SPRAWOZDANIE

Zaawansowane aplikacje WWW

Technologia AJAX

09.12.2015

Karol Suwalski 125NCI B

https://github.com/SuwalskiKarol/orw.git

Wstęp

.NET AJAX jest zbiorem rozszerzeń, które umożliwiają wykorzystanie AJAX w aplikacjach ASP.NET. Zalety ASP.NET AJAX to:

- Odciążenie serwera na rzecz przetwarzania po stronie klienta.
- Możliwość wykorzystania elementów interfejsu użytkownika znanych z aplikacji Windows, np. wskaźniki postępu.
- Aktualizacja jedynie tych części strony, które tego wymagają.
- Integracja danych z różnych źródeł za pomocą odwołań do usług Web
- Wsparcie dla większości popularnych i używanych przeglądarek.

W ASP.NET możemy zaimplementować AJAXA do naszej aplikacji używając:

- **XMLHTTPRequest object in JavaScript** – standard W3C. Najpopularniejsza forma używania AJAXA.Wiadomo, o co chodzi.
- **jQuery AJAX** – jQuery jest biblioteką, która ułatwia nam korzystanie z JavaScriptu. W tym przypadku ułatwia nam prace przy używaniu XMLHTTPRequest.
- **Microsoft AJAX library**– JavaScriptowyframeworkrozbudowujący oraz ułatwiający pracę z AJAXEM.
- **AJAX Control Toolkit** - kolekcja kontrolek, które włączają do naszej aplikacji funkcjonalności AJAXA.
- **ASP.NET AJAX Server controls** – kontrolki te pozwalają umieszczać dane dotyczące wyglądu przeglądarki i funkcjonalności serwera w spójnych obiektach wielokrotnego użytku.

AJAX Extensions

Są to podstawowe kontrolki dostępne w ASP.NET bez potrzeby ich instalacji.

ScriptManager

Koordynuje i rejestruje skrypty odpowiedzialne za częściowe odświeżanie strony, odpowiada za konfigurowanie, zwolnienie bądź debugowanie skryptów wysyłanych do przeglądarki oraz interakcję skryptów z metodami usług sieciowych.

```
<asp:ScriptManagerID="ScriptManager1"runat="server"EnablePageMethods="true">
```

UpdatePanel

Umożliwia aktualizację wybranej części strony przy użyciu asynchronicznych żądao. Jest kluczowymelementem AJAX w ASP.NET.

```
<asp:UpdatePanelID="UpdatePanel2"runat="server">
<ContentTemplate>
       <asp:TextBoxID="TextBox2"runat="server">tekst</asp:TextBox>
              <asp:ButtonID="Button1"runat="server"OnClick="Button1_Click"Text="Oblicz"
              OnClientClick="CallServerAdd(); return false;"/>
</ContentTemplate>
```

```
</asp:UpdatePanel>
```

ContentTemplate – jest kontenerem dla kontrolek, które mają być dynamicznie aktualizowane. Dodawanie kontrolek jest możliwe tylko do tej części.

UpdateProgress

Kontrolka UpdateProgress udostępnia informacje o statusie aktualizacja kontrolki UpdatePanel. Kontrolka wyświetla zawartość określoną przez właściwość ProgressTemplate. W momencie aktualizacji kontrolki UpdatePanel pojawia się zawartość zdefiniowana w kontrolce UpdateProgress.

```
<asp:UpdateProgressID="UpdateProgress1"runat="server">
<ProgressTemplate>
Proszęczekać...
</ProgressTemplate>

</asp:UpdateProgress>
```

Timer

Kontrolka Timer odpowiada za komunikację zwrotną do serwera w określonych odstępach czasu. Możliwe jest wysyłanie całej strony lub jej części ujętej w kontrolce UpdatePanel. Właściwość Interval określa ilość milisekund, po której nastąpi komunikacja zwrotna do serwera.

```
<asp:TimerID="Timer1"runat="server"Interval="5000"OnTick="Timer1_Tick">
```

ASP.NET AJAX Server Control Extender

Jest rozszerzeniem standardowego ASP.NET Web server controls.Oferuje mnóstwo gotowych, zarówno wyjątkowo prostych, jak i złożonych kontrolek serweraobejmujących funkcjonalność frameworka AJAX. Co więcej z jego pomocą można również tworzyć własne kontrolki, posiadające funkcjonalności, których nie zaimplementowano w kontrolkach już istniejących.

Przykład pliku odpowiadającego za kontrole serwera ServerControl.cs

```
using System;
...
usingSystem.Xml.Linq;

namespace AjaxServerControl1
 {
publicclassServerControl1 : ScriptControl
     {
public ServerControl1()
         {
//
// TODO: Add constructor logic here
//
         }


protectedoverrideIEnumerable<ScriptDescriptor>GetScriptDescriptors()
```

```
        {
ScriptControlDescriptor descriptor =
newScriptControlDescriptor("AjaxServerControl1.ClientControl1", this.ClientID);
yieldreturn descriptor;
        }

// Generate the script reference
protectedoverrideIEnumerable<ScriptReference>GetScriptReferences()
       {
yieldreturnnewScriptReference("AjaxServerControl1.ClientControl1.js",
this.GetType().Assembly.FullName);
}
     }

 }
```

GetScriptReferences()–Zwraca obiekty ScriptReference. Obiekty te zawierają informacje o skryptach zapewniających funkcjonalność po stronie klienta(np. IEnumerableinterface).

GetScriptDescriptors() - Zwraca obiekty ScriptDescriptor. Obiekty te zawierają informacje o skryptach klienta, które wykorzystywane są przy kontroli serwera WWW.

Przykład pliku odpowiadającego za kontrole po stronie klienta ClientControl1.js

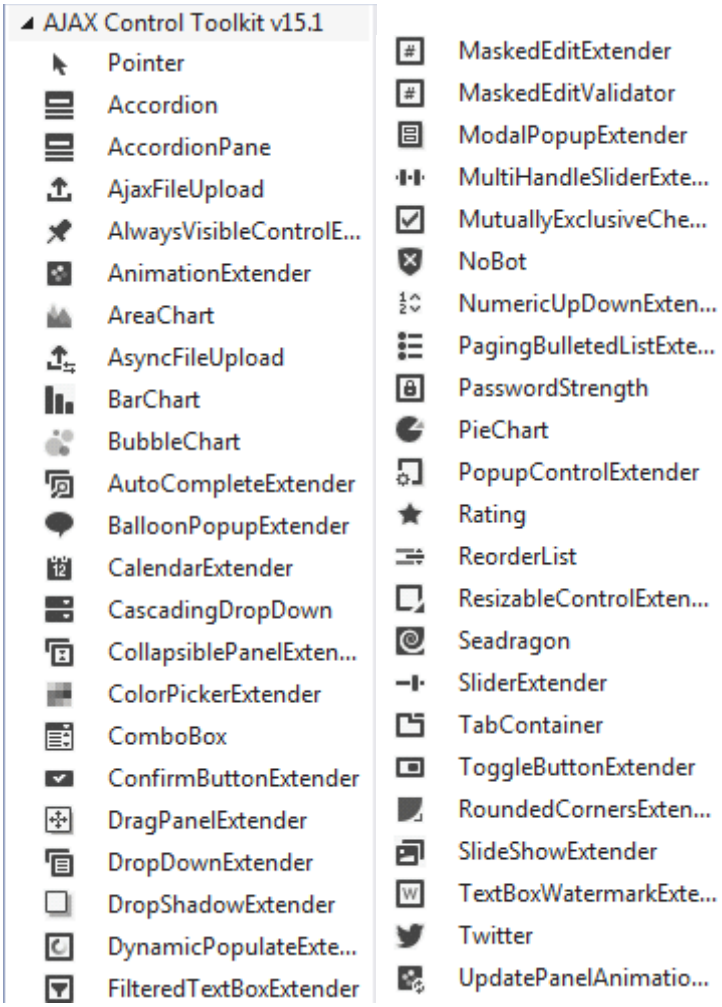
```
Type.registerNamespace("AjaxServerControl1");

 AjaxServerControl1.ClientControl1 = function(element) {
AjaxServerControl1.ClientControl1.initializeBase(this, [element]);
 }

 AjaxServerControl1.ClientControl1.prototype = {
initialize: function() {
AjaxServerControl1.ClientControl1.callBaseMethod(this, 'initialize');

// Add custom initialization here
     },
dispose: function() {
//Add custom dispose actions here
AjaxServerControl1.ClientControl1.callBaseMethod(this, 'dispose');
     }
 }
AjaxServerControl1.ClientControl1.registerClass('AjaxServerControl1.ClientControl1',
Sys.UI.Control);


if (typeof(Sys) !== 'undefined') Sys.Application.notifyScriptLoaded();
```

- AjaxServerControl1.ClientControl1 = function(element) –konstruktor klasy zawierający elementy DOM.
- AjaxServerControl1.ClientControl1.prototype = Deklaracja klasy zawierająca inicjalizowanie oraz usuwanie metod.
- AjaxServerControl1.ClientControl1.registerClass() –rejestruje klasę w przestrzeni nazw klienta.

AJAX Control Toolkit

Narzędzie to dodaje nowe kontrolki oraz umożliwia rozszerzyć pozostałe kontrolki o nowe funkcjonalności.

Lista dodanych kontrolek



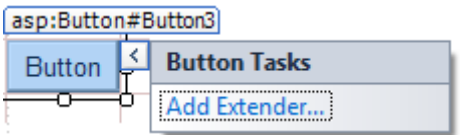
Przykład użycia rozszerzenia:

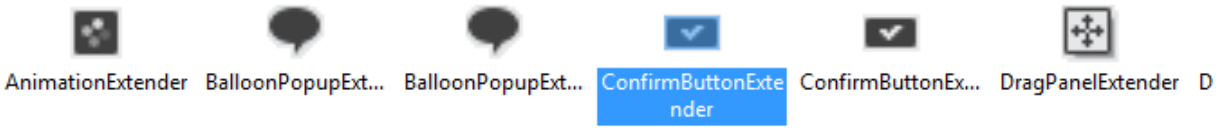
Zamiast pisać cały kod do wyświetlania komunikatu

```
<asp:ButtonID="Button2"runat="server"Text="Button"OnClientClick="CallServerMethod();
return false;"/>

<scripttype="text/javascript">

functionCallServerMethod() {

PageMethods.GetMessage(OnSuccess);
        }
functionOnSuccess(result) {
alert(result);
}
```

```
[WebMethod]
publicstaticstringGetMessage()
{
return"jakaś wiadomość";


    }
```

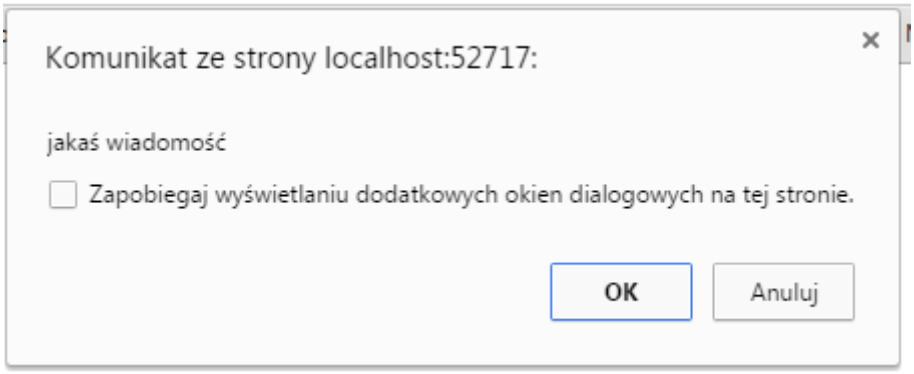
Można do buttona dodać rozszerzenie





Po dodaniu Extendera w pliku aspx do buttona dochodzi kod odpowiadający za jego obsługę.

```
<asp:ButtonID="Button3"runat="server"Text="Button"/>
<ajaxToolkit:ConfirmButtonExtenderID="Button3_ConfirmButtonExtender"runat="server"Beha
viorID="Button3_ConfirmButtonExtender"ConfirmText=" jakaświadomość
"TargetControlID="Button3"/>
```



Wykorzystywanie technologii AJAX w ASP.NET jest do tego stopnia uproszczone, że przeciętny twórca aplikacji może ją wykorzystywać nie wiedząc w ogólne o istnieniu obiektu XMLHTTPRequest.