Číslo a název šablony	III/2 Inovace a zkvalitnění výuky prostřednictvím ICT
Číslo didaktického materiálu	EU-OPVK-VT-III/2-ŠR-114
Druh didaktického materiálu	DUM
Autor	RNDr. Václava Šrůtková
Jazyk	čeština
Téma sady didaktických materiálů	Programování v C# v příkladech I
Téma didaktického materiálu	Objekty a třídy
Vyučovací předmět	Seminář z informatiky
Cílová skupina (ročník)	Žáci ve věku 16–17 let
Úroveň žáků	začátečníci
Časový rozsah	1–2 vyučovací hodiny
Klíčová slova	Objekty, třídy, jmenné prostory, výčtový typ
Anotace	Studenti pracují s grafickými třídami C#, rozlišují jejich vlastnosti, metody a události. Učí se využívat dokumentace.
Použité zdroje	ELLER, Frank. *C# - začínáme programovat: podrobný průvodce začínajícího uživatele*. 1. vyd. Praha: Grada, 2002, 240 s. ISBN 80-247-0324-6. OCHRANOVÁ, Renata a Michal KOZUBEK. *Objektově orientované programování v Turbo Pascalu*. 1. vyd. Brno: Masarykova univerzita, 1993, 117 s. ISBN 80-210-0659-5. VYSTAVĚL, Radek. *Moderní programování: sbírka úloh k učebnici pro začátečníky*. 2. vyd. Ondřejov: moderníProgramování, 2008, 2 sv. ISBN 978-80-903951-5-2. VYSTAVĚL, Radek. *Moderní programování: učebnice pro začátečníky*. Ondřejov: moderníProgramování s.r.o, 2007, 2 sv. ISBN 978-80-903951-0-7.
Typy k metodickému postupu učitele, doporučené výukové metody, způsob hodnocení, typy k individualizované výuce apod.	Text je možno využít ke společné práci, samostatné přípravě studentů, domácímu studiu apod. Při společné práci je vhodné nejprve obtížnější úlohy rozebrat, potom společně se studenty implementovat na počítači. (Rozbor nejlépe na tabuli, synchronní řešení s promítáním) Prezentace obsahuje stručné shrnutí poznatků potřebných pro řešení příkladů. V pracovním listu je zadání cvičení – většinou se jedná o úlohy, které by měli studenti naprogramovat samostatně. Není nutné, aby všichni zpracovali všechno, vhodné je diferencovat podle jejich

	zájmu a schopností. Obtížnější úlohy jsou označeny hvězdičkou. Součástí materiálu je zdrojový kód těchto příkladů. Návrh způsobu hodnocení: ohodnocení samostatné práce během hodiny např. podle volby a počtu úloh a elaborace řešení (efektivnost, komentáře…).

Metodický list k didaktickému materiálu

Prohlášení autora

Tento materiál je originálním autorským dílem. K vytvoření tohoto didaktického materiálu nebyly použity žádné externí zdroje s výjimkou zdrojů citovaných v metodickém listu.

Obrázky (schémata a snímky obrazovek) pocházejí od autora.

114. Objekty a třídy

Každý objekt v programování zahrnuje: (napravo jsou uvedeny příklady)

Vlastnosti Panel1.Text, Panel1.BorderStyle, TextBox1.Text)

Události Panel1.Paint, Button1.Click

Metody Panel1.Refresh(), MessageBox.Show("…")

Třída je datovým typem objektů – třída textových políček – TextBox, třída tlačítek – Button apod. Třídy fungují jako šablony objektů – všechny objekty stejné třídy mají tytéž vlastnosti, události a metody. Objektům se také říká **instance třídy** – ve smyslu jednoho konkrétního výskytu.

Chceme-li se dozvědět více, můžeme pátrat v dokumentaci, nápovědě, diskusních fórech.

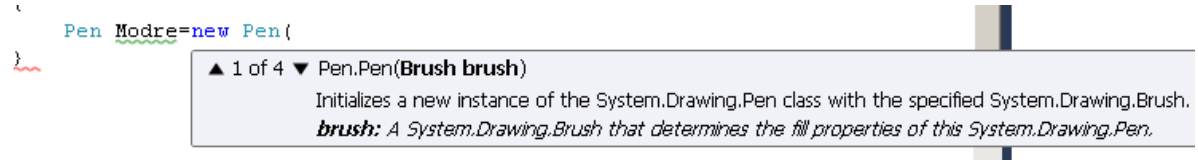
Pokud ve vývojovém prostředí zvolíme **Help/Manage Help Settings** a vybereme **Online Help,** volba **View Help** nás přesune na stránku **Visual Studia 2010**. Pak stačí do okna **MSDN Search** zadat např. *class pen* a pak si vybrat. Získáme informace a příklady ke konstruktorům, metodám, vlastnostem i událostem. (pero události nemá, ale tlačítko ano, vyzkoušejte si i Buttons)

Více o třídě Pen

Konstruktory

jsou metody, které vytvářejí nový objekt. v C# je jméno konstruktoru vždy shodné se jménem třídy. Třída může mít víc konstruktorů, které se liší svými parametry.

V okamžiku, kdy v programu zapíšeme vytvoření nového objektu voláním konstruktoru, našeptávač nabídne



4 možnosti. Vyzkoušejme některé.

1. z nich nabízí štětec – můžeme využít známé vlastnosti Brushes.barva

```
Pen Modre = new Pen(Brushes.Aqua);
```
(Ale Brush poskytuje možností víc)

```
Pen Rude = new Pen(Color.Red);
```
(jediným parametrem je barva)

Volání konstruktoru je vlastně příkazem pro vytvoření nové proměnné typu Pen s jedním parametrem – barvou, do něhož byla dosazena červená barva. Nové objekty se vždy vytvářejí s pomocí slova **new**.

Poslední možnost nabízí i 

tloušťku:

```
Pen SilneZelene = new Pen(Color.Green, 20);
```

Nabízí-li se více konstruktorů, vybíráme si ten nejvhodnější.

Vlastnosti

Při práci s konstruktory jsme si všimli, že důležité vlastnosti per budou barva a tloušťka. V programu je můžeme měnit přímo přiřazovacím příkazem:

```
Modre.Color = Color.GreenYellow;
            Modre.Width = 10;
            SilneZelene.Width = 50;
```

Všimněte si, že možnost volby vlastností není dána konstruktorem.

Jmenné prostory

Možná jste si všimli, jak vypadá začátek našeho zdrojového objektu, který vytváří vývojové prostředí.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Windows.Forms;


namespace Objekty
```

vlastně označuje začátek našeho program, kdybychom navrhovali vlastní třídu (pokračování tohoto kurzu), činili bychom to v dalším Namespace.

Třídy jsou organizovány ve jmenných prostorech, to, že některý používáte, dáváte najevo klauzulí **using** a jménem příslušného Namespace na začátku programu. Pokud píšeme WindowsFormApplication, jsou příslušné řádky do našeho projektu přidány automaticky.

System.Drawing například obsahuje třídy Graphics, Color, Pen a další související s grafikou, většina ovládacích prvků pro aplikace Windows patří do `System.Windows.Forms`. Kdyby klauzule Using na začátku chyběla, museli bychom používat názvy včetně příslušného Namespace – např. System.Drawing.Pen, System.Windows.Forms.textBox apod. Vyzkoušejte si to.

Statické a autogenerující složky třídy

Když vytvoříte nový objekt Pen Modre, můžete s jeho vlastnostmi zacházet pomocí zápisu Modre.Color apod.

Statické složky třídy se neváží na konkrétní objekt (v dokumentaci jsou značeny písmenem S) a jejich názvy se přesněji určují jménem třídy.

Statické metody jsou např. Convert.ToString, MessageBox.Show (nevytvářeli jsme žádný objekt typu Convert nebo MessageBox) Mezi statické vlastnosti patří třeba Brushes.Pink, která je vlastnost typu Brush.

Navíc existují **autogenerující složky** třídy, což jsou vlastnosti a metody, jejichž hodnotou je přímo objekt dané třídy, např. Color.Red definuje červenou barvu, patří sem také následující metoda: **FromArgb**, která míchá barvu ze složek RGB. (Naopak komponenty barev můžeme získat přes Barva.R, barva.G, barva.B)

Příklad 1

Vypíšeme si barevné složky nějaké barvy – třeba Orange a naopak namícháme barvu pomocí funkce FromArgb(R,G,B) kde R, G, B jsou celá čísla od 0 do 255. (podobně funguje funkce RGB na webu)

Přebarvovat panel barevným čtvercem budeme při události každého ze vstupních políček **TextChanged** (hlavní událost) – Panel1.Refresh() zase umístíme do společné obsluhy.

Při míchání barev je třeba zachytit výjimku, jinak bude program padat.

```
private void buttonOrange_Click(object sender, EventArgs e)
        {
            Color oran = Color.Orange;
            labelR.Text = "R: "+Convert.ToString(oran.R);
            labelG.Text = "G: " + Convert.ToString(oran.G);
```

```
        labelB.Text = "B: " + Convert.ToString(oran.B);
    }
    private void Prebarvi_Panel(object sender, EventArgs e)
    {
        panel1.Refresh();
    }
    private void panel1_Paint(object sender, PaintEventArgs e)
    {
        try
        {
            Graphics kp = e.Graphics;
            int R = Convert.ToInt32(textBoxR.Text);
            int G = Convert.ToInt32(textBoxG.Text);
            int B = Convert.ToInt32(textBoxB.Text);
            Color michana = Color.FromArgb(R, G, B);
            Pen pero = new Pen(michana);
            kp.DrawRectangle(pero, 10, 10, 80, 80);
        }
        catch
        {
        }
    }
```

Výčtový typ

Je-li datový typ definovaný výčtem, může daná vlastnost nabývat pouze hodnot uvedených v tomto výčtu. Patří sem např. styl čáry při kreslení perem – vlastnost **DashStyle** třídy pen.

Možné hodnoty:

Custom	programátorem definovaná
Dash	čárkovaná
DashDot	čerchovaná
DashDotDot	dvojitě čerchovaná
Dot	tečkovaná

Solid plná

Abyste ji ovšem mohli použít, je třeba k jmenným prostorům přidat:

```
using System.Drawing.Drawing2D;
```

Příklad 2

Na panelu vyznačte silnou tečkovanou (čtverečkovanou) úhlopříčku.

```
private void panel2_Paint(object sender, PaintEventArgs e)
        {
            Graphics kp = e.Graphics;
            Pen pero = new Pen(Color.Coral,10);
            pero.DashStyle = DashStyle.Dot;
            kp.DrawLine(pero, 0, 0, 100, 100);
        }
```

Důležité

Objekty obsahují vlastnosti, události a metody.

Třída zahrnuje více objektů téhož druhu.

Konstruktor je metoda, která vytváří nové instance (objekt) třídy. Před jeho voláním je slovo **new.**

Vlastnosti třídy Pen: Color, **Width, DashStyle**

Třídy se združují do jmenných prostorů (**Namespace**) Ty, které využívá náš program je třeba vyjmenovat na začátku, vždy po slově **using**

Ve třídách mohou být **statické složky,** které se neváží na konkrétní objekt. Mohou být také autogenerující – např. vlastnost Color.Blue určuje modrou barvu, zatímco metoda Color.FromArgb(R,G,B) vrací barvu namíchanou z jejich celočíselných parametrů.

Pracovní list

Cvičení

Vymyslete si vlastní příklad na bohatší kreslení s nastavováním vlastností per, buď něco sami namalujte nebo nechte vznikat obrázek na přání uživatele. Zkuste další možnosti kreslení – křivky, text.

Řešení

Pro inspiraci:

```
private void Form1_Paint(object sender, PaintEventArgs e)
        {
            Graphics kp = e.Graphics;
            //vytvoření obdélníku opsaného oblouku
            Rectangle ob =new Rectangle(10,10,100,100);
            //kreslení oblouku
            kp.DrawArc(Pens.Black, ob, 30, 150);
            //vytvoření fontu pro kreslení textu
            Font f=new Font("Arial", 15);
            //kreslení textu
            kp.DrawString("Barbarbar", f, Brushes.Red, 20, 20);

            // Kreslení textu malinko jinak - MSDN help
            String drawString = "Sample Text";
            // Create font and brush.
            Font drawFont = new Font("Arial", 16);
            SolidBrush drawBrush = new SolidBrush(Color.Black);
            // Create point for upper-left corner of drawing.
            PointF drawPoint = new PointF(150.0F, 150.0F);
            // Draw string to screen.
            e.Graphics.DrawString(drawString, drawFont, drawBrush, drawPoint);
            // Trojúhelník z bodů
             Point A = new Point (10,10);
             Point B = new Point (100,10);
             Point C = new Point (200,200);
             Point [] body=new Point[]{A, B, C};
             Plocha.FillPolygon(Brushes.Blue, body);
        }
...
private void Form1_Paint(object sender, PaintEventArgs e)
        {
```

```
        Pen Modre = new Pen(Brushes.Azure);
        Pen Zelene = new Pen(Color.Green);
        Pen RudeTluste = new Pen(Brushes.Red, 20);
        Pen Posledni = new Pen(Color.Black, 10);
        Graphics kp = e.Graphics;
        kp.DrawLine(Modre, 0, 0, 100, 100);
        kp.DrawLine(Zelene, 0, 100, 100, 0);
        kp.DrawLine(RudeTluste, 0, 50, 100, 50);
        kp.DrawLine(Posledni,50,0,50,100);
        Brush Muj = new SolidBrush(Color.BlueViolet);
        kp.FillEllipse(Muj, 100, 0, 100, 100);
        //obrázek jako výplň
        Bitmap obr = new Bitmap("obr.jpg");
        //obrázek uložen ve složce projektu /bin/debug
        Brush Pokus = new TextureBrush(obr);
        kp.FillEllipse(Pokus, 0, 100, 200, 200);
        Posledni.DashStyle = DashStyle.Dot;
        kp.DrawRectangle(Posledni, 110, 110, 200, 200);
    }
```