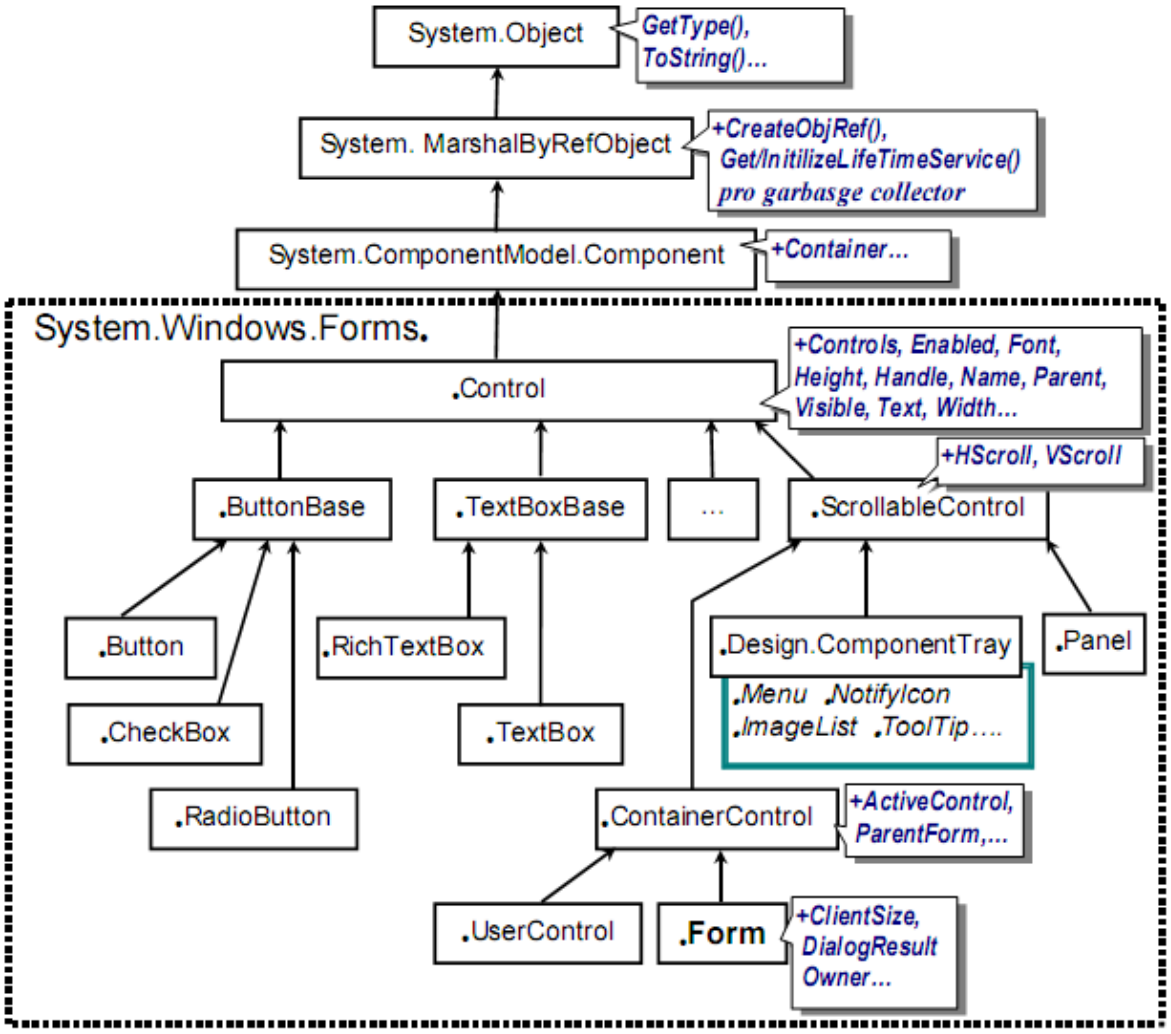
WinForms

- = formuláře Windows = knihovna ovládacích prvků Windows
- Jmenný prostor `System.Windows.Forms`

Hierarchie tříd



- `Form`
  - Tvorba a zobrazení oken
  - = jakékoli okno GUI aplikace (okno, dialog, panel nástrojů…)
- `Control`
  - Základ pro všechna okna
  - Specializace = jedn. ovládací prvky (tlačítka, …)

Úvod do programování Winforms

Nejmenší WinForms aplikace

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

```
namespace WinForms_Hello2
{
	class Program
	{
		[STAThread]
		static void Main(string[] args)
		{
			Form form = new Form();
			Application.Run(form);
		}
	}
}
```

  - Pozn.: `form` lze zobrazit i jinak – viz dále dialogy (`form.Show()`, `form.ShowModal()`)

- Výsledek:

- Porovnání s WinAPI aplikací:
  - `public static void Run(Form mainForm)`
    - ukrývá smyčku zpráv, kterou spravuje pro `form`
    - konec `form` = konec `Run()` = konec aplikace
  - `Form` obsahuje `protected override void WndProc(ref Message m)`
    - zděděno od `ContainerControl`
    - = procedura okna, pokud nutné – lze předefinovat a obsluhovat zprávy ručně
  - Drtivá většina zpráv – zabalena do Událostí C#
  - Ovládání vzhledu – Vlastnosti a metody `Form`

Události ve WinForms

- Prakticky všechny zprávy Windows → události (= všechny ovládací prvky mají události při změně stavu uživatelem)
- Tvary delegátů:
  - Většina událostí: `delegate void EventHandler(object sender, EventArgs e)`
  - Některé události specializované třídy argumentů, např. kreslení `PaintEventArgs`, myš `MouseEventArgs`, …
- Není nutné na všechny události reagovat, jen pokud chceme

Vlastní třídy formulářů, ovládací prvky a události

- Reálná aplikace = vlastní třída formuláře (odvozena od `Form`)
  - Ovládací prvky = přidané členské proměnné

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

```
using System.Text;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Windows.Forms;

namespace WinForms_Hello2
{
	class Program
	{
		[STAThread]
		static void Main(string[] args)
		{
			Formular form = new Formular();
			Application.Run(form);
		}
	}

	class Formular : Form
	{
		Button tlacitko;

		public Formular()
		{
			// nastaveni vlastností formu
			this.Text = "Aplikace";
			// pridani tlacitka
			tlacitko = new Button();		// nove tlacitko
			tlacitko.Text = "Tlačítko";	// popisek tlacitka
			tlacitko.Location = new Point(50, 100);
				// pozice na formu
			tlacitko.Click += new EventHandler(tlacitko_Click);
				// registrace udalosti
			this.Controls.Add(tlacitko);
				// musi byt, jinak ovl. prvek nebude videt
		}

		// Obsluha udalosti - klik na tlacitko
		void tlacitko_Click(object sender, EventArgs e)
		{
			// MessageBox.Show("Ahoj");	// zakl. message box
			MessageBox.Show("Ahoj", "Zpráva", MessageBoxButtons.OK,
				MessageBoxIcon.Information);
				// dokonaly message box
		}
	}
}
```

Start aplikace

- Postup:
  - `Form()` – vytvoření formuláře, aktualizace jeho dat
  - `Form.Load` – těsně před zobrazením
  - `Form.Activated` – okno získalo fokus
  - `Form.Paint` – požadavek na překreslení obsahu

Viditelnost a vzhled okna

- stav okna: `enum Form.WindowState { Minimized, Maximized, Normal}`

- Minimalizace okna – události
  - `Move` → `Layout` → `Resize` → `SizeChanged` → `Deactivate`
- Maximalizace okna
  - `Activated` → `Move` → `Layout` → `Resize` → `SizeChanged` → `Paint`

Ukončení aplikace

- Ruční – zavřením formuláře
- Programově:
  - `Application.Exit()` – Zavře všechny formuláře aplikace, ukončí smyčku zpráv
  - `Form.Close()` – uzavře formulář
- události při ukončení
  - `Form.FormClosing` – po požadavku na uzavření formuláře (dříve `Form.Closing`)
  - `Form.FormClosed` – po uzavření formuláře (dříve `Form.Closed`)
  - `Form.Deactivate`
- Stornování ukončení:

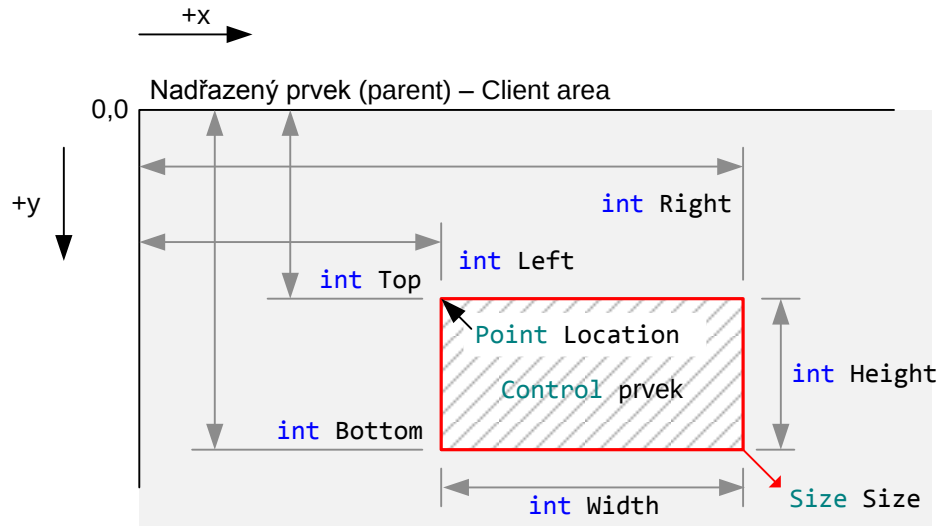
```
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
	if (MessageBox.Show("Opravdu si přejete skončit?", "",
		MessageBoxButtons.YesNo) == DialogResult.No)
	{
		e.Cancel = true;	// stornovani udalosti (ukonceni programu)
	}
}
```

Třída `Control` – Vlastnosti

- Jen výběr, vše viz MSDN

Pozice a rozměry

- Vše Get i Set
- Pozice – v rámci client area rodičovského prvku (property `Controls`)
- Rozměry – v pixelech (px)



Kotvení a dokování

- Vše Get i Set
- Co se děje při změně velikosti a proporcí hostitelského prvku
- `AnchorStyles Anchor` – změna rozměrů prvku při změně rozměrů rodiče
- `DockStyle Dock` – změna pozice prvku při změně rozměrů rodiče

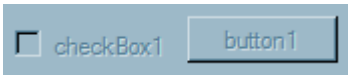
Hierarchie prvků

- Vztah vlastník (owner) – podřízené prvky
-

Vzhled

- Vše Get i Set
- `string Text` – textový popisek prvku

- `Color BackColor` – barva pozadí
- `Color ForeColor` – barva popředí
- `bool Enabled` – možnost přenést na prvek fokus



- `bool Visible` – viditelnost prvku v aplikaci
- `Rectangle ClientRectangle` – client area prvku
- `Size ClientSize` – client area prvku

Ovládání

- Vše Get i Set
- `ContextMenuStrip ContextMenuStrip` – kontextové menu prvku
- `int TabIndex` – pořadí prvku v kontejneru (rodičovský prvek) pro přepínání fokusu tabelátorem

- `bool TabStop` – může-li prvek získat fokus klávesou TAB.

Design

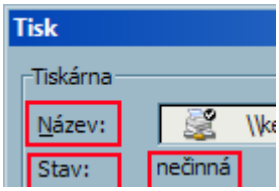
- `string Name` – **jen při návrhu GUI v návrháři VS**; podle tohoto řetězce vytvoří VS studio jméno proměnné prvku v automaticky generovaném kódu.

Standardní ovládací prvky

- Pozn.: Vlastnosti uvedené dále jsou pouze ty navíc oproti `Control`
- Události a vlastnosti uvedeny pouze ty nejpoužívanější

`Label`

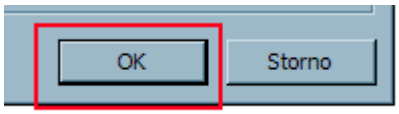
- Textový popisek (např. co kam zadat)



- Vlastnosti
  - `TextAlign` – zarovnání textu, výčet typu `ContentAlignment` (např. vlevo a dolů)

`Button`

- Standardní tlačítko



- Události:
  - `Click` – odpálena při:
    - Kliku na tlačítko levým tl.myši
    - ENTER na tlačítku, které má focus