# Triggering an Animation in another Control

Christian Wenz

## Overview

The Animation control in the ASP.NET AJAX Control Toolkit is not just a control but a whole framework to add animations to a control. Generally, launching an animation is triggered by user interaction with the same control. It is however also possible to interact with one control and then animation another control.

## Steps

First of all, include the **ScriptManager** in the page; then, the ASP.NET AJAX library is loaded, making it possible to use the Control Toolkit:

```
<asp:ScriptManager ID="asm" runat="server" />
```

The animation will be applied to a panel of text which looks like this:

```
<asp:Panel ID="panelShadow" runat="server" CssClass="panelClass">
  ASP.NET AJAX is a free framework for quickly creating a new
   generation of more efficient,
  more interactive and highly-personalized Web experiences that
   work across all the
  most popular browsers.<br />
  ASP.NET AJAX is a free framework for quickly creating a new
   generation of more efficient,
  more interactive and highly-personalized Web experiences that
   work across all the
  most popular browsers.<br />
  ASP.NET AJAX is a free framework for quickly creating a new
   generation of more efficient,
  more interactive and highly-personalized Web experiences that
   work across all the
  most popular browsers.<br />
</asp:Panel>
```

In the associated CSS class for the panel, define a nice background color and also set a fixed width for the panel:

```
<style type="text/css">
  .panelClass {background-color: lime; width: 300px;}
</style>
```

In order to start animating the panel, an HTML button is used. Note that **<input type="button" />** is favoured over **<asp:Button />** since we do not want a postback when the user clicks on that button.

```
<input type="button" id="Button1" runat="server" Value="Launch
   Animation" />
```

Then, add the **AnimationExtender** to the page, providing an **ID**, the **TargetControlID** attribute and the obligatory **runat="server"**. It is important to set **TargetControlID** to the ID of the button (the element triggering the animation), not to the ID of the panel (the element being animated)

```
<ajaxToolkit:AnimationExtender ID="ae" runat="server"
   TargetControlID="Button1">
```

Within the **<Animations>** node, place animations as usual. In order to make them change the panel, not the button, set the **AnimationTarget** attribute for every animation element within **AnimationExtender**. The value for **AnimationTarget** is the ID of the panel, of course. That way, the animations happen with the panel, not with the triggering button. Here is the **AnimationExtender** markup for this scenario:

```
<ajaxToolkit:AnimationExtender ID="ae" runat="server"
   TargetControlID="Panel1">
   <Animations>
     <OnClick>
       <Sequence>
         <EnableAction Enabled="false" />
         <Parallel>
           <FadeOut Duration="1.5" Fps="24"
  AnimationTarget="Panel1" />
           <Resize Width="1000" Height="150" Unit="px"
  AnimationTarget="Panel1" />
         </Parallel>
       </Sequence>
     </OnClick>
   </Animations>
</ajaxToolkit:AnimationExtender>
```

Note the special order in which the individual animations appear. First of all, the button gets deactivated once the animation runs. Since there is no **AnimationTarget** attribute in the **<EnableAction>** element, this animation is applied to the originating control: the button. The next two animation steps shall be carried out parallelly (**<Parallel>** element). Both have their **AnimationTarget** attributes set to **"Panel1"**, thus animating the panel, not the button.
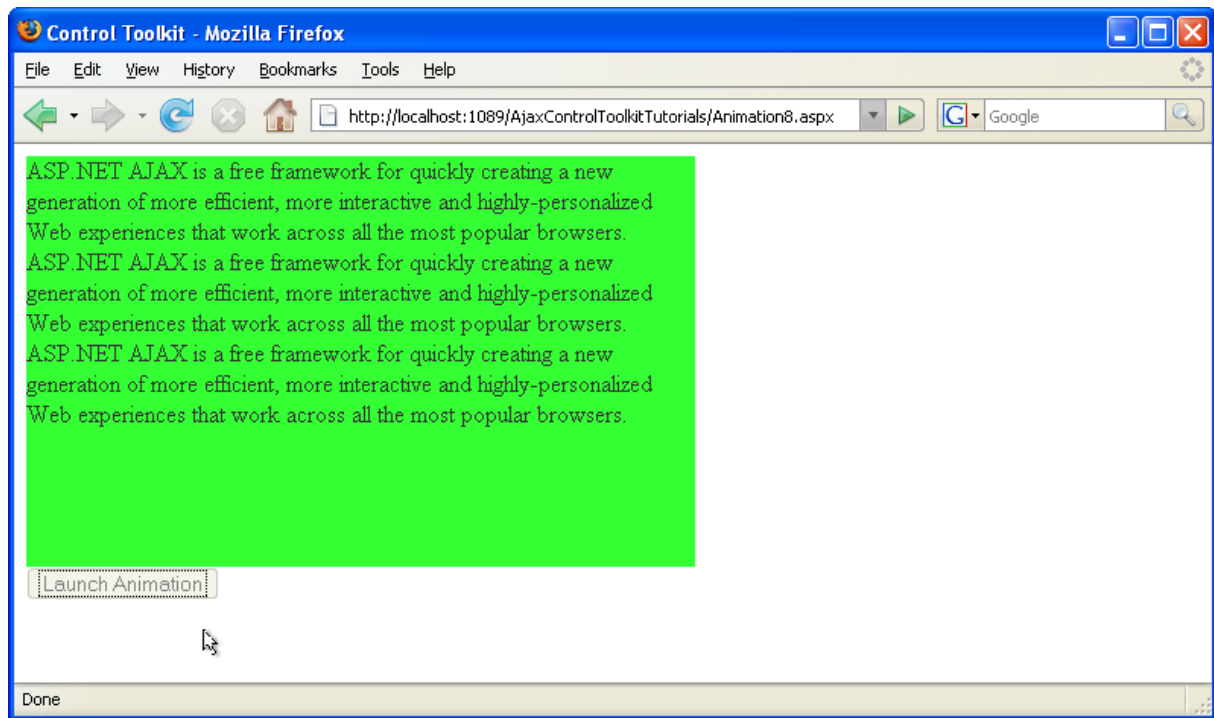
**A mouse click on the button starts the panel animation**