

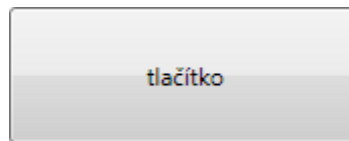
# 3. WPF - Šablony & Trigry

---

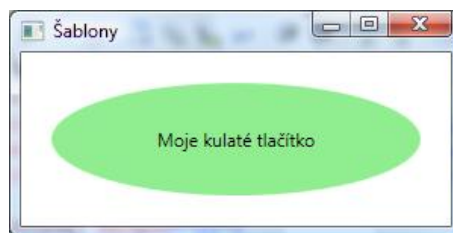
## Templates neboli Šablony

Zatímco styly slouží pouze ke kosmetickým úpravám prvků v aplikaci, pomocí vlastnosti *Template* můžeme vzhled celé kontroly úplně přepsat.

Podívejme se na standardní WPF tlačítko.



Skládá se z textového pole a obdelníku. U tlačítka můžeme změnit barvu pozadí nebo velikost písma, protože tlačítko má tyto vlastnosti *FontSize* a *Background*. Pokud ale chceme, aby tlačítko mělo jiný vzhled (například aby bylo kulaté), nezbyvá nám nic jiného než přepsat šablonu (*Button.Template*) této kontroly, neboli znovu nadefinovat z jakých prvků se má skládat.



```
<Button>
  <Button.Template>
    <ControlTemplate>
      <Grid>
        <Ellipse Fill="LightGreen" />
        <Label VerticalAlignment="Center"
              HorizontalAlignment="Center"
              Content="Moje kulaté tlačítko" />
      </Grid>
    </ControlTemplate>
  </Button.Template>
</Button>
```

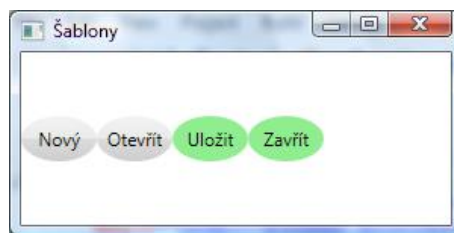
Povinný element je *ControlTemplate*, do kterého vložíme prvky, ze kterých se má naše kontrola nově skládat.

\* Obsah uvedený mezi tagy odpovídá vždy některé z vlastností dané kontroly. Například u Labelu jsou tedy tyto zápisy ekvivalentní

```
<Label>můj text</Label>  
a  
<Label Content="můj text" />
```

## Šablony ve stylech

Ve stylech pracujeme s vlastností *Template* stejně jako s jinými vlastnostmi. Podívejme se na jednoduchý příklad, ve kterém pomocí stylu přepíšeme šablonu všech tlačítek.



```
<StackPanel Orientation="Horizontal">  
  <StackPanel.Resources>  
    <!-- Obecný styl -->  
    <Style TargetType="Button">  
      <Setter Property="Template">  
        <Setter.Value>  
          <ControlTemplate>  
            <Grid Width="50" Height="30">  
              <Ellipse Fill="{TemplateBinding Button.Background}" />  
              <Label Content="{TemplateBinding Button.Content}"  
                HorizontalAlignment="Center"  
                VerticalAlignment="Center" />  
            </Grid>  
          </ControlTemplate>  
        </Setter.Value>  
      </Setter>  
    </Style>  
  </StackPanel.Resources>  
  <Button Content="Nový" />  
  <Button Content="Otevřít" />  
  <Button Content="Uložit" Background="LightGreen" />  
  <Button Content="Zavřít" Background="LightGreen" />  
</StackPanel>
```

Nejspíš jste si všimli zápisu hodnot *Ellipse.Fill* a *Label.Content*. To nám umožní, aby hodnoty těchto vlastností bylo možné zadávat přímo u tlačítka.

Pomocí *TemplateBinding* tedy pracujeme uvnitř šablony s hodnotami zadanými u kontroly. V tomto případě jsme použili barvu zadanou ve vlastnosti *Background* tlačítka na vyplnění elipsy.

```

<Ellipse Fill="{TemplateBinding Button.Background}" />
...
<Button Content="Uložit" Background="LightGreen" />

```

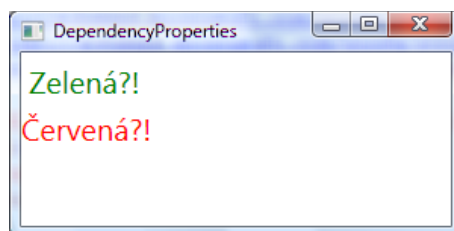
Obdobným způsobem je načten i text.

\* Všiměte si, že u tlačítek, u kterých není vlastnost *Background* zadaná, se použila hodnota ze standardního vzhledu.

## Dependency properties

Jedná se o speciální vlastnosti (*properties*) kontrol. Čím jsou zvláštní? Hodnota takovéto vlastnosti je závislá na vlastnostech jiných kontrol. Jak to přesně funguje?

*Příklad:*



```

<Border TextElement.Foreground="Red" TextElement.FontSize="20">
  <StackPanel>
    <Label Foreground="Green">
      <TextBlock>Zelená?!</TextBlock>
    </Label>
    <TextBlock>Červená?!</TextBlock>
  </StackPanel>
</Border>

```

Co je na této aplikaci zajímavé? Ačkoliv *TextBlocky* nemají nastavenou barvu textu (*Foreground*), není barva textu standardní. Jedná se totiž o *Dependency properties* a tak hodnoty těchto vlastností jsou převzaty z nejbližšího nadřazeného elementu.

*TextBlock* s červeným textem tedy přebíral barvu z elementu *StackPanel*, který tuto hodnotu přebíral z elementu *Border*.

## Co jsou Trigry?

Trigry (anglicky *Triggers*) nám v XAMLu umožní reagovat na události kontrol, na hodnoty vlastností (property) kontrol nebo na datové hodnoty. Podle toho taky rozlišujeme trigry na:

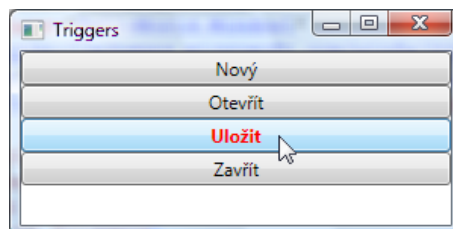
- *Property triggers* - reagují na hodnotu vlastnosti (property) kontroly
- *Event triggers* - reagují na události
- *Data triggers* - reagují na hodnotu proměné

Trigry se používají nejčastěji ve spojení se styly. Podívejme se na použití všech tří typů ve stylech.

### Property trigger

Property trigry reagují pouze na dependency properties (vlastnosti) kontrol. V property trigrech můžeme nastavit hodnoty vlastností pomocí elementu *Setter* a nebo pracovat s animacemi. Jakmile podmínka trigru neplatí, vrátí se hodnoty zpět do původního stavu.

*Příklad:* Chceme, aby po najetí myší nad tlačítko byl text červený a zvýrazněný. Použijeme vlastnost *IsMouseOver* tlačítka, která je *True* pokud je myš nad tlačítkem, v opačném případě je *False*.



```
<StackPanel>
  <StackPanel.Resources>
    <Style TargetType="Button">
      <Style.Triggers>
        <Trigger Property="Button.IsMouseOver" Value="True">
          <Setter Property="Button.FontWeight" Value="Bold" />
          <Setter Property="Button.Foreground" Value="Red" />
        </Trigger>
      </Style.Triggers>
    </Style>
  </StackPanel.Resources>
  <Button>Nový</Button>
  <Button>Otevřít</Button>
  <Button>Uložit</Button>
  <Button>Zavřít</Button>
</StackPanel>
```

*Setter* se tedy vykoná pouze pokud hodnota *IsMouseOver* je *True*. Pokud myši z tlačítka odjedeme, tlačítko se vrátí do původního stavu.

\* Před názvy vlastností je lepší zadávat typ kontroly

## Multi trigry

MultiTrigry používáme pokud u property trigru potřebujeme zadat více podmínek.



```
<StackPanel>
  <StackPanel.Resources>
    <Style TargetType="Button">
      <Style.Triggers>
        <MultiTrigger>
          <!-- Podmínky -->
          <MultiTrigger.Conditions>
            <Condition Property="Button.IsMouseOver" Value="True" />
            <Condition Property="Button.Content" Value="Zavřít" />
          </MultiTrigger.Conditions>
          <!-- Setter -->
          <Setter Property="Button.FontWeight" Value="Bold" />
          <Setter Property="Button.Foreground" Value="Red" />
        </MultiTrigger>
      </Style.Triggers>
    </Style>
  </StackPanel.Resources>
  <Button>Nový</Button>
  <Button>Otevřít</Button>
  <Button>Uložit</Button>
  <Button>Zavřít</Button>
</StackPanel>
```

Nyní jsou elementy *Setter* použity pouze na tlačítko s textem "Zavřít" pokud je kurzor myši nad tlačítkem.

## Event trigger

V event trigrech můžeme pracovat pouze s animacemi. Ty probereme až v následujícím díle, proto implementaci animací v následující ukázce můžete vynechat.

*Příklad:* Po najetí myši nad tlačítko se výška tlačítka změní z původní hodnoty 20px na 30px. Jakmile tlačítko myši opustíme, vrátí se zpět do původní velikosti. Změny velikostí trvají 0.5 sekundy.

```
<StackPanel>
  <StackPanel.Resources>
    <Style TargetType="Button">
      <Setter Property="Button.Height" Value="20" />
      <Style.Triggers>
        <!-- Roztáhnout tlačítko -->
        <EventTrigger RoutedEvent="Button.MouseEnter">
          <EventTrigger.Actions>
            <BeginStoryboard>
              <Storyboard>
                <DoubleAnimation To="30"
                                   Storyboard.TargetProperty="Height"
                                   Duration="0:0:0.5" />
              </Storyboard>
            </BeginStoryboard>
          </EventTrigger.Actions>
        </EventTrigger>
        <!-- Zúžit na původní výšku -->
        <EventTrigger RoutedEvent="MouseLeave">
          <EventTrigger.Actions>
            <BeginStoryboard>
              <Storyboard>
                <DoubleAnimation To="20"
                                   Storyboard.TargetProperty="Height"
                                   Duration="0:0:0.5" />
              </Storyboard>
            </BeginStoryboard>
          </EventTrigger.Actions>
        </EventTrigger>
      </Style.Triggers>
    </Style>
  </StackPanel.Resources>
  <Button>Nový</Button>
  <Button>Otevřít</Button>
  <Button>Uložit</Button>
  <Button>Zavřít</Button>
</StackPanel>
```

## Data trigry (používáme zřídka)

Property trigry můžeme použít pouze na vlastnosti (dependency property) kontrol, Data trigry nám zase umožní reagovat na hodnotu uživatelských proměných. Na použití Data trigrů se podíváme v kapitole DataBinding.

## Trigry bez stylů?

Proč zde uvádím používání trigrů pouze ve spojení se styly? Proč bychom nemohli používat trigry přímo v kontrolách?

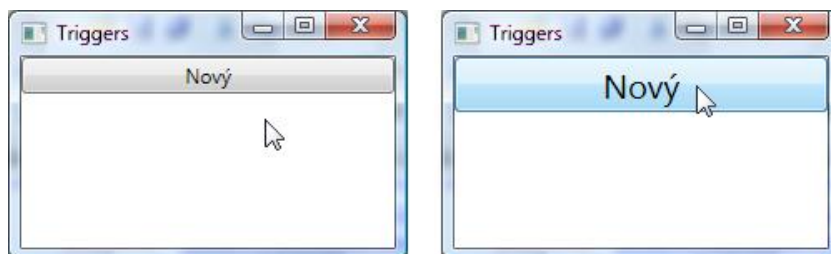
IntelliSense nám při psaní kódu nabídne tento způsob zápisu:

```
<Button>
  <Button.Triggers>
    <Trigger Property="Button.IsMouseOver" Value="True">
      <Setter Property="Button.FontSize" Value="20" />
    </Trigger>
  </Button.Triggers>
</Button>
```

*^^ tento kód ale vyvolá chybu při běhu aplikace*

Bohužel vývojáři .Net Framework 3.0 z časových důvodů nestihli implementovat Data trigry a Property trigry u kontrol (fungují takto pouze Event trigry) a tak jediná možnost jak tento nedostatek obejít je použít styly.

*Ukázka použití Property trigru u tlačítka pomocí stylů*



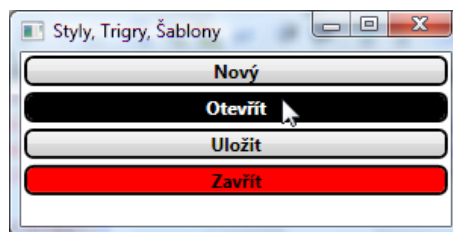
```
<StackPanel>
  <Button Content="Nový">
    <Button.Style>
      <Style>
        <Style.Triggers>
          <Trigger Property="Button.IsMouseOver" Value="True">
            <Setter Property="Button.FontSize" Value="20" />
          </Trigger>
        </Style.Triggers>
      </Style>
    </Button.Style>
  </Button>
</StackPanel>
```

Tabulka kdy a kde je možné jednotlivé trigry používat

	použití ve stylech		použití v kontrole	
	Setter	Animace	Setter	Animace
Property trigger	ano	ano	ne	ne
Event trigger	ne	ano	ne	ano
Data trigger	ano	ano	ne	ne

## Demo

V následujícím demu spojíme dohromady styly, trigry a šablony a vytvoříme si vlastní vzhled tlačítek v aplikaci.



```

<StackPanel>
  <StackPanel.Resources>
    <Style TargetType="Button">
      <!-- Šablona -->
      <Setter Property="Button.Template">
        <Setter.Value>
          <ControlTemplate>
            <Border x:Name="myBorder" BorderBrush="Black"
              BorderThickness="2"
              CornerRadius="5"
              Margin="2"
              Background="{TemplateBinding Button.Background}">
              <ContentControl Content="{TemplateBinding Button.Content}"
                FontWeight="Bold"
                HorizontalAlignment="Center" />
            </Border>
            <!-- Trigry -->
            <ControlTemplate.Triggers>
              <Trigger Property="Button.IsMouseOver" Value="True">
                <Setter Property="Border.Background" Value="Black"
                  TargetName="myBorder" />
                <Setter Property="Button.Foreground" Value="White" />
              </Trigger>
            </ControlTemplate.Triggers>
          </ControlTemplate>
        </Setter.Value>
      </Setter>
    </Style>
  </StackPanel.Resources>
  <Button Content="Nový" />
  <Button Content="Otevřít" />
  <Button Content="Uložit" />
  <Button Content="Zavřít" />
</StackPanel>

```

šablona

trigry

```

        </Setter.Value>
    </Setter>
</Style>
</StackPanel.Resources>
<Button Content="Nový" />
<Button Content="Otevřít" />
<Button Content="Uložit" />
<Button Content="Zavřít" Background="Red" />
</StackPanel>

```

1. použili jsme obecný (nepojmenovaný) styl, takže tento vzhled je použit na každé tlačítko ve *StackPanelu*
2. přepsali jsme šablonu tlačítka, tzn. museli jsme znovu nadefinovat prvky tlačítka a přidali efekty
3. obsah tlačítka nemusí být pouze text, ale například i obrázek nebo jiný element, proto pro vypsání *Contentu* jsme použili element *ContentControl*; obsah načítáme (*TemplateBinding*) z vlastnosti *Content* tlačítka
4. použili jsme *Property trigger* - po najetí na tlačítko se změní barva textu a pozadí tlačítka
  - *TargetName* stanoví prvek v šabloně, na který má být element *Setter* použit; u *Button.Foreground* nebylo potřeba *TargetName* použít, protože se jedná o *Dependency property* takže elementy uvnitř šablony převezmou hodnotu z nadřazených elementů

## Závěr

Díky šablonám máme kontroly ve WPF úplně pod kontrolou, pokud se nám nelíbí standartní vzhled, můžeme si navolit vlastní a přitom zároveň zachovat funkčnost kontroly. Trigry nám zase umožní vytvářet efekty a reagovat na události nebo vlastnosti kontrol.

Více informací o WPF naleznete na [www.unitedstatesof.net/wpf](http://www.unitedstatesof.net/wpf)

Aleš Šturala, 25. 1. 2007

<http://hidentity.org/hid/CZ123456>