

Construire un contrôle serveur Wysiwyg, 3^{ème} partie

1. Améliorations et perspectives possibles

Optimisation du Html

Remplacement par exemple des balises « strong » par « b ». On optimise ainsi le volume html de la page finale. Ce n'est pas à négliger.

Utilisation d'une feuille de style

Des fenêtres de propriétés dans notre petit éditeur pourraient prendre en compte une « class » (éventuellement, choix d'un nom de classes parmi ceux disponible dans la feuille de style que l'on préciserait). Il suffirait d'ajouter un link de notre css dans notre éditeur... (Vous savez comment faire maintenant !).

Réaliser un parseur du fichier css pour obtenir les class disponibles...

Intégration une bibliothèque d'images

Vous pouvez le faire. Pour ma part, je préfère dissocier Contenu texte et Images dans mes développements.

Optimisation des scripts javascript

On peut augmenter la compatibilité de notre petit éditeur, en développant des scripts javascript pour d'autres navigateurs web. Pour les mettre en place, rien de plus facile avec le pattern « stratégie ».

Création d'autres contrôles : vers un CMS ?

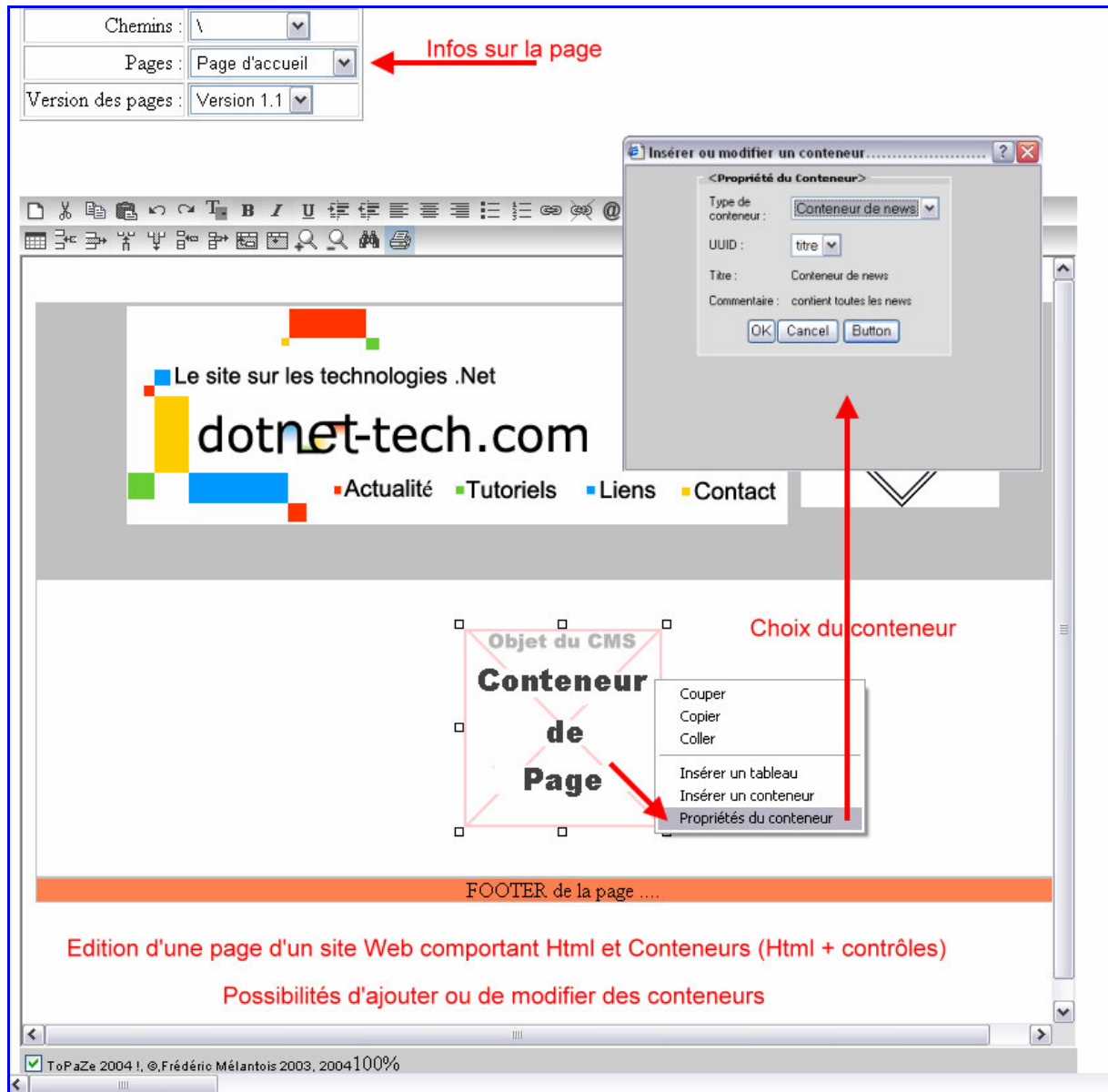
Nous pouvons détourner la fonction première de notre contrôle pour en faire un éditeur de conteneur de contrôle perso (custom control) puis un éditeur de contrôle perso.

Voici ci-dessous, ce que je développais il y a quelques mois, avant d'être repris par ma passion du traitement d'images ;-)

J'avais entrepris de développer un éditeur de pages (pour back-office). Les contenus de mes pages se trouvent en base de données. Mon principe est simple et classique : une page est constitué de html et de zones de contenu (je les appelle « conteneurs »). Voici ce que cela donne en back-office (par l'image, vous allez mieux comprendre) :

.NET passionnément, tout simplement

Construire un contrôle serveur Wysiwyg, 3^{ème} partie



Une fois mise en place la structure de la page (Html + conteneurs), il nous faut éditer nos conteneurs.

De quoi est fait un conteneur ?

Un conteneur est fait de Html et de contrôles serveur (ceux-ci respecte un certain « contrat » pour être « pluggable » en base de données et être donc reconnu par le conteneur).

.NET passionnément, tout simplement

Construire un contrôlé serveur Wysiwyg, 3^{ème} partie

The screenshot shows a WYSIWYG editor interface. At the top, there are two dropdown menus: 'Conteneurs' with 'titre' selected and 'Version des conteneurs' with 'version 1' selected. A red arrow points to these menus with the text 'Choix du conteneur dans la page'. Below this is a toolbar with various icons. The main editing area shows two identical boxes, each containing the text 'Objet du CMS Articles en tranches'. A red arrow points to the code generated for one of these boxes, which is: `<TR><TD vAlign=top align=left width=200 bgColor=silver><P></P></TD>`. A red arrow points to this code with the text 'Le code généré (un parseur s'occupe d'enregistrer en base)'. A context menu is open over the second box, with 'Insérer un composant' highlighted. A dialog box titled 'Insérer ou modifier un composant...' is also visible, showing 'Type de composants' set to 'article' and 'UUID' set to 'UUID1'. A red arrow points to this dialog with the text 'Choix du type de contrôles serveur et de son instance'. At the bottom of the screenshot, there are two lines of red text: 'Edition d'un conteneur dans une page web' and 'Possibilité d'ajouter, modifier des contrôles serveur'. The status bar at the bottom of the window shows 'ToPaZe 2004 !, ©, Frédéric Mélançois 2003, 2004 100%'.

L'enregistrement et le déploiement de notre page génèrent un fichier dont voici la source :

```
<%@ Page language="c#" ValidateRequest="false" Codebehind="index.aspx.cs" AutoEventWireup="false" Inheri
<%@ Register TagPrefix="cc1" Namespace="Topaze" Assembly="Topaze" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
<HEAD>
<title>Page d'accueil</title>
<META http-equiv="Content-Type" content="text/html; charset=utf-8">
</HEAD>
<body leftMargin="0" topMargin="0" marginheight="0" marginwidth="0">
<form id="Form1" method="post" runat="server">
<cc1:TZEPage id="TZEPage2" runat="server" UUID="687687-1010190980"></cc1:TZEPage></form>
</body>
</HTML>
```

On a ici un contrôlé serveur (« TZEPage ») qui va charger le Html et les contrôles serveur respectant un même contrat.

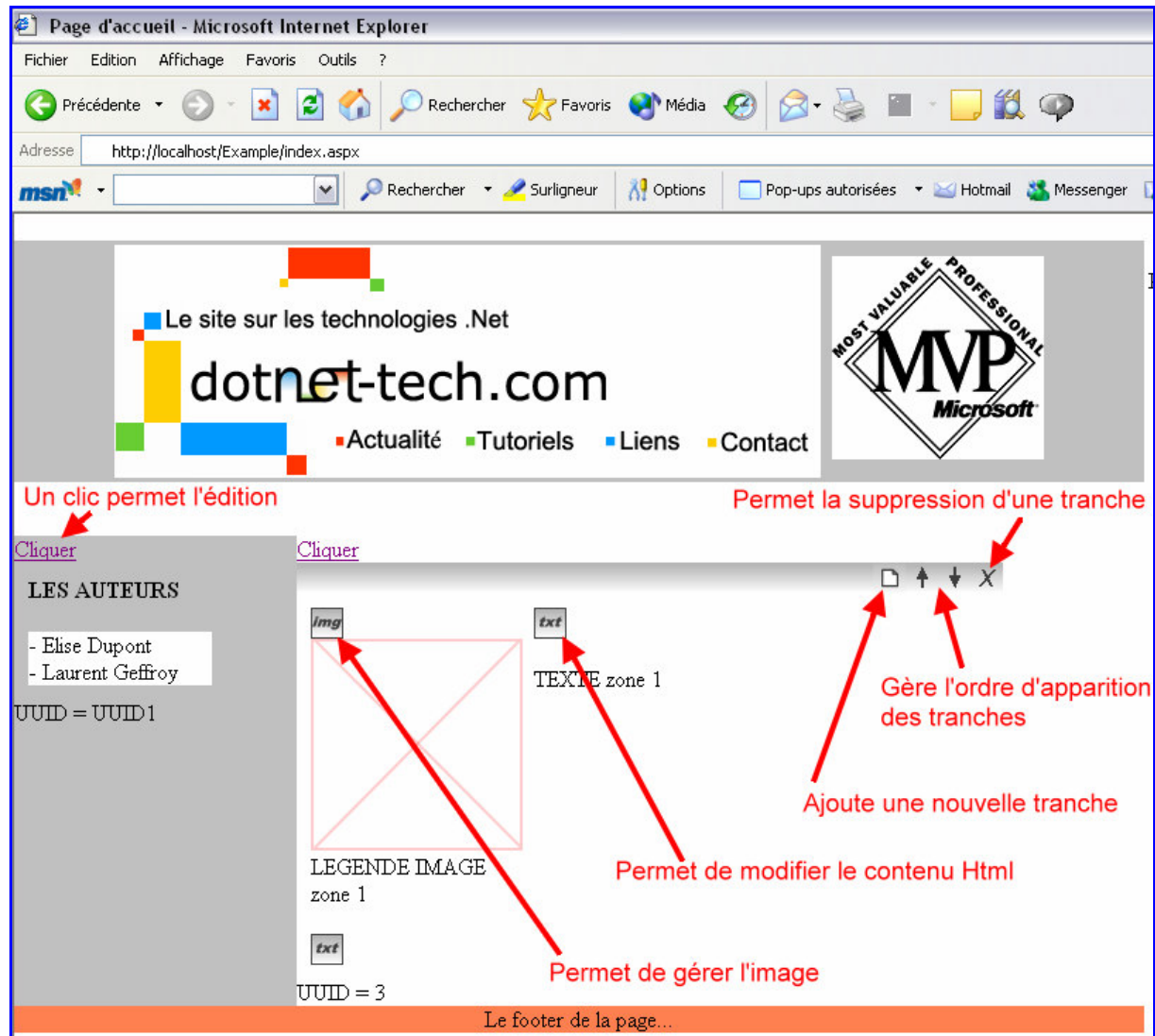
.NET passionnément, tout simplement

Construire un contrôle serveur Wysiwyg, 3^{ème} partie

Concrètement, un contrôle serveur « TZEPage » charge un ou plusieurs contrôles serveur de type conteneur « TZEConteneur » qui eux-mêmes chargent un ou plusieurs contrôles serveur « pluggable » (Ici, dans notre exemple « TZEArticle »).

On peut bien évidemment faire évoluer notre gestionnaire de contenu en rajoutant de nouveaux types de contrôles serveur « pluggable » respectant le contrat fixé. L'ensemble est donc facilement évolutif.

Lors du lancement de cette page, nous avons le rendu suivant (en « mode manager ») :



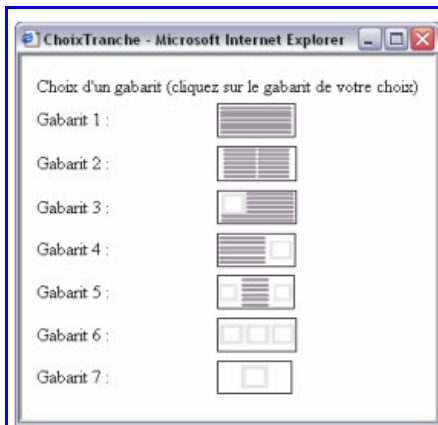
En comparant avec le back-office, vous pouvez repérer où se trouve le conteneur (il peut y en avoir plusieurs) et ce qu'il contient.

En mode « manager », les zones (« Article en tranches ») sont éditables pour notre client comme vous pouvez le voir.

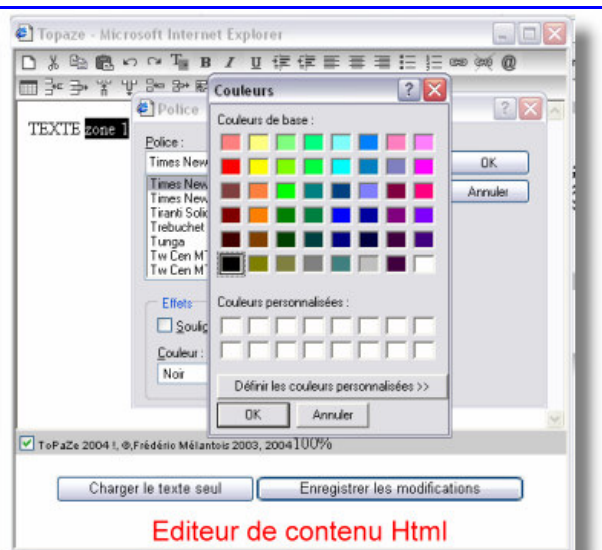
Ci-dessous, les différentes éditions de contenu possibles pour l'article en tranche (« TZEArticle »).

.NET passionnément, tout simplement

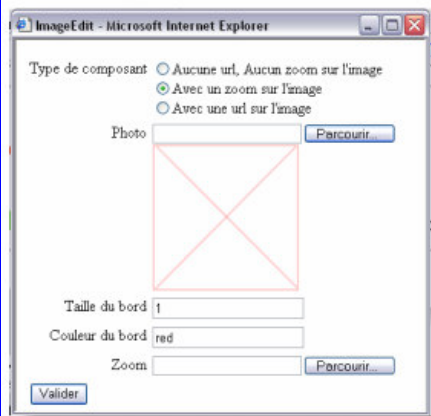
Construire un contrôle serveur Wysiwyg, 3^{ème} partie



Ajout d'une nouvelle tranche



Editeur de contenu Html



Gestion de l'image

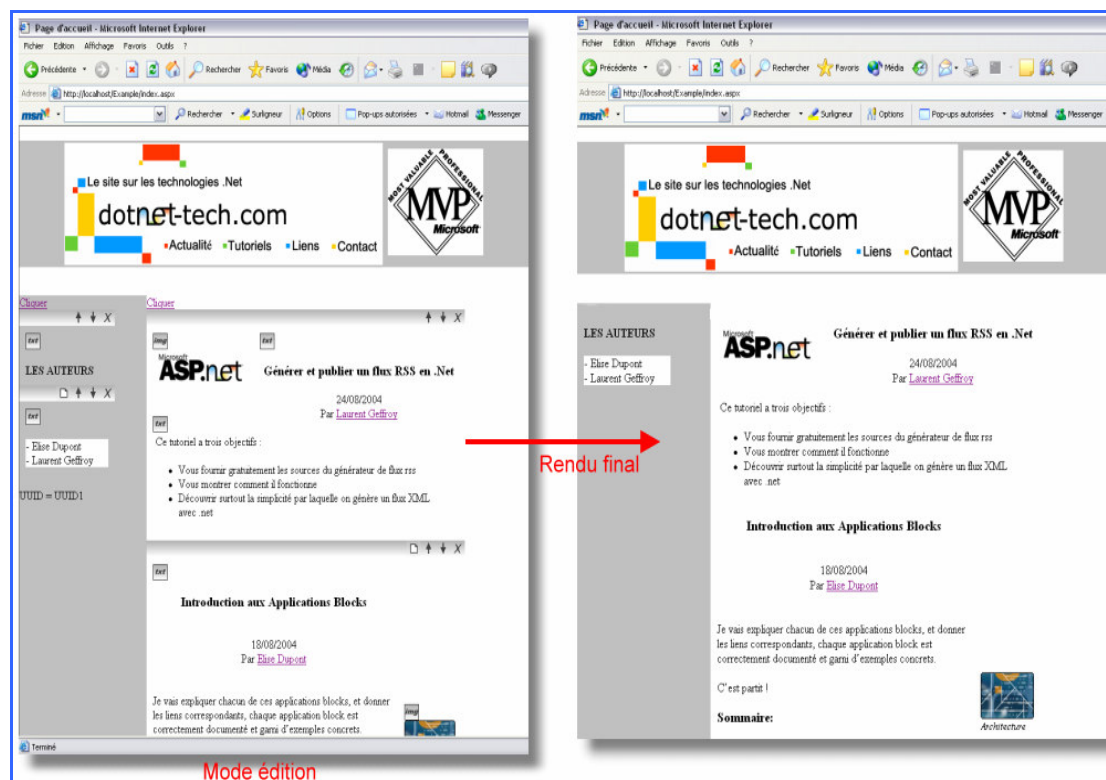
Dont notre article vous a présenté les sources



Diagramme de la base de données ;-)

.NET passionnément, tout simplement

Construire un contrôle serveur Wysiwyg, 3^{ème} partie



J'espère vous avoir montré qu'il était relativement facile de construire un contrôle serveur ASP.NET et d'envisager d'autres perspectives...

A noter que, ce qui est présenté en « améliorations et perspectives », n'est pas fourni en source de cet article. J'ai mis en « stand by » depuis plusieurs mois le développement de « Topaze ». A vrai dire aucun futur employeur ne s'est montré réellement intéressé, comme quoi passion et profession ne vont pas toujours de paire ;-)

Cela sera peut-être l'occasion de faire quelques nouveaux articles dès que j'aurai repris l'envie de le continuer.

En attendant, vous disposez des sources du contrôle serveur Wysiwyg de base. Puisque je n'ai pas mis de copyright aux sources, je vous demande simplement de partager vos trouvailles avec la Communauté .NET . Bon code !

Remerciement à Christophe Duhamel (aussi Ex-Allabanien, <http://www.marmiton.org>) pour l'idée des gabarits en tranches.

2. En savoir plus :

Présentation du Design Pattern Chaîne de Responsabilité (Tech Head Brothers)
<http://www.techheadbrothers.com/DesktopDefault.aspx?tabindex=1&tabid=7&Ald=34>

Les expressions régulières (Labo-Dotnet)
<http://www.labo-dotnet.com/labodotnet/?target=showonearticle&ID=45>

.NET passionnément, tout simplement

Construire un contrôle serveur Wysiwyg, 3^{ème} partie

Advanced ASP.NET Server-Side Controls (MSDN Magazine)

<http://msdn.microsoft.com/msdnmag/issues/01/03/visualprog/default.aspx>

FCK Editor (Open source)

<http://sourceforge.net/projects/fckeditor/>

Free Text Box (Open source)

<http://sourceforge.net/projects/freetextbox/>

Le livre : Developing ASP.NET Server Controls and Components (avec un exemple de contrôle Wysiwyg)

<http://www.microsoft.com/mspress/books/5728.asp>

Le newsgroup consacré à la construction des contrôles serveur ASP.NET

News : microsoft.public.dotnet.framework.aspnet.buildingcontrols

Contactez l'auteur :

Frédéric Mélantois

Email : fmelantois@free.fr