

Interopérabilité FLASH MX et ASP.Net via les XML WebServices : Recadrage d'une image et Custom Control

05/12/2005

Par laurent GEFROY

Droit de diffusion:

L'ensemble ou partie de ce document ainsi que le code mis à disposition, ne peut être diffusé sur d'autres sites Web sans l'autorisation au préalable de son créateur. L'utilisation de l'image Flash et de son code sont utilisables à des fins non commerciales. Pour toute utilisation commerciale, merci de demander l'utilisation à son auteur.

Avant Propos :

Amoureux de Flash depuis 5 ans et passionné par .Net, il fallait faire un tutoriel reliant les deux. J'ai développé avec difficulté il y a 4 ans tout un forum en Flash et ASP3.0. Echanges via des fichiers ASP, faiblesse de Flash 4 sur les ascenseurs, bref, j'ai réussi mais non sans mal. Mais aujourd'hui, le même développement avec les outils du troisième millénaire serait bien plus simple.

J'ai donc décidé de faire un développement qui permettrait de recadrer une image JPG en fonction d'un certain nombre de paramètres. Le tout est développé sur le Framework 2.0, VisualStudio 2005 et Flash MX Professional (8). [Attention, le code entièrement en vert est de l'ActionScript !!!](#)

Ce tutoriel passe en revue plusieurs techniques :

- La création d'un Custom Control ;
- L'injection de codes JavaScript dans le custom Control ;
- La mise en place d'un XML WebService manipulant une image JPG ;
- Le développement d'une application Flash (Chargement dynamique d'image, barre de progression) et la consommation du WebService

1. La création d'un Custom Control

Afin de manipuler mon image JPG, je passe d'abord par une image flash dans laquelle je dois passer un certain nombre de paramètres. Ces paramètres seront passés dans l'URL. Pas très sécurisé car le chemin de l'image sur mon serveur est passé en clair. Pour une application en production, mieux vaut faire autrement (Passer un ID qui serait utilisé dans l'image Flash pour trouver le chemin complet).

J'ai alors décidé de créer un Custom Control qui construira mon script Flash.

Nouveauté du Framework 2.0, le code de mon Custom Control est placé dans la directory App_Code, tout comme celui de mon WebService. Voici mon Custom Control, ccImageTool :

```
[AspNetHostingPermission (SecurityAction.Demand,  
Level = AspNetHostingPermissionLevel.Minimal),  
AspNetHostingPermission (SecurityAction.InheritanceDemand,  
Level = AspNetHostingPermissionLevel.Minimal),  
DefaultProperty ("FlashUrl"),  
ToolboxData ("<{0}:ImageTool runat=\"server\"> </{0}:ImageTool>")]
```

Les attributs vont me permettre de régler les règles de sécurité ainsi que l'apparence de mon control dans le designer.

Viennent ensuite la série des paramètres de mon Custom Control :

```
[Bindable(true),
Category("Appearance"),
DefaultValue(""),
Description("Chemin de l'image FLASH"),
Localizable(true)]
public virtual string FlashUrl
{
    get
    {
        string s = (string)ViewState["FlashUrl"];
        return (s == null) ? String.Empty : s;
    }
    set
    {
        ViewState["FlashUrl"] = value;
    }
}
```

Ici on détermine l'url de l'image Flash. Les valeurs sont insérées dans le ViewState, ce qui permettra de les récupérer en cas dePostBack.

Afin de définir le rendu, on utilise la méthode suivante :

```
protected override void RenderContents(HtmlTextWriter writer)
{
```

Afin de bien fonctionner, mon image Flash a besoin de 2 fichiers Externes, l'un en Javascript, l'autre en VBScript. Pour insérer mon Javascript, j'utilise la méthode RegisterStartupScript :

```
Page.ClientScript.RegisterStartupScript(typeof(Page), "Flash", "<script
language=\"JavaScript\" src=\"\" + JScripURL + "/ImgTools.js\"></script>
");
Page.ClientScript.RegisterStartupScript(typeof(Page), "FlashVB", "<script
language=\"VBScript\" src=\"\" + JScripURL + "/ImgToolsVB.js\"></script>
");
```

Je construis ensuite l'url de mon image Flash avec ces différents arguments nécessaires. Random permet d'éviter que Flash ne mette en cache le rendu, configfile paramètre les possibilités de redimensionnement, imgname donne l'url de l'image JPG a afficher, et imgpath son chemin sur le serveur.

```
string v_flash = FlashUrl + "?random=" + System.Guid.NewGuid().ToString()
+ "&configfile=" + ConfigFile + "&imgname=" + JpgName + "&imgpath=" +
JpgOnServer;
```

Je fabrique en Javascript le code Flash nécessaire pour l'insérer dans du HTML

```
string jscript = "";
jscript += "\r\n<script language=\"JavaScript\"
type=\"text/javascript\">\r\n";
jscript += "var hasRightVersion = DetectFlashVer(requiredMajorVersion,
requiredMinorVersion, requiredRevision);\r\n" ;
jscript += "if(hasRightVersion) { \r\n";
jscript += "var oeTags = '<object classid=\"clsid:D27CDB6E-AE6D-11cf-96B8-
444553540000\"'";
jscript += "\r\n + 'width=\"\" + Flash_Width + "\" height=\"\" + Flash_Height
+ \"\" '";
```

```
jscript += "\r\n
+'codebase=\"http://download.macromedia.com/pub/shockwave/cabs/flash/swflas
h.cab\">';
jscript += "\r\n +'<param name=\"movie\" value=\"\" + v_flash + \"
/><param name=\"menu\" value=\"false\" /><param name=\"quality\"
value=\"high\" /><param name=\"bgcolor\" value=\"\" + Flash_BgColor + \"
/>';
jscript += "\r\n +'<embed src=\"\" + v_flash + \"\" menu=\"false\"
quality=\"high\" bgcolor=\"\" + Flash_BgColor + \"\" '";
jscript += "\r\n +'width=\"\" + Flash_Width + \"\" height=\"\" + Flash_Height
+ \"\" name=\"ImgTool\" align=\"middle\"";
jscript += "\r\n +'play=\"true\"' + 'loop=\"false\"' + 'quality=\"high\"'
+ 'allowScriptAccess=\"sameDomain\"' + 'type=\"application/x-shockwave-
flash\"';
jscript += "\r\n +
'pluginspage=\"http://www.macromedia.com/go/getflashplayer\">' +
'</embed></object>'; ";
jscript += "\r\n document.write(oeTags);";
jscript += "\r\n} else {";
jscript += "\r\nvar alternateContent = 'Version Flash trop ancienne. Ce
contenu requiert Macromedia Flash Player.'";
jscript += "\r\n + '<a
href=\"http://www.macromedia.com/go/getflash/\">Obtenir Flash</a>';";
jscript += "\r\n document.write(alternateContent);";
jscript += "\r\n}\r\n</script>";
```

L'image Flash est également insérée via un Javascript.

```
Page.ClientScript.RegisterStartupScript(typeof(Page), "Flash_Movie",
jscript);
}
```

Il est également possible de passer les variables à l'application Flash via un Javascript. N'oubliez pas d'ajouter dans la construction de l'image flash le paramètre `swLiveConnect= « true »`.

2. Le XML Webservice de découpage

Je crée maintenant mon Webservice qui me permettra de recadrer mon image. Il est préférable de développer par couches. Idéalement, votre méthode de découpage devrait être dans la couche Business, appelée par le Webservice. Dans un souci de clareté, le code est dans le Webservice. Ce n'est pas très clean, mais c'est parfait pour cet exemple.

Mon Webservice attend 5 arguments :

- Le chemin de l'image. Comme je vous l'ai dit, en production, évitez ce type de chose et prévoyez une méthode plus sécurisée.
- La position en x du cadre de découpe,
- La position en y du cadre de découpe,
- La longueur du cadre de découpe,
- La largeur du cadre de découpe.

Mon Webservice retourne une structure avec un boolean et une valeur (Message d'erreur ou url de l'image redimensionnée).

```
[WebMethod(Description="Redimensionne une image JPG")]
public WSRetour Redimensionne(string img_path, int x, int y, int longueur, int
largeur)
{
    WSRetour v_result = new WSRetour();
    v_result.Ok = false;
    v_result.Retour = "Erreur Générale";

    try
    {
```

Je commence par charger mon image passée en argument dans un objet Image.

```
System.Drawing.Image img;

img = System.Drawing.Image.FromFile(img_path);
```

Il est mis dans un objet Bitmap afin de permettre sa manipulation.

```
Bitmap bmpImage = new Bitmap(img);

// Creation du rectangle de destination
Rectangle recCrop = new Rectangle(x, y,
    longueur, largeur);

// Création de la bitmap
Bitmap bmpCrop = new Bitmap
    (longueur, largeur,
    bmpImage.PixelFormat);

Graphics gphCrop = Graphics.FromImage(bmpCrop);
```

L'image découpée est mise dans un rectangle de destination.

```
// Rectangle de destination
Rectangle recDest = new Rectangle(0, 0,
    longueur, largeur);

gphCrop.DrawImage(bmpImage, recDest,
    recCrop.X, recCrop.Y,
    recCrop.Width, recCrop.Height,
    GraphicsUnit.Pixel);
```

Le Bitmap est sauvegardé.

```
        bmpCrop.Save("c:\\inetpub\\wwwroot\\ImgTools\\testOK.jpg",  
                    ImageFormat.Jpeg);  
  
        v_result.Ok = true;  
        v_result.Retour = "http://localhost/ImgTools/testOK.jpg";  
    }  
    catch (Exception ex)  
    {  
        v_result.Retour = ex.Message;  
    }  
    return v_result;  
}
```

Voilà, le tour est joué pour la partie du Webservice.

3. La page ASPX d'appel

Je commence par référencer mon Custom Control dans ma page ASPX.

```
<%@ Register TagPrefix="aspImageTool" Namespace="CustomControls" %>
```

J'inclus ensuite mon Control. Les valeurs de mes paramètres peuvent se faire en code managé.

```
<aspImageTool:ccImageTool  
    Runat="Server" id="WC1" Flash_BgColor="#FFFFFF" FlashUrl="/ImgTools/ImgTool.swf"  
    ConfigFile="config.xml" JpgName="http://localhost/imgtools/105.jpg"  
    JpgOnServer="c:\\inetpub\\wwwroot\\imgtools\\105.jpg" JScriptsUrl="/imgTools" />
```

4. L'application Flash

Le but ici n'est pas de rentrer dans l'intégralité du Code Flash, mais plutôt de vous expliquer quelques fonctionnements de cette application

4.1 La récupération des paramètres

Les données sont récupérées via les arguments de l'URL de l'image Flash. Le fichier de configuration XML est récupéré dans la variable configfile.

```
<?xml version="1.0" encoding="utf-8" ?>  
<settings>  
    <selectorWidth value="400"/>  
    <selectorHeight value="100"/>  
    <selectorXModifier value="false"/>  
    <selectorYModifier value="true"/>  
    <selectorProportionnel value="false"/>  
    <goToMax value="true"/>  
    <displayControlPanel value="true"/>  
</settings>
```

Il permet de définir la longueur et la largeur du « découpeur ». `displayControlPanel` permet d'afficher ou de rendre invisible un outil de control des diverses positions dans l'image. Le `selectorXModifier` et `selectorYModifier` permette d'avoir la main ou non sur le redimensionnement en longueur et en hauteur du sélecteur. Le `selectorProportionnel` permet de conserver la proportionnalité du sélecteur par rapport aux valeurs d'origines. Quand au `goToMax` va permettre d'étendre le sélecteur aux dimensions maximum de l'image.

On lit de cette façon le fichier XML. J'ai utilisé un script qui permet de rechercher un nœud par son nom.

```
var _xml = new XML();
_xml.ignoreWhite = true;
_xml.onLoad = function(success){
    if (success){
        var _rootNode = this.firstChild;
        if (_rootNode.hasChildNodes) {
            elements = this.getElementsByTagName("selectorWidth");
            if (elements.length > 0) {
                item = elements[0].attributes["value"];
                _root.ControlPanel.selecteur_width.text = item;
                SELECTOR._width = int(item);
            }
        }
    }
}
_xml.load(configfile);
```

On charge le fichier Xml dans l'objet `_xml`. Puis, sur le chargement, on vérifie si le chargement est terminé. La récupération de la valeur de l'attribut du nœud se fait avec la méthode `getElementByTagName`.

4.2 La consommation du Webservice

Sur le clic du bouton, nous invoquons la méthode `CData` qui consomme le Webservice.

```
on (release) {
    // Redimensionnement
    CData();
}
```

N'oubliez pas d'importer les références `mx.services.*` de Flash MX.

```
import mx.services.*;

function CData()
{
    ...

    _root.ErrorXML.lb_error.text = "Interrogation du Webservice";
}
```

Je crée mon Objet Webservice dont l'url est inscrite dans la variable `WSURL`.

```
var _WebServices = new Webservice(_root.WSURL);
```

Je consomme mon Webservice

```
v_result = _WebServices.Redimensionne(_root.imgpath,
                                     int(_root.ControlPanel.crop_x.text),
                                     int(_root.ControlPanel.crop_y.text),
                                     int(_root.ControlPanel.crop_width.text),
                                     int(_root.ControlPanel.crop_height.text));
```

Sur le résultat de mon Webservice, je récupère mes valeurs en utilisant l'objet result.

```
v_result.onResult = function(result) {
    _root.ErrorXML._visible = true;
    if (string(result.Ok) == "true")
    {
        _root.ErrorXML.lb_error.text = "Chargement image recadrée";
        _root.smallloader._visible = true;
        _root.smallloader.contentPath = result.Retour;
        _root.ErrorXML._visible = false;
    }
    else
    {
        _root.ErrorXML.lb_error.text = result.Ok + "-" + result.Retour;
    }
    _root.ControlPanel.v_message.text = result.Ok;
}
```

En cas de problème, je gère aussi les erreurs

```
v_result.onFault = function(fault){
    _root.ErrorXML._visible = true;
    _root.ErrorXML.lb_error.text = "Erreur d'opération";
}
}
```

5. Pour conclure

Voilà, l'application pourrait bien entendue être optimisée. Elle permet d'inclure une sorte d'application riche dans une solution Web pour travailler vos images. Il serait également possible de créer une sorte de « PhotoShop » entièrement réalisée en Flash et en .Net coté serveur... A méditer.



6. En savoir plus

Création d'un Custom Control

http://www.ondotnet.com/pub/a/dotnet/excerpt/progaspdotnet_14/index2.html

Le préchargement d'une image dans flash avec une barre de progression

<http://bolo.developpez.com/prechargement/>

Chargement de variables Javascript dans Flash MX

http://www.macromedia.com/fr/support/flash/ts/documents/javascript_comm.htm

L'auteur

<http://www.laurentgeffroy.com>