Dotnet France
Technologies Sharepoint, SQL Server & .NET

Association Dotnet France

Ajax Client History Points

Version 1.0

James RAVAILLE
http://blogs.dotnet-france.com/jamesr

Sommaire

1 Introduction

1.1 Présentation

Dans le cours sur les bases fondamentales de Microsoft ASP .NET Ajax publié sur Dotnet-France, nous avons étudié le rôle et la mise en œuvre du contrôle *UpdatePanel*, dans les pages ASP .NET. Nous avons particulièrement vu que ce contrôle permettait de mettre à jour des parties de pages. Cependant, si un utilisateur navigue dans les différentes pages mises dans son cache, il verra que les mises à jour partielles de page ne sont pas mises en cache, ce qui peut provoquer un disfonctionnement de l'application, côté client.

Dans ce cours, nous vous proposons alors d'étudier le mécanisme que propose le Service Pack 1 du Framework .NET 3.5, pour gérer le cache côté client, lors de la mise à jour partielle de pages : Ajax Client History Point.

1.2 Pré-requis

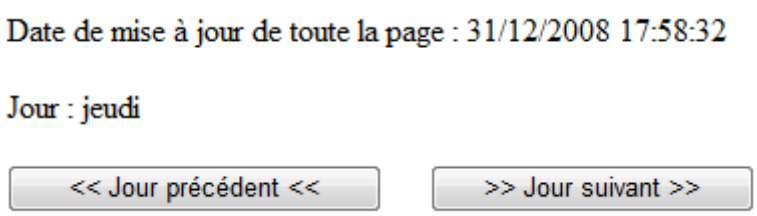
Avant le lire ce cours, nous vous conseillons :

- D'être familier avec la création et conception de pages ASP .NET.
- De lire les cours sur les bases fondamentales de Microsoft ASP .NET Ajax.

2 Développement d'une simple application avec ASP .NET Ajax

2.1 Présentation de l'application

Nous allons commencer par développer une application très simple. Cette application contient la page suivante :



Le but de cette page est :

- D'afficher la date et l'heure courante lors de la mise à jour complète de la page.
- De permettre aux utilisateurs, de passer de manière séquentielle d'un jour de la semaine à un autre.

Le code de cette page est le suivant :

```
// C#

<form id="form1" runat="server">
    <asp:ScriptManager ID="ScriptManager1" runat="server">
    </asp:ScriptManager>

    Date de mise à jour de toute la page : <asp:Label
ID="LblDateHeureMajPage" runat="server"></asp:Label>
    <br />
    <br />

    <asp:UpdatePanel ID="UpdatePanel1" runat="server">
        <ContentTemplate>
            Jour : <asp:Label ID="LblJourCourant"
runat="server"></asp:Label>
            <br />
            <br />
            <asp:Button ID="CmdAllerJourPrecedent" runat="server"
                Text="<< Jour précédent <<"
onclick="CmdAllerJourPrecedent_Click" />
                
            <asp:Button ID="CmdAllerJourSuivant" runat="server" Text=">>
Jour suivant >>"
                onclick="CmdAllerJourSuivant_Click" />
        </ContentTemplate>
    </asp:UpdatePanel>
</form>
```

```
// C#

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class PremierePage : System.Web.UI.Page
{
    public enum JoursSemaine {
        lundi = 1, mardi = 2, mercredi = 3, jeudi = 4, vendredi = 5,
        samedi = 6, dimanche = 7
    }

    protected void Page_Load(object sender, EventArgs e) {
        JoursSemaine oJourSemaine;
        oJourSemaine = JoursSemaine.lundi;
        if (!this.IsPostBack) {
            if (Session["JourCourant"] == null) {
                Session["JourCourant"] = oJourSemaine;
                LblJourCourant.Text = oJourSemaine.ToString();
            }
        }
        LblDateHeureMajPage.Text = DateTime.Now.ToString();
    }

    protected void CmdAllerJourPrecedent_Click(object sender, EventArgs e)
    {
        JoursSemaine oJourSemaine;
        if (Session["JourCourant"] == null)
            oJourSemaine = JoursSemaine.lundi;
        else {
            oJourSemaine = (JoursSemaine)Session["JourCourant"];
            if (((int)oJourSemaine) > 1)
                oJourSemaine--;
        }
        Session["JourCourant"] = oJourSemaine;
        LblJourCourant.Text = oJourSemaine.ToString();

    }

    protected void CmdAllerJourSuivant_Click(object sender, EventArgs e)
    {
        JoursSemaine oJourSemaine;
        if (Session["JourCourant"] == null)
            oJourSemaine = JoursSemaine.lundi;
        else {
            oJourSemaine = (JoursSemaine)Session["JourCourant"];
            if (((int)oJourSemaine) < 7)
                oJourSemaine++; // On passe au jour précédent.
        }
        Session["JourCourant"] = oJourSemaine;
        LblJourCourant.Text = oJourSemaine.ToString();
    }
}
```

```
' VB .NET

<form id="form1" runat="server">
    <asp:ScriptManager ID="ScriptManager1" runat="server">
    </asp:ScriptManager>

    Date de mise à jour de toute la page : <asp:Label
ID="LblDateHeureMajPage" runat="server"></asp:Label>
    <br />
    <br />

    <asp:UpdatePanel ID="UpdatePanel1" runat="server">
        <ContentTemplate>
            Jour : <asp:Label ID="LblJourCourant"
runat="server"></asp:Label>
            <br />
            <br />
            <asp:Button ID="CmdAllerJourPrecedent" runat="server"
                Text="<< Jour précédent <<" />
                
            <asp:Button ID="CmdAllerJourSuivant" runat="server" Text=">>
Jour suivant >>" />
        </ContentTemplate>
    </asp:UpdatePanel>
</form>
</body>
```

```vb
' VB .NET

Public Enum JoursSemaine As Integer
    lundi = 1
    mardi = 2
    mercredi = 3
    jeudi = 4
    vendredi = 5
    samedi = 6
    dimanche = 7
End Enum

Protected Sub Page_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load
    Dim oJourSemaine As JoursSemaine
    oJourSemaine = JoursSemaine.lundi
    If (Not Me.IsPostBack) Then
        If (Session("JourCourant") Is Nothing) Then
            Session("JourCourant") = oJourSemaine
            LblJourCourant.Text = oJourSemaine.ToString()
        End If
    End If
    LblDateHeureMajPage.Text = DateTime.Now.ToString()
End Sub

Protected Sub CmdAllerJourPrecedent_Click(ByVal sender As Object, ByVal e
As System.EventArgs) Handles CmdAllerJourPrecedent.Click
    Dim oJourSemaine As JoursSemaine
    If (Session("JourCourant") Is Nothing) Then
        ' On se positionne sur le premier jour de la semaine.
        oJourSemaine = JoursSemaine.lundi
    Else
        oJourSemaine = CType(Session("JourCourant"), JoursSemaine)
        If (oJourSemaine > 1) Then
            ' On passe au jour suivant.
            oJourSemaine -= 1
        End If
    End If
    Session("JourCourant") = oJourSemaine
    LblJourCourant.Text = oJourSemaine.ToString()
End Sub

Protected Sub CmdAllerJourSuivant_Click(ByVal sender As Object, ByVal e
As System.EventArgs) Handles CmdAllerJourSuivant.Click
    Dim oJourSemaine As JoursSemaine
    If (Session("JourCourant") Is Nothing) Then
        ' On se positionne sur le premier jour de la semaine.
        oJourSemaine = JoursSemaine.lundi
    Else
        oJourSemaine = CType(Session("JourCourant"), JoursSemaine)
        If (oJourSemaine < 7) Then
            ' On passe au jour suivant.
            oJourSemaine += 1
        End If
    End If
    Session("JourCourant") = oJourSemaine
    LblJourCourant.Text = oJourSemaine.ToString()
End Sub
```

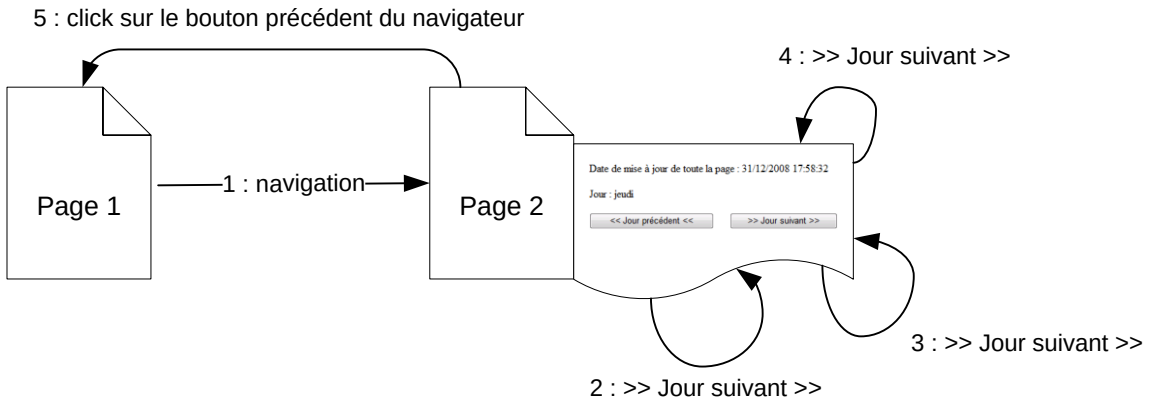
2.2 Observation sur l'utilisation du cache côté client

On peut remarquer, que la partie du formulaire permettant de passer d'un jour à l'autre de la semaine, est positionné dans un contrôle *UpdatePanel*. Autrement dit, cette partie de l'application est mise à jour, de manière indépendant de la page. Ainsi, lorsqu'on utilise les boutons permettant de passer au jour précédent, ou au jour suivant, le jour demandé apparaît.

Maintenant, si jamais nous essayons de revenir sur les jours précédents ou suivants au travers du navigateur :

- Soit les boutons du navigateur ne sont actifs, car la page n'a pas été rafraîchie dans son intégralité, ou c'est la première page sur laquelle le navigateur a été ouvert.
- Soit nous retournons sur la page précédente ou suivante. Le jour précédent ou le jour suivant ne s'affiche pas.

Ces explications sont illustrées par le schéma suivant :



Suivant les étapes indiquées ci-dessus, les jours apparaissent dans l'ordre indiqué ci-dessous :

- Etape 1 : lundi
- Etape 2 : mardi
- Etape 3 : mercredi
- Etape 4 : jeudi
- Etape 5 : retour sur la page 1

Le but de la gestion du cache côté client lors de la mise à jour partielle de page, est de permettre d'afficher le jour précédent, au lieu de retourner sur la page précédente.

3 Gestion du cache côté client lors de mise à jour partielle de page

3.1 Activation de la gestion du cache

Pour activer la mise en cache (côté client) des pages mises partiellement à jour, valorisons la propriété *EnableHistory* du contrôle *ScriptManager* à *True*.

Deux remarques :

- La mise en cache (côté client) des pages, mises partiellement à jour, est désactivée par défaut.
- Cette opération n'est pas possible au travers du contrôle *ScriptManagerProxy*.

```
// C# et VB .NET

<asp:ScriptManager ID="ScriptManager1" runat="server"
EnableHistory="true" />
```

3.2 Création de points historiques

Un point historique du navigateur correspond à une marque, permettant de statuer l'état d'une page, lors de la mise à jour partielle d'une page.

Pour créer un point historique du navigateur, il est nécessaire d'appliquer la méthode *AddHistoryPoint* sur le contrôle *ScriptManager*. Cette méthode permet de maintenir l'état des donnes côté serveur, ainsi qu'une clé utilisée comme données identifiant le point d'historique.

Dans notre exemple, ou créer des points historiques ? Nous allons en créer dans le code exécuté lors de la mise à jour partielle des pages, dans les méthodes permettant de passer au jour précédent ou suivant. Pour ne pas dupliquer le code, nous créer une nouvelle méthode nommée *EnregistrerDansCache*.

```
// C#

private void EnregistrerDansCache(JoursSemaine aJourSemaine)
{
    if (ScriptManager1.IsInAsyncPostBack && !ScriptManager1.IsNavigating)
    {
        // Création du point historique;
        System.Collections.Specialized.NameValueCollection DataContext =
new NameValueCollection();
        DateTime oDt = DateTime.Now;
        DataContext.Add("DateHeureCourante", oDt.ToString());
        DataContext.Add("JourCourant", ((int)aJourSemaine).ToString());

        // Ajout du point historique.
        ScriptManager1.AddHistoryPoint(DataContext, "Cache " +
oDt.ToString());
    }
}
```

```vb
' VB

Private Sub EnregistrerDansCache(ByVal aJourSemaine As JoursSemaine)
    If ScriptManager1.IsInAsyncPostBack And Not
ScriptManager1.IsNavigating Then
        ' Création du point historique;
        Dim DataContext As New
System.Collections.Specialized.NameValueCollection()
        Dim oDt As DateTime = DateTime.Now
        DataContext.Add("DateHeureCourante", oDt.ToString())
        DataContext.Add("JourCourant",
Convert.ToInt32(aJourSemaine).ToString())

        ' Ajout du point historique.
        ScriptManager1.AddHistoryPoint(DataContext, "Cache " +
oDt.ToString())
    End If
End Sub
```

Lors de la création d'un point d'historique, on enregistre la date et l'heure courante de la mise en cache, ainsi que le jour courant. Ces données pour être utilisées côté serveur, lorsque le client souhaitera naviguer dans son historique.

A la fin de la méthode des méthodes *CmdAllerJourPrecedent_Click* et *CmdAllerJourSuivant_Click*, on ajoute la ligne de code suivante :

```csharp
// C#

this.EnregistrerDansCache(oJourSemaine);
```

```vb
' VB

Me.EnregistrerDansCache(oJourSemaine)
```

3.3 Navigation dans l'historique

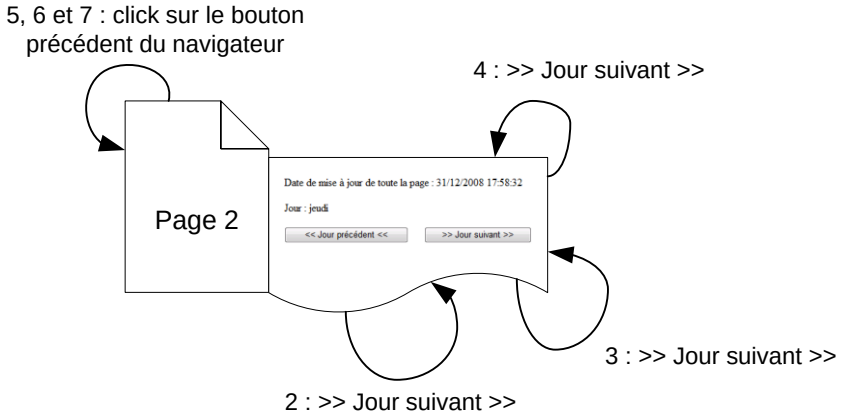
Pour naviguer dans l'historique, lors des clicks sur les boutons de navigation du navigateur, il est nécessaire de parcourir les points historiques qui ont été enregistrés. Pour ce faire, nous devons implémenter l'évènement *Navigate* sur le contrôle *ScriptManager* (un postback est effectué côté serveur). Dans cette méthode, il est suffit de récupérer le contexte de données lié au point d'historique courant, et d'exploiter les données qu'il contient, afin de mettre à jour la page. Dans notre cas, nous affichons la date et l'heure de mise en cache, ainsi que le jour de la semaine. Si rien n'est présent dans le cache, alors aucune date/heure n'est affichée et le premier jour de la semaine est affiché.

```
// C#

protected void ScriptManager1_Navigate(object sender, HistoryEventArgs e)
{
    NameValueCollection DataContext = e.State;
    JoursSemaine oJourSemaine;

    if (DataContext != null)
    {
        string dtS = DataContext.Get("DateHeureCourante");
        if (!String.IsNullOrEmpty(dtS))
        {
            LblDateHeureMajPage.Text = dtS;
        }
        else
        {
            LblDateHeureMajPage.Text = String.Empty;
        }

        string idJourSemaine = DataContext.Get("JourCourant");
        if (!String.IsNullOrEmpty(idJourSemaine))
        {
            oJourSemaine = (JoursSemaine)(int.Parse(idJourSemaine));
        }
        else
        {
            oJourSemaine = JoursSemaine.lundi;
        }

        Session["JourCourant"] = (int)oJourSemaine;
        LblJourCourant.Text = oJourSemaine.ToString();
    }
}
```

```
' VB

Protected Sub ScriptManager1_Navigate(ByVal sender As Object, ByVal e As
HistoryEventArgs) Handles ScriptManager1.Navigate
    Dim DataContext As NameValueCollection = e.State
    Dim oJourSemaine As JoursSemaine

    If DataContext IsNot Nothing Then
        Dim dtS As String = DataContext.Get("DateHeureCourante")
        If Not String.IsNullOrEmpty(dtS) Then
            LblDateHeureMajPage.Text = dtS
        Else
            LblDateHeureMajPage.Text = String.Empty
        End If

        Dim idJourSemaine As String = DataContext.Get("JourCourant")
        If Not String.IsNullOrEmpty(idJourSemaine) Then
            oJourSemaine = CType(idJourSemaine, JoursSemaine)
        Else
            oJourSemaine = JoursSemaine.lundi
        End If

        Session("JourCourant") = Convert.ToInt32(oJourSemaine)
        LblJourCourant.Text = oJourSemaine.ToString()
    End If
End Sub
```

3.4 Test de l'application

Nous allons donc exécuter les mêmes étapes que précédemment (avant la mise en place de la gestion de l'historique) :



Suivant les étapes indiquées ci-dessus, les jours apparaissent dans l'ordre indiqué ci-dessous :

- Etape 1 : lundi
- Etape 2 : mardi
- Etape 3 : mercredi
- Etape 4 : jeudi
- Etape 5 : mercredi (après postback sur le serveur)
- Etape 6 : mardi (après postback sur le serveur)
- Etape 7 : lundi (après postback sur le serveur)

Lors de chaque click sur le bouton précédent du navigateur, la page est exécutée côté serveur, suite à une opération de postback. Dans notre cas, les méthodes *Page_Load* et *ScriptManager1_Navigate* sont exécutées. Dans cette dernière :

- Les données mises précédemment en cache sont alors lues et affichées dans la page (date/heure de mise en cache et jour de la semaine).
- Enregistrement dans le contexte de session du jour de la semaine affiché. Ainsi, lorsque les utilisateurs cliquent sur les boutons *Jour précédent* et *Jour suivant*, le jour attendu est affiché.