РОЛЯТА НА БАЗОВИТЕ ЗАДАЧИ ПРИ ИЗУЧАВАНЕ НА СЪБИТИЙНО ПРОГРАМИРАНЕ ЧРЕЗ СРЕДАТА VISUAL C# В СРЕДНОТО УЧИЛИЩЕ

Стефка Анева

РЕЗЮМЕ

Настоящата разработка разглежда ролята на базовите задачи при изучаване на събитийно програмиране чрез средата Visual C# в профилираната подготовка по информатика в средното училище. Дискутирани са някои възможности, които предоставя средата Visual C# и правилния подбор при използването на основни елементи на ГПИ за реализиране на диалог с потребителя по време на изпълнение на дадено приложение. Разгледана е примерна задача.

Ключови думи: събитийно програмиране, визуално програмиране, графичен потребителски интерфейс, елемент, събитие, събитийна процедура.

ВЪВЕДЕНИЕ

В наши дни визуалното програмиране е специфична област от софтуерната индустрия и е основно средство за разработване на софтуерни продукти. С непрекъснатото развитие и усъвършенстване на технологиите и средствата за изграждане на графичен потребителски интерфейс, процесът на създаване на приложен програмен продукт все повече се превръща и в дизайнерска дейност за създаване на ефективен и достъпен графичен потребителски интерфейс и максимално удобни за работа интерфейсни форми на приложенията.

В сферата на образованието трябва да реагира адекватно на необходимостта да се подготвят кадри, способни да използват и развиват новите технологии. Изучаването на събитийно програмиране в профилираната подготовка по информатика в средното училище чрез подходящо подбран набор от учебни задачи с различна степен на сложност дава възможност на учениците да се запознаят с основните принципи и възможности на визуалното програмиране и да усвоят основни технологии и механизми за реализиране на програми, управлявани от събития с достъпен графичен потребителски интерфейс.

РОЛЯТА НА БАЗОВИТЕ ЗАДАЧИ В ОБУЧЕНИЕТО ПО СЪБИТИЙНО ПРОГРАМИРАНЕ

При **базовите задачи** в обучението по събитийно програмиране се акцентира върху запознаване с предназначението и възможностите на следните основни елементи (обекти) на ГПИ, които се използват често при изграждане на графичен потребителски интерфейс:

- **етикет** – за показване на текст (надписи) върху формата;
- **команден бутон** – за реализиране на действия (команди);
- **текстово поле** – за въвеждане на текстови данни в режим на изпълнение;
- **изображение** – за визуализиране на графични изображения;
- **списъчна кутия** – за изобразяване на списък от символни низове за избор;
- **комбинирана кутия** – за редактиране на текст с възможност за избор и от падащ списък;
- **кутии с отметка** (контролни кутии) – за избор на независими по между си възможности от типа Да/Не;
- **радиобутон** (изборен бутон) – за алтернативен избор на една от няколко взаимно изключващи се възможности.

В (АНЕВА, 2010) е представен проект за организация на учебния процес и методика за провеждане на профилирано обучение по информатика за темата „*Събитийно програмиране в среда на графичен потребителски интерфейс”,* като за целта се използва средата на Visual C#. Предложени са набор базови задачи за запознаване с основните възможности, които предоставя средата Visual C# за създаване на програми, управлявани от събития с ГПИ, включващ горе посочените основни елементи на ГПИ.

<u>В резултат от обучението на този базов етап ученикът трябва да:</u>

- знае да моделира и разработва графичен потребителски интерфейс с визуални средства;
- умее да избира подходящ елемент на ГПИ в съответствие с необходимата функционалност на графичния интерфейс;
- умее да настройва свойствата на елементите на ГПИ в режим на проектиране и режим на изпълнение;
- знае да програмира подразбиращи се и други събития за основни елементи на ГПИ.
- знае механизма за деклариране на променливи и умее да декларира и използва локални и глобални променливи.
- знае механизма за извикване на една събитийна процедура в друга.

Използваме идеята за стъпаловиден инструментариум (ГРОЗДЕВ, 2005) и като следваща стъпка при обучението по събитийно програмиране отделяме решаването на задачи, които се базират на развитие и комбиниране на вече реализирани до момента базови задачи.

На този етап в резултат от обучението ученикът трябва да:

- умее да създава приложения с подходящ дизайн и оформление на ГПИ.
- развие и усъвършенства своите умения за правилен подбор на подходящи елементи на ГПИ в съответствие с необходимата функционалност на графичния интерфейс на приложението.
- затвърди своите знания и умения за деклариране и използване на променливи и да разбере и осмисли необходимостта от преобразуване на данните в определени случаи.
- разбере необходимостта от използване на елемент на ГПИ за групиране във форма от приложение, в което са включени групи радиобутони, предоставящи различна функционалност.
- разбере необходимостта да се използват масиви от елементи на ГПИ.
- се запознае с технологията за добавяне на форма в приложение или създаване на приложение, което съдържа повече от една форми.

В (РАХНЕВ, 2010) са разгледани и систематизирани основни практически принципи, процеси и технологии, играещи важна роля при изграждането на ефективен графичен потребителски интерфейс. В (АНГЕЛОВ, 2010) са представени основни подходи за реализиране на многоезичен потребителски интерфейс и съдържание на дадено софтуерно приложение и са дискутирани някои основни моменти относно въпроса за профилиране и персонализиране на потребителския интерфейс. В (SHOTLEKOV, 2010) са предложени набор от критерии за оценка на качеството при реализиране на студентски проекти за ефективен уеб дизайн и интерактивно съдържание. В (VALCHANOV, 2009) са дискутирани някои основни аспекти относно значението и ролята на „добрите" интерфейси при създаване на бизнес информационни системи.

В настоящата разработка ще разгледаме примерна задача, свързана с извършване на математически пресмятания, като за целта е необходимо да бъде реализиран диалог с потребителя в режим на изпълнение на приложението с помощта на различни способи и чрез използване на различни базови елементи на ГПИ, изучени до този момент. По този начин учениците ще осмислят по-задълбочено функционалното предназначение на тези елементи. В режим на изпълнение на дадено приложение могат да бъдат реализирани различни възможности за диалог с потребителя и въвеждане на входни данни от различен тип. В разгледаната задача ще бъдат демонстрирани следните възможности:

1. Реализиране на диалог с потребителя в режим на изпълнение чрез използване на диалогова кутия за съобщения MessageBox.
2. Реализиране на диалог с потребителя чрез бутони за алтернативен избор на една от няколко взаимно изключващи се възможности (т.е. избор на математическо пресмятане, което да бъде изпълнено).
3. Реализиране на диалог с потребителя чрез опционални бутони за избор на режим на въвеждане на данни в текстово поле (с проверка за коректност

на въведените данни или режим на ограничаване на използването на дадени клавиши от клавиатурата).

4. Въвеждане и корекция на данни в режим на изпълнение с помощта на текстово поле, при което се допуска въвеждане на произволни стойности и извършване на проверки за коректност на въведената информация. Особеността при този тип въвеждане се състои в това, че трябва да се използва променлива, в която да се съхрани въведената информация в текстовото поле. Текстовото поле връща като резултат въведеното в него във вид на текстова стойност, затова при необходимост трябва да бъде преобразуван резултата по подходящ начин, за да може да се реализира присвояването на въведената стойност.
5. Въвеждане и корекция на данни в режим на изпълнение с помощта на текстово поле, при което се ограничава въвеждането от клавиатурата, като се свежда до въвеждане само на допустими за това символи. В този случай е необходимо да се използва събитийна процедура, която обработва събитието KeyPress – натискане на клавиш от клавиатурата.
6. Въвеждане и корекция на числови данни в режим на изпълнение чрез използване на хоризонтални или вертикални скролиращи ленти за въвеждане на по-голяма или по-малка стойност. Този начин на **последователно** въвеждане е удачен да се прилага при задачи, в които трябва да бъдат въвеждани цели числа в някакъв диапазон. Неудобството в случая се състои в това, че тези елементи дават възможност за въвеждане на целочислени стойности в даден диапазон, обхващаш интервала между минималната и максималната стойност, които са стойности на свойствата Minimum и Maximum на скролиращата лента.

ПРИМЕРНА ЗАДАЧА

Ще разгледаме примерна задача, в която се демонстрират различни възможности да въвеждане на числови данни и способи за осъществяване на диалог с потребителя в режим на изпълнение.

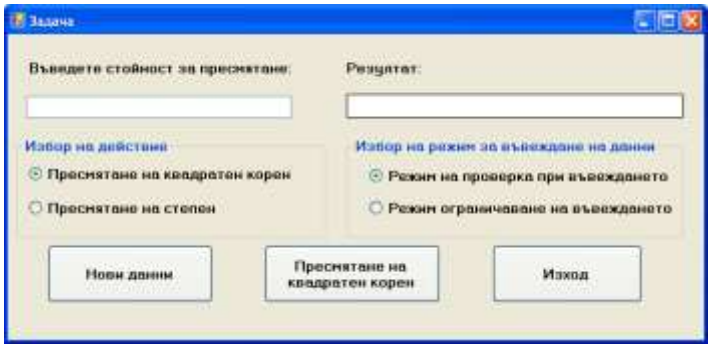
Задача: Създайте приложение, което да извършва следните пресмятания:

- ✓ при въвеждане на дадено число в текстово поле, с помощта на команден бутон да се пресмята квадратния корен на числото и полученият резултат да се визуализира в етикет (Фиг. 1).
- ✓ при въвеждане на дадено число в текстово поле и стойност за степенен показател във второ текстово поле, с помощта на същия команден бутон да се пресмята съответната степен на числото (Фиг. 2).

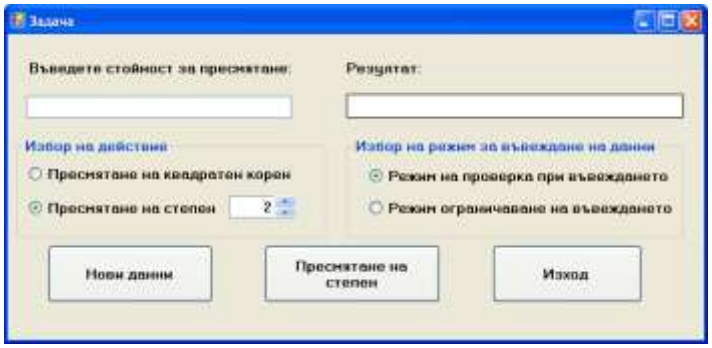
Да се реализира възможност за избор на режим на въвеждане на данни:

- ✓ с проверка за коректност след въвеждането, т.е дали въведените данни отговарят на допустимите стойности за извършване на съответното пресмятане;
- ✓ с ограничаване на въвеждането: при пресмятане на квадратен корен от клавиатурата да могат да се въвеждат само цифри и десетична запетая,

а при пресмятане на степен – цифри, десетична запетая и знака ”минус” за въвеждане на отрицателни числа.
Съхранете създаденото от вас приложение с име Zadacha.



Фиг. 1 *Пресмятане на квадратен корен*



Фиг. 2 *Пресмятане на степен*

Основни цели:

- учениците да задълбочат своите познания и умения за работа с вече разгледаните в базовите задачи елементи – етикет, команден бутон, текстово поле и радиобутон;
- да разберат предназначението на елемента за групиране (GroupBox) на функционално обособени елементи на ГПИ;
- учениците да разберат предназначението и да се запознаят с някои свойства на разгледания в задачата нов елемент на ГПИ - вертикална скролираща лента (VScrollBar).
- да развият своите умения да декларират и използват локални и глобални променливи;
- да усъвършенстват своите умения за ползване на условен оператор в C#;
- да се запознаят с механизмите за преобразуване на текст в число и обратно;
- да разширят своите знания и умения относно използване на различни възможности и технологии за реализиране на диалог с потребителя по време на изпълнение на програмата.

На учениците се предлага следното **решение:**

1) **Първи етап:** Създаване на ГПИ на приложението;

При създаването на добър естетически издържан и максимално достъпен за работа графичен интерфейс на приложението е необходимо да бъдат съобразени следните основни моменти (РАХНЕВ, 2010):

- ✓ да бъдат групирани функционално обособените елементи на ГПИ – в случая това са радиобутоните, които дават възможност за избор на режим на въвеждане на информация в текстовото поле при изпълнение на приложението и радиобутоните за избор на действие за пресмятане.
- ✓ да бъдат подравнени всички елементи на ГПИ и групи от дизайна на приложението.
- ✓ да се изберат подходящи елементи на ГПИ за вход и изход – т.е механизма, по който потребителя взаимодейства с приложението – в случая се използват текстови полета за въвеждане на входните данни и етикет за визуализация на получените резултати.
- ✓ да бъдат скривани и визуализирани части от интерфейса на приложението само когато са необходими – в случая текстовото поле и вертикалната скролираща лента, чрез които се задава степента за пресмятане са видими само когато е избран опционалния бутон за пресмятане на степен;
- ✓ Когато е възможно да се задават начални подразбиращи стойности за някои елементи на ГПИ – в случая за текстовото поле и вертикалната скролираща лента, чрез които се задава стойността на степенния показател за пресмятане на степен е зададена подразбираща стойност 2.

2) **Втори етап:** Настройка на някои свойства на елементите на ГПИ в режим на проектиране (таблица № 1).

Таблица № 1

Елементи на ГПИ	Име на елемент	Свойства
Form	**Form1**	Text=”Задача”
Label	**label1**	Text=”Въведете стойност за пресмятане:”
Label	**label2**	Text=”Резултат:”
TextBox	**txtInput**	TextAlign=Right
Label	**lblResult**	Text=”” ; TextAlign= MiddleRight BorderStyle=FixedSingle ; BackColor=white
RadioButton	**ButtonKV**	Text=”Пресмятане на квадратен корен”
RadioButton	**ButtonStepen**	Text=” Пресмятане на степен”
TextBox	**TxtStepen**	Text=”2”; TextAlign= Right
VScrollBar	**vsb1**	Minimum=0; Maximum=100; Value=2; LargeChange=1; SmallChange=1;
GroupBox	**groupBox1**	Text=”Избор на действие”
GroupBox	**groupBox2**	Text=”Избор на режим за въвеждане на данни”
RadioButton	**radioButton1**	Text=”Режим на проверка при въвеждането”
RadioButton	**radioButton2**	Text=”Режим ограничаване на въвеждането”
Button	**cmdres**	Text=”Пресмятане на квадратен корен”
Button	**cmdNew**	Text=”Нови данни”
Button	**cmdExit**	Text=”Изход”

3) **Трети етап:** Добавяне на програмен код към елементите.

- ✓ Реализиране на действието на бутона ***Нови данни***.

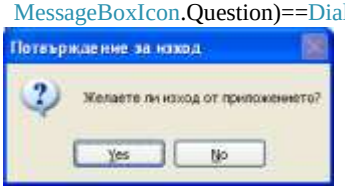
```
private void cmdNew_Click(object sender, EventArgs e)
   { txtInput.Text = ""; lblResult.Text = ""; txtInput.Focus(); p = 0; p1 = 0;}
```

- ✓ Реализиране на действието на бутона за пресмятане на квадратен корен или степен (т. е. с един и същи бутон ще се извършват пресмятанията, като в зависимост от действието, което е избрано в даден момент от изпълнение на задачата се изменя и надписа на бутона – ***Пресмятане на квадратен корен*** или ***Пресмятане на степен***).

```
private void cmdResult_Click(object sender, EventArgs e)
   {if (txtInput.Text == "")
      {MessageBox.Show("Не сте въвели нищо в текстовото поле!",
                       "Грешка при въвеждане");
       cmdNew_Click(sender, e);}
    else
    {bool flag = false; int br = 0; int m = 0;
    if (radioButton2.Checked) flag = true;
    if (radioButton1.Checked)
     {for (int i = 0; i < txtInput.Text.Length; i++)
     {if ((System.Char.IsDigit(txtInput.Text, i) == true) || (txtInput.Text[i] == ',')) br = br + 1;
      if (txtInput.Text[i] == ',') m = m + 1;}
     if (m <= 1 && ((br == txtInput.Text.Length) || ((br == txtInput.Text.Length - 1)
        && (txtInput.Text[0] == '-')))) flag = true;}
    if (flag == true)
     {if (ButtonKV.Checked == true)
      {double a; a = double.Parse(txtInput.Text);
       if (a >= 0) {a = Math.Sqrt(a); lblResult.Text = a.ToString();}
       else if (a < 0)
       {MessageBox.Show("Въвели сте отрицателно число!", "Грешка при въвеждане");
        cmdNew_Click(sender, e);}}
      if (ButtonStepen.Checked == true)
       {double x, a, n; x = double.Parse(txtInput.Text); n = double.Parse(txtStepen.Text);
        a = Math.Pow(x, n); lblResult.Text = a.ToString();}}
    else
      {MessageBox.Show("Не сте въвели числова стойност!", "Грешка при въвеждане");
       cmdNew_Click(sender, e);}}}
```

- ✓ Реализиране на действието на бутона ***Изход*** – с потвърждение за изход чрез диалогова кутия за съобщения (Фиг. 3).

```
private void cmdExit_Click(object sender, EventArgs e)
   {if (MessageBox.Show("Желаете ли изход от приложението?",
                        "Потвърждение за изход", MessageBoxButtons.YesNo,
                        MessageBoxIcon.Question)==DialogResult.Yes) Close();}
```



Фиг. 3

- ✓ Реализиране на събитийна процедура txtInput_KeyPress, която управлява въвеждането на данни в текстовото поле txtInput при избран

Режим ограничаване на въвеждането. При пресмятане на квадратен корен трябва да се ограничи въвеждането на всички други символи от клавиатурата, различни от цифрите от 0 до 9 и десетичната запетая, т.е. да могат да се въвеждат в текстовото поле само неотрицателни реални числа. При пресмятане на степен трябва да се ограничи въвеждането на всички други символи от клавиатурата, различни от цифрите от 0 до 9, десетичната запетая и знака „-”. Също така трябва да се съобрази да не се допуска повторно натискане на десетична запетая и знака „-”. За целта е необходимо да се декларират две глобални променливи p и p1, чрез които да се регулира броя на десетичната запетая и знака „-”. При всяко следващо въвеждане на нова стойност за пресмятане стойността на тези две променливи трябва да бъде нулирана.

```
int p = 0; int p1 = 0;
private void txtInput_KeyPress(object sender, KeyPressEventArgs e)
   {if ((radioButton2.Checked==true)&& (ButtonKV.Checked==true))
     {if (((e.KeyChar < (char)Keys.D0) || (e.KeyChar > (char)Keys.D9))
           && (!(e.KeyChar == (char)',')) && (!(e.KeyChar == (char)Keys.Back)))
        e.KeyChar = (char)0; }
    if ((radioButton2.Checked == true)&& (ButtonStepen.Checked==true))
     {if (((e.KeyChar < (char)Keys.D0) || (e.KeyChar > (char)Keys.D9))
           && (!(e.KeyChar == (char)',')) && !(e.KeyChar == (char)'-')
           && (!(e.KeyChar == (char)Keys.Back))) e.KeyChar = (char)0; }
     if (e.KeyChar == (char)',') p=p+1;
     if (e.KeyChar == (char)'-') p1 = p1 + 1;
     if ((e.KeyChar == (char)',')&&(p>1)) e.KeyChar = (char)0;
     if ((e.KeyChar == (char)'-') && (p1 > 1)) e.KeyChar = (char)0;}
```

- ✓ Реализиране на събитийна процедура Form1_Load за начална инициализация на формата на приложението.

```
private void Form1_Load(object sender, EventArgs e)
   {ButtonKV.Checked = true; int v; v=int.Parse(txtStepen.Text);
    vsb1.Value = v; radioButton1.Checked = true;}
```

- ✓ Реализиране на събитийна процедура vsb1_Scroll за коригиране на стойността на степенния показател.

```
private void vsb1_Scroll(object sender, ScrollEventArgs e)
   {int b; b=vsb1.Value; txtStepen.Text = b.ToString();}
```

- ✓ Реализиране на събитийна процедура txtStepen_KeyPress за директно коригиране на степенния показател от диапазона стойности на елемента vsb1 чрез натискане на клавиша Enter. Аналогично и тук допълнително може да бъде реализиран контрол на въвеждането, както при текстовото поле txtInput. По принцип в елемента txtStepen може да се въвеждат **произволни** цели или реални числа за стойност на степенния показател. Чрез елемента vsb1 могат да се въвеждат **последователно** само ограничен брой целочислени стойности за степенния показател в режим на изпълнение (т.е. в диапазона между минималната и максималната стойност на елемента vsb1). Тъй като в задачата и двата елемента txtStepen и vsb1 се използват за задаване на степенния показател, то

техните стойности е препоръчително да бъдат свързани, т.е. да бъдат едни и същи. С други думи, главното предназначение на текстовото поле txtStepen е предимно да визуализира текущата стойност на елемента vsb1. Затова при въвеждане на данни в полето txtStepen клавиша Enter може да се използва само в случаите, когато се въвежда число от диапазона стойности на елемента vsb1.

```
private void txtStepen_KeyPress(object sender, KeyPressEventArgs e)
   {if (e.KeyChar == (char)Keys.Return)
     {int a=int.Parse(txtStepen.Text);
      if ((a >= vsb1.Minimum) && (a <= vsb1.Maximum))
        {int p; p = int.Parse(txtStepen.Text); vsb1.Value = p;}
      else { int s = vsb1.Value; txtStepen.Text = s.ToString(); }}
```

- ✓ Реализиране на събитийните процедури ButtonKV_CheckedChanged и ButtonStepen_CheckedChanged - извършват настройка на надписа на бутона за пресмятане в зависимост от избраното действие и визуализация на елементите текстово поле за въвеждане на степенен показател и вертикална скролираща лента само в режим за пресмятане на степен.

```
private void ButtonKV_CheckedChanged(object sender, EventArgs e)
   {cmdResult.Text=ButtonKV.Text; txtStepen.Visible = false;
    vsb1.Visible = false; cmdNew_Click(sender,e);}
private void ButtonStepen_CheckedChanged(object sender, EventArgs e)
   {cmdResult.Text = ButtonStepen.Text; txtStepen.Visible = true;
    vsb1.Visible = true; cmdNew_Click(sender, e);}
```

На този етап от обучението по събитийно програмиране могат да бъдат разгледани и други аналогични задачи, реализиращи междупредметни връзки с обучението по математика. Например, в (ГЪРОВ, 2006) са предложени набор от опорни задачи по темата „Алгоритми и задачи от теория на числата”, които по подобен начин могат да бъдат реализирани със средствата на визуалното програмиране.

ЛИТЕРАТУРА

АНГЕЛОВ, И. & РАХНЕВ, А. (2010) Многоезичен уеб-базиран редактор за описание на семантично структурирани знания и информация, *Сборник доклади на Национална научна конференция „Образованието в информационното общество”*, Пловдив, 27-28.05.2010 г., 317-323.

АНЕВА, С. (2010) Система от базови задачи за изучаване на събитийно програмиране чрез средата Visual C# в средното училище, *Сборник доклади на Национална научна конференция „Образованието в информационното общество”*, Пловдив, 27-28.05.2010 г., 239-251.

ГРОЗДЕВ, С. & КЕНДЕРОВ, П. (2005) Инструментариум за откриване и подкрепа на изявени ученици по математика. *Сборник доклади на 34 конференция на СМБ, Математика и математическо образование*, Боровец, 6-9.04.2005 г., 53-64.

ГЪРОВ, К. & ТОДОРОВА, Е. (2006) Примерна система от опорни задачи по темата „Алгоритми и задачи от теория на числата" за подготовка на талантливи ученици по информатика. *Сборник доклади на 35 конференция на СМБ, Математика и математическо образование*, Боровец, 5-8.04.2006 г., 299-305.

РАХНЕВ, А. & СТОЕВА, М. (2010) Принципи и технологии за изграждане на потребителски интерфейс за уеб и десктоп приложения, *Сборник доклади на Национална научна конференция „Образованието в информационното общество*", Пловдив, 27-28.05.2010 г., 308-316.

SHOTLEKOV, I. & RAHNEV, A. (2010) Evaluating the Quality of Student Web Design Projects, *Proceedings of the 39th Spring Conference of the Union of Bulgarian Mathematicians, Mathematics And Education In Mathematics*, Albena, 6-10.04.2010, pp. 227-236.

VALCHANOV, N., TERZIEVA, T., SHKURTOV, V. & ILIEV, A. (2009) Approaches in building and supporting Business Information Systems, *Proceedings of the International Scientific Conference „Information technologies in management and business"*, Varna, 16-17.10.2009, pp. 100-106.

Стефка Йорданова Анева
ФМИ, ПУ „П. Хилендарски", Катедра „Компютърни технологии"
Пловдив 4003, бул. България №236, e-mail: stfaneva@uni-plovdiv.bg

THE ROLE OF BASIC PROBLEMS IN LEARNING OF EVENT-DRIVEN PROGRAMMING WITH VISUAL C# ENVIRONMENT IN HIGH SCHOOL

Stefka Aneva

ABSTRACT

The paper considers the role of basic problems in learning of event-driven programming with Visual C# environment in the specialized training in Informatics in high school. Some options in Visual C# and correct selection using the basic elements of GUI for the realization of user dialogue during the execution of an application are discussed.

Keywords: event-driven programming, visual programming, graphic user interface, element, event, event procedure.

Stefka Yordanova Aneva
236 Bulgaria Blvd., 4003 Plovdiv, Bulgaria
Faculty of Mathematics and Informatics, Paisii Hilendarski University of Plovdiv,
Department „Computer Technologies", e-mail: stfaneva@uni-plovdiv.bg