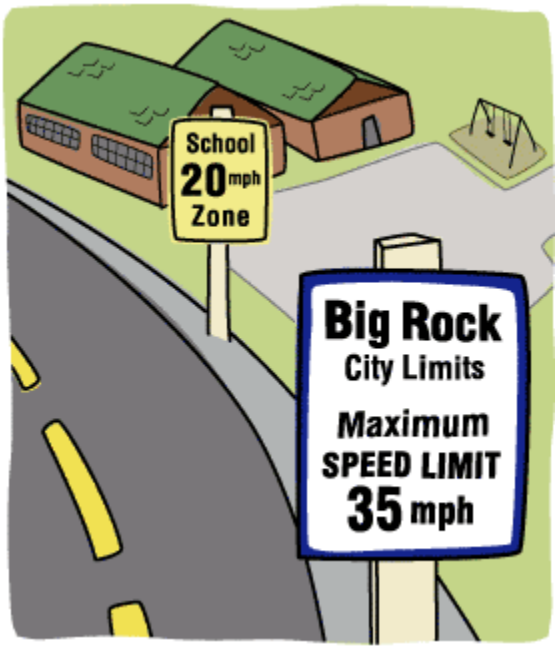
Можете да декларирате променливите глобално или локално. До сега кодът, който писахме, беше в обработчиците на събитията за контролите. Например, добавяхме код в обработчика на генерираното при натискане на бутон събитие. Променливите, които се декларират в тези обработчици са локални и могат да бъдат променяни само от кода в съответни обработчик. Те не са достъпни за останалия код от формата. Стойностите на локалните променливи само докато се изпълнява кода от обработчика на събитието. След това техните стойности се изгубват. Пример за локална променлива е декларираната в обработчика на генерираното при натискане на бутон събитие.

За да декларирате локална променлива, поставете ключовата дума Dim в кода на обработчика на дадено събитие. Декларирането на променливите е задължително преди да напишете какъвто и да било друг код. Ето и един пример:

```
Private Sub Button1_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
Dim MyName As String
MyName = "Bill"
TextBox1.Text = MyName
End Sub
```

Полезен съвет

MyName е декларирана в процедурата. След като изпълни израза End Sub MyName се унищожава.



До сега не сме използвали глобални променливи. Те могат да бъдат четени или записвани в рамките на целия код за формата и са достъпни от всички обработчици на събития. Стойностите на глобалните променливи се съхраняват през цялото време, когато формата е отворена. Пример за глобална променлива е брояч, който помни колко пъти са натиснати бутоните от формата.

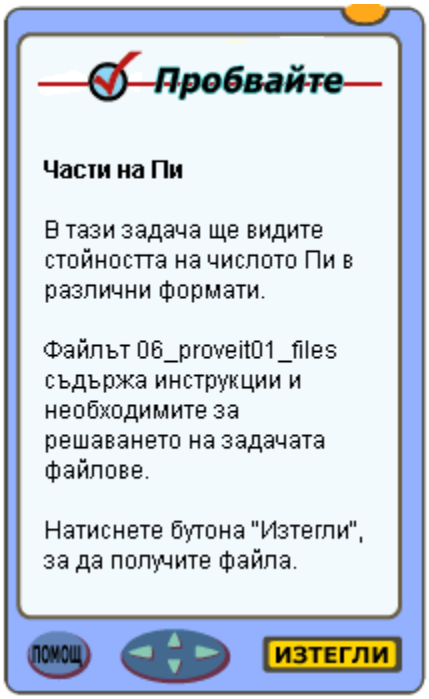
За да декларирате глобална променлива, трябва да поставите ключовата дума Dim в класа на формата преди обработчиците на събития. Разгледайте следния пример:

```
Dim TotalButtonClicks As Single
```

Променливата TotalButtonClicks е глобална променлива. Тя може да бъде четена или записвана навсякъде в кода на формата.

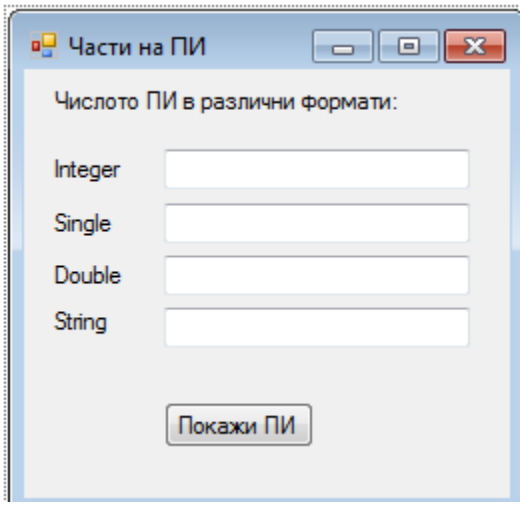
Ако декларирате всичките си променливи в класа на формата, извън обработчиците на събития, то те ще бъдат глобални и стойностите им ще могат да бъдат четени или записвани навсякъде.

06 Пробвайте 01 Части на ПИ



Любимият ми десерт, когато ходя на ресторант е ябълков пай. Това ми напомни по колко различни начина може да бъде представено числото Пи.

1. Създайте форма подобна на показаната по-долу:



2. В обработчика на събитието Button1_Click, дефинирайте променливи от следните типове:

   Integer
   Single
   Double
   String

3. Задайте стойности за Пи на всяка променлива като използвате вградения в Visual Basic Пи-калкулатор Math.PI

4. Асоциирайте Math.PI към всяка променлива.

5. Асоциирайте всяка една от променливите към съответното текстово поле.

Ако програмата работи правилно, я покажете на своя учител.

Инициализиране на променливи

Веднъж след като сте декларирали една променлива, можете да я използвате в кода си. Преди това, обаче трябва да й зададете инициализираща стойност. Можете да използвате произволна стойност стига да е от подходящ тип, т.е. от типа на променливата. Инициализирането на променлива става посредством оператора за присвояване. Разгледайте следните примери.

```
Dim MyName As String
MyName = "Paul Bunion"

Dim MyWeight As Integer
MyWeight = 128

Dim DrivingDistance As Single
DrivingDistance = 12.8
```

Променливата MyName е декларирана от тип а String и е инициализирана със стойността "Paul Bunion". Променливата MyWeight е декларирана от Integer тип и е инициализирана със стойността 128. Променливата DrivingDistance е декларирана от тип Single и е инициализирана със стойността 12.8.

Ето ви един шаблон, който можете да използвате, за да декларирате и инициализирате променливи. Това ви гарантира, че използването на променливата ще започне с подходяща стойност. Използването на неинициализирани променливи може да доведе до грешка в програмата ви.

```
Dim MyAge As Integer = 100
```

Инициализирането на променливи е подобно на това в C#:

```
int myOrbit;
myOrbit=123000;
```

Използване на променливи

Сега след като сте декларирали и инициализирали променливата си, можете да я използвате винаги, когато стойността ѝ ви е необходима. Разгледайте примерите.

```
Dim MyName As String
MyName = "Joe Cooker"
MessageBox.Show(MyName)
```

Какво съобщение предполагате, че ще се визуализира в диалоговия прозорец?
Джо Кокър!

```
Dim MyName As String
MyName = "Peter, Paul and Mary"
TextBox1.Text = MyName
```

Можете да променяте стойността на променливата посредством оператора за присвояване. Какво ще се покаже е текстовото поле?

```
Dim MyName As String
MyName = "Bill"
MyName = "Bob"
MyName = "Ben"
TextBox1.Text = MyName
```

Кое от имената ще се изпише в контрола TextBox1? Ако заложите на "Ben", ще спечелите! Променливата MyName приема стойност Bill, след това Bob и накрая Ben. Тъй като последната стойност, която ще получи е Ben, то тя ще се визуализира и в текстовото поле.

Можете да присвоите стойността на една променлива на друга променлива.

```
Dim MyName As String
Dim YourName As String
MyName = "Allen Ginsberg"
```

```
YourName = MyName
MessageBox.Show(YourName)
```

В диалоговият прозорец ще се покаже съобщение Allen Ginsberg.

Ето и код на C#. Можете да видите някои прилики и някои разлики. Вижте как декларираме променливи. Малко по-различно е от Visual Basic 2010.

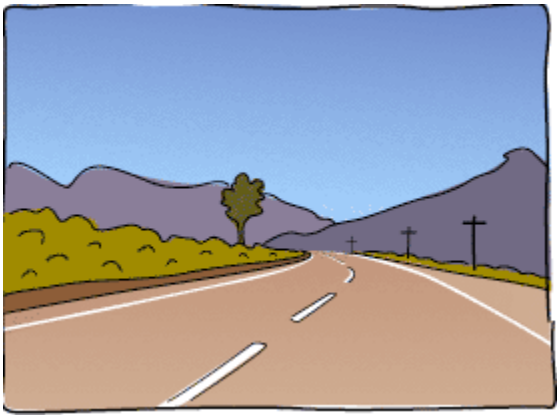
```
string myName;
string YourName;
MyName = "Allen Ginsberg";
YourName = myName;
MessageBox.Show(YourName);
```

Ето ви един пример,при който ще възникне грешка, тъй като се опитваме да присвоим стойността на променлива от един тип на променлива от друг тип. Следният код няма да се компилира.

```
Dim MyName As String
Dim DVDPrice As Single
MyName = "Paul Bunion"
DVDPrice = MyName
```

Запомнете, че променливите се декларират от определен тип и могат да приемат стойности само от този тип.

Променливите в примери



Разказах ви за това как да декларирате и инициализирате променливи. Има ли още нещо? Нека да напишем код като използваме променливи!

Създайте ново Windows приложение с Visual Studio с име XACTLY. Отворете прозореца Toolbox и добавете върху формата един бутон и две текстови полета. Изчистете текста от текстовите полета. Задайте стойност "Show XY" за свойството Text на бутона. Задайте стойност True за свойството ReadOnly на текстовите полета. Щракнете двукратно върху бутона, з ада създадете обработчик за събитието Button1_Click. Добавете следния код:

```
Dim XName As String
Dim YName As String
XName = "X is my name"
YName = "Y is my name"
TextBox1.Text = XName
TextBox2.Text = YName
```

Полезен съвет

Секцията може да се разбере по-лесно, ако ученикът паралелно създава същото Visual Studio приложение.

Изградете и стартирайте приложението. Натиснете бутона "Show XY". Кодът ви декларира две променливи от тип String – Xname и YName. Те получават като стойност низ от символи, затворени между кавички. Свойството Text на едното текстово поле приема стойността на променливата XName, а свойството Text на другото текстово поле – стойността на променливата YName. Опитайте да промените стойностите на променливите XName и YName в кода, след което стартирайте програмата отново.

А сега да модифициране програмата. Добавете трето текстово поле. Задайте празен низ за свойството Text. Уверете се, че стойността на свойството ReadOnly е False.

Щракнете двукратно върху бутона, за да редактирате обработчика на събитието Button1_Click. Променете кода така, че да изглежда по следния начин:

```
Dim XName As String
Dim YName As String
Dim ZName As String
ZName = TextBox3.Text
XName = ZName
YName = XName
TextBox1.Text = XName
TextBox2.Text = YName
```

Изградете и стартирайте приложението. Въведете някакъв текст в третото текстово поле. Натиснете бутона "Show XY". Какво се случва? Вашият код декларира три променливи от тип String. Променливата ZName получава стойност от свойството Text на третото текстово поле. Променливата XName приема стойността на променливата ZName. Променливата YName приема стойността на променливата XName. Свойството Text на първото текстово поле приема стойността на променливата XName, а свойството Text на второто текстово поле – стойността на променливата YName. Кодът по-горе показва как да задаваме стойност от една променлива на друга променлива.

Ето и един последен пример, който илюстрира използването на глобални променливи. Създайте ново Windows приложение с име TotalButtonClicks. Добавете върху формата три бутона. Декларирайте следната променлива в класа на формата:


```
Dim TotalButtonClicks As Integer
```

Щракнете двукратно върху първия бутон, за да създадете обработчик на събитието Button1_Click и добавете следния код в него.

```
TotalButtonClicks = TotalButtonClicks + 1
MessageBox.Show(TotalButtonClicks)
```

Добавете същия код и в обработчиците на събитията Button2_Click и Button3_Click.

Изградете и стартирайте приложението. Щракайте върху бутоните в произволна последователност. Текущият общ брой на щракванията върху бутоните се извежда в диалогов прозорец. Това е така, защото декларирахме глобална променлива с име TotalButtonClicks в кода на формата. Тя се използва за съхраняване на общия брой на щракванията върху бутоните. Нейната стойност се инкрементира всеки път, когато потребителят натисне някой от бутоните. Стойността на глобалната променлива се съхранява докато програмата не бъде затворена.

Други типове

Във Visual Basic освен прости типове можете да използвате още много други типове, за да декларирате променливите си. Те са вградени в Visual Studio средата и могат да се използват от всички езици за програмиране включени във Visual Studio 2010. Много от тях са свойства на т.нар. Системни класове. Системните класове притежават код, който осигурява базовата функционалност на тези езиците. Съществена част от изучаването на програмирането с езиците е усвояването на функционалността на Системните класове.

За сега не е необходимо да изучавате Системните класове. Ники и аз ще ви показваме код, който използва свойствата и методите на Системните класове, така че вие ще се научите да ги разпознавате. Ще ви дам пример за използване на Системните класове за промяна на цвета на формата.
Отворете приложението ShowXY и добавете втори бутон на формата. Задайте стойност "Color" за свойството му Text. Щракнете двукратно върху бутона, за да създадете обработчик на събитието Button2_Click и добавете в него следния код:

```
Dim MyColor As System.Drawing.Color
MyColor = System.Drawing.Color.Blue
Form1.ActiveForm.BackColor = MyColor
```

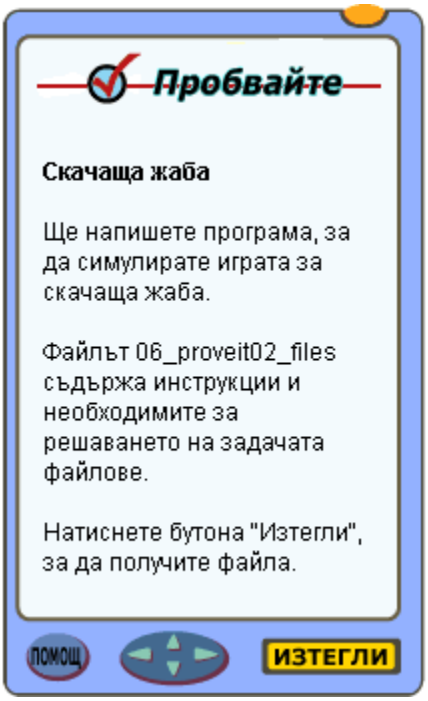
Изградете и стартирайте приложението. Натиснете втория бутон.

Вашият код декларира променлива с име MyColor от тип System.Drawing.Color. Променливата MyColor приема стойност System.Drawing.Color.Blue, с което получава валиден син цвят. Накрая свойството BackColor на формата приема стойността на променливата MyColor. Забележете, че свойството BackColor получава стойност от тип System.Drawing.Color.

Можете да разгледате Системните класове. Начин за това е да напишете System следвано от точка в прозореца с кода на формата. Помощното средство IntelliSense ще ви покаже списък със свойствата и методите на класа System. Изберете елемент от списъка и напишете точка, за да видите, ако разбира се има такива, неговите свойства и методи.

Пристигане в Дамаск

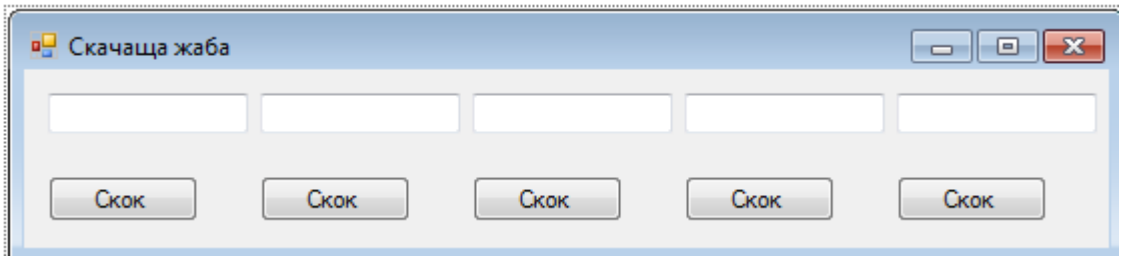
Вече сме в Дамаск! Сигурен съм, че се забавлявахте. Нека да направим задачите, които са на джобния компютър, след което да решим теста, за да ни перфорират контролната карта. Сега след като имате знания за променливите, ще ни помогнете в преодоляването на последното предизвикателство от Комисията по състезанията.



06 Пробвайте 02 Скачаща жаба

Впечатлен от блатото, което видяхме тази сутрин, реших да напишем програма за скачащата жаба.

Създайте форма подобна на показаната по-долу или използвайте шаблон 02 от секция 06, който е разработен за тази цел:

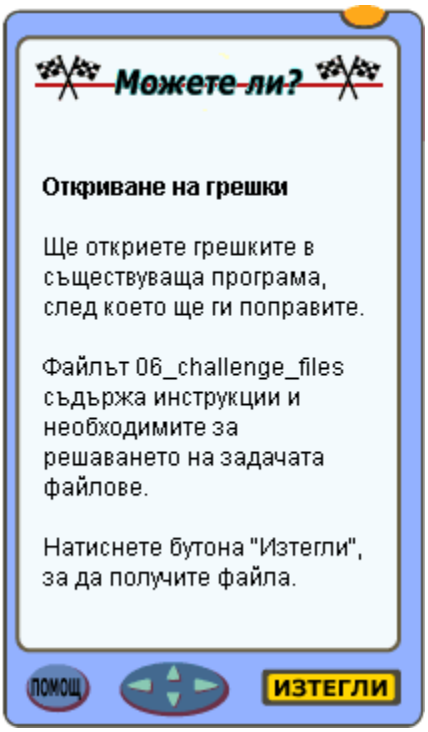


Напишете “Frog1” в първото текстово поле и “Frog2” – във второто.

Когато потребителя натисне бутона с текст ”Скок”, съдържанието на текстовото поле над него ще се появи в текстовото поле, разположено след две съседни текстови полета вдясно. Не забравяйте да изчистите съдържанието на текущото текстово поле.

Ако програмата работи коректно, я покажете на учителя си.

06 Можете ли? Откриване на грешки



Необходимо е да упражним откриването на грешки в кода на програмите. Повечето от нас знаят, че такива грешки се наричат бъгове.

Програмата от шаблона на секция 06 притежава бъгове. Всеки ред с изключение на един има грешка. Вашата задача е да поправите грешките така, че програмата да се изгради и изпълни.

Отворете файла Splat.sln. Открийте бъговете!

Ако програмата работи, правилно я покажете на учителя си.

Полезен съвет

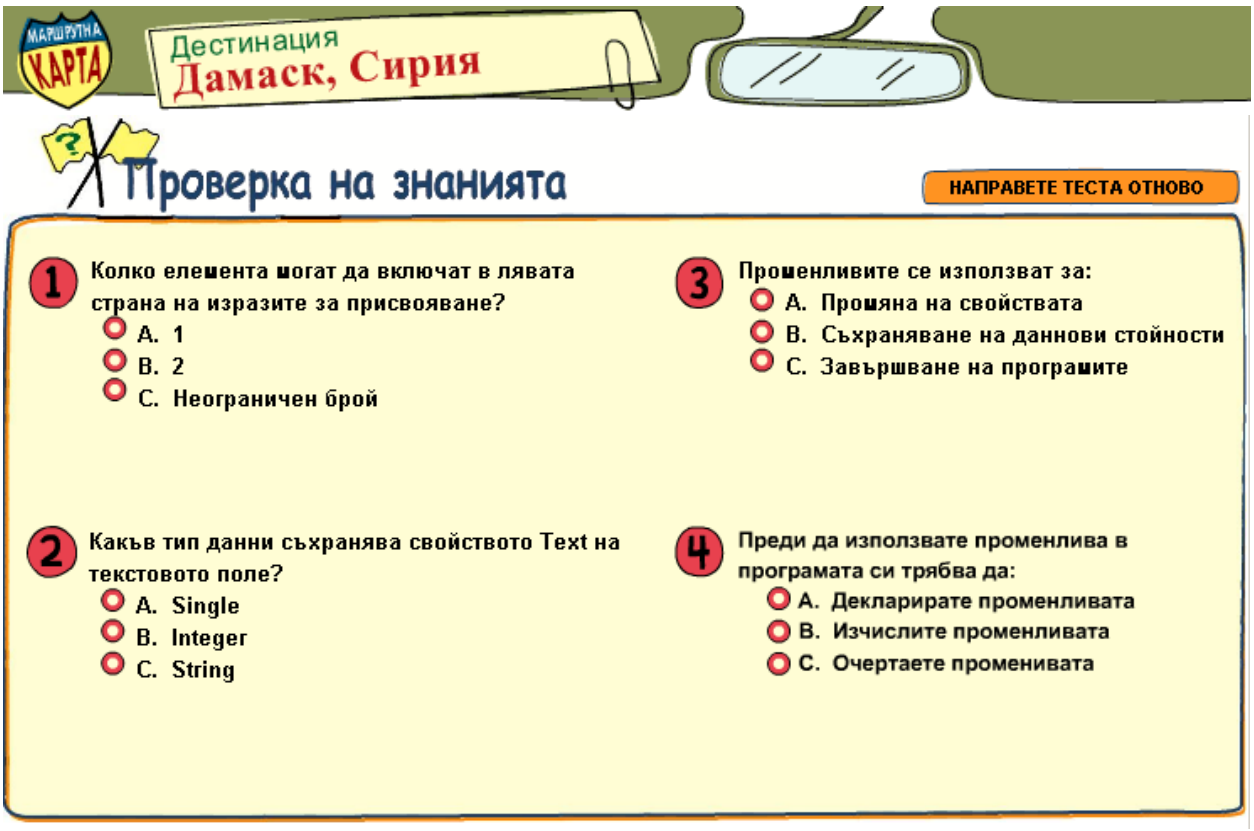
Шаблонът съдържа много умишлени грешки. От ученика се очаква да поправи кода. Не се изисква точно определена корекция. Например, ако X = 123.4 е грешно, то X = 123 е правилно, както и X = 1234

Продължение / Обобщение

Кой е старши капитан Грейс Хоупър?
Защо зададохме този въпрос?

Полезен съвет

Капитан Хоупър е открил първия бъг в информационните технологии - истинска буболечка в електронната лампа на първите компютри.


МАРШРУТНА КАРТА
Дестинация
Дамаск, Сирия
Проверка на знанията
НАПРАВЕТЕ ТЕСТА ОТНОВО
1
Колко елемента могат да включат в лявата страна на изразите за присвояване?
A. 1
B. 2
C. Неограничен брой
2
Какъв тип данни съхранява свойството Text на текстовото поле?
A. Single
B. Integer
C. String
3
Променливите се използват за:
A. Промяна на свойствата
B. Съхраняване на даннови стойности
C. Завършване на програмите
4
Преди да използвате променлива в програмата си трябва да:
A. Декларирате променливата
B. Изчислите променливата
C. Очертаете променивата