

## TP 1

# Bases d'un site Web en ASP.NET (origine Microsoft)

## 1 Introduction

Le but de ce TP est de vous présenter les techniques de mise en œuvre de la charte graphique et de structuration des pages de l'application web.

### 1.1 Contexte fonctionnel

L'objectif du TP est de construire pas à pas un site web de petites annonces (type bonnes affaires) proposant des services de :

- dépôt d'une annonce en ligne.
- consultation de toutes les annonces publiées.
- gestion du compte des utilisateurs du site qui souhaitent sélectionner des annonces et revenir régulièrement sur la consultation de leur sélection.

#### Contexte fonctionnel

- Il vous faut maintenant définir une charte graphique pour votre application web ainsi que la structure des pages du site. Vous allez mettre en place deux thèmes au choix, nommés **Default** et **Chaud**.

Thème  
**Default**  
sur  
tons bleu-gris.



Thème **Chaud**  
sur tons jaune.



- Ces caractéristiques graphiques vont s'appliquer à la page d'accueil *Default.aspx*, , ainsi qu'à une nouvelle page de contenu *QuiSommesNous.aspx* qui présente l'entreprise AffairesSansRisque.



Les fichiers utiles auxquels font référence les exercices sont disponibles dans le répertoire **fourni** pour le TP

## 2 Créer une première application web

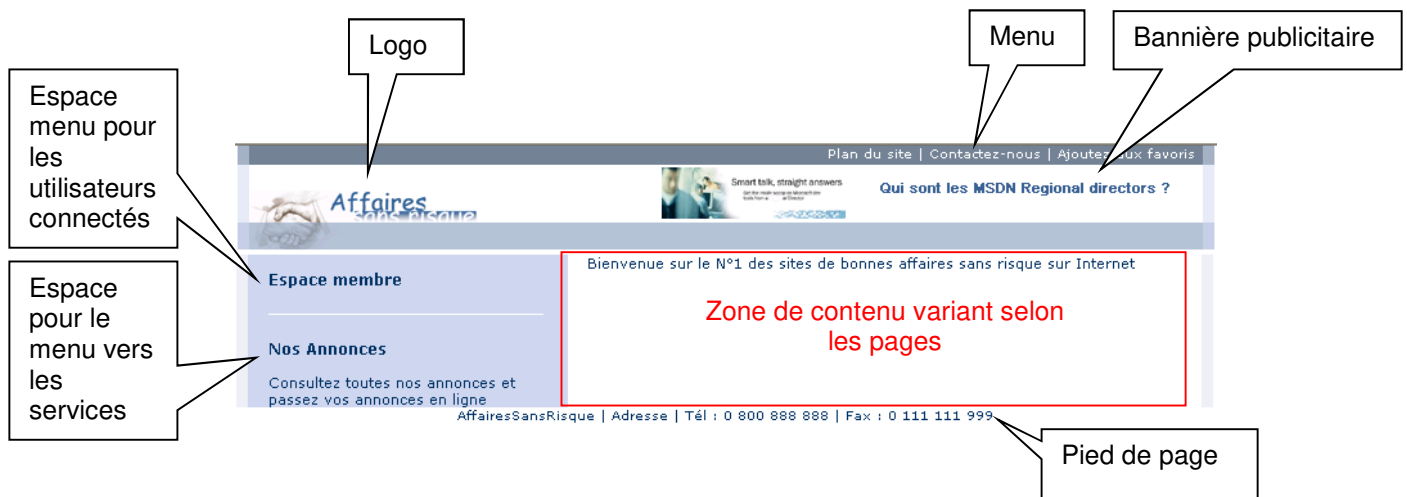
Partie 1 :

1. Créez un nouveau projet de type Web :
  - *Nouveau Site Web* → *Site Web ASP .Net Vide*
  - Sélectionnez le chemin du répertoire et donnez un nom au site :  
Attention vous ne devez pas travailler sur un répertoire réseau à cause des questions de sécurité. Sélectionnez par exemple le volume ou la partition D :
  - Sélectionnez le nom de votre projet puis à l'aide du bouton contextuel ajouter un nouvel élément. Sélectionner un fichier de type Web Form. Par défaut son nom est "default.aspx".
2. Ouvrez le fichier *Default.aspx.cs* en mode *Design*
  - *Déposer un contrôle Label*
  - A l'aide de l'onglet *propriété* changer le texte du Label en *Bienvenu*
  - Noter la syntaxe du Label en mode ASP
  - Vous pouvez « jouer » avec le mini navigateur html
3. Exécuter votre application Web
  - Que remarquez-vous au niveau de l'URL ?
4. Lancer l'exécution en mode debug
  - Que remarquez-vous ?
5. Afficher la page *Default.aspx.cs* pour faire apparaître la fenêtre de code appelé code behind
  - Modifier le texte du label précédemment créé par l'intermédiaire de la méthode *Page\_Load()*
  - `Label1.Text="Création du premier site ASP";`
  - Réactualiser votre page Web dans Firefox ou IE  
Vous devez constater que votre modification a été prise en compte sans compilation de votre code. Pourquoi ?

## 3 Créer une page Maître

### Contexte fonctionnel

Supposons que la structure et la charte graphique de votre site ressemble à ceci :



Toute la question est comment dessiner une telle structure de page ?

### 3.1 Diviser la page en zones de contenu

Déroulement :

Avant de vous lancer dans la construction d'une page maître, vous devez réfléchir à la structure des pages de votre site et en isoler les différents contenus logiques que l'on retrouve sur chacune d'entre elles.

Vous pouvez isoler :

- les zones de navigations principales, tels que les menus ou liens hypertextes.
- les zones de navigations secondaires,
- l'en-tête de page, avec le logo de la société, une bannière de publicité etc...
- le pied de page,
- la zone de contenu contextuel en fonction de la page en cours,
- etc...

.La page de référence sera décomposée en différentes zones de contenu.

Par exemple on peut avoir la description suivante :

Plan du site   Contactez-nous   Ajouter aux favoris	
Logo	Panneau publicitaire
Chemin de navigation	
Espace membre	Zone réservée aux pages enfants
Menu privé	
Services	
Menu des services	
Pied de page	

Zones de contenu	Description du contenu	Représentation du contenu
En-tête	Une barre avec des liens vers le plan du site, comment vous contacter, etc...	Toute la zone contient un fond d'écran original et coloré faisant office de bannière pour l'ensemble du site. La barre de liens et la barre de navigation sont alignées sur le bord droit de la zone. Le logo est à gauche.
	Un logo	
	Un panneau publicitaire	
	Une barre de navigation donnant des informations telles que le chemin de la page en cours dans la structure des pages du site.	
Corps de page	Une partie « espace membre » destinée aux utilisateurs authentifiés sur le site et aux informations de leur compte personnel. Une partie « Services » contenant le menu des services proposés par le site. Une partie centrale affichant le contenu dynamique d'une page à l'autre du site.	L'espace membre et la zone de services sont dans une barre colorée alignée sur le bord gauche. La zone centrale est au contraire calée à droite et sur fond blanc.
Pied de page	Panneau contenant des informations du type adresse, téléphone, fax de l'entreprise	La zone doit être centrée et est simplement sur fond blanc.

## 3.2 Etablir une feuille de style

6. Ajouter la feuille de style qui vous a été fournie dans votre projet. Vous pouvez faire un glisser déposer sinon à l'aide du menu contextuel de votre souris sélectionner "ajouter un élément existant"
  - Le fichier de style doit se trouver dans votre projet

- Ouvrez la barre *Structure CSS*. Cette barre comporte des *Classes*, des *Éléments*, des *ID*, ...
- Regardez le contenu de cette feuille de style et remarquez que cette feuille de styles fait référence à des images situées dans un répertoire nommé *Images*. Vous devez faire glisser ce répertoire dans votre projet (glisser déposer) :

### 3.3 Créer la page Maître

1. Créez une page Maître :
  - À l'aide du menu contextuel de votre souris ajouter un *nouvel élément*
  - Sélectionnez *page Maître* et gardez le nom par défaut : *MasterPage.master*
  - Observez la page générée dans l'éditeur de Source : C'est une page identique à toute autre page, excepté :
    - a. L'extension de fichier *.master* au lieu de *.aspx*.
    - b. La directive de page *@ Master* au lieu de *@ Page*.
    - c. Elle peut contenir un ou plusieurs contrôles *<asp:ContentPlaceholder>* destinés à recevoir le contenu personnalisé des futures pages enfants.
  - Basculez en mode Design. La page est vide et contient uniquement par défaut un contrôle ContentPlaceholder.
2. Ajoutez une référence à la feuille de style :
  - En mode *Source*, ajouter une balise *<link>* à l'intérieur de la balise **<head>** comme suit :
 

```
<head runat="server">
  <title></title>
  <link href="default.css" rel="stylesheet" type="text/css">
</head>
```
3. Dessinez les différentes zones de la page Maître encapsulées dans des balises *<div>* marquées par les classes de style correspondantes :
  - Toujours en mode **Source**, supprimez le contrôle *<asp:contentplaceholder>* et sa balise *<div>* englobante. La balise *<form>* est maintenant vide.
 

```
<body>
  <form id="form1" runat="server">
  </form>
</body>
```
  - ajoutez la zone d'en-tête de la page :

```
<!-- Zone : En tête -->
<div class="header">
  <!-- Zone : menu en haut de la page -->
  <div class="menu">Plan du site | Contactez-nous | Ajoutez aux favoris</div>
  <!-- Zone : panneau publicitaire -->
  <div class="rotator"></div>
  <!-- Zone : barre de navigation affichant le chemin de la page en cours -->
  <div class="nav"></div>
</div>
```

Ici, nous avons fait le choix d'écrire le menu en texte uniquement pour simplifier.

- Remarquez le contenu de la feuille de style pour chacune de ces zones :

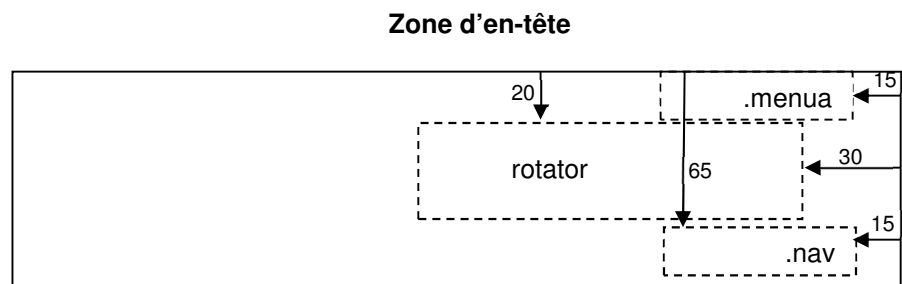
La première définit la zone d'en-tête dans sa globalité et surtout mais en place l'image de fond *header.gif* qui va démarquer par ses couleurs les différentes parties de l'en-tête :



```
.header :
{
  background-image: url(Images/header.gif);
  background-repeat: no-repeat;
  position: relative;
  width: 760px;
  height: 85px;
  margin-left: auto;
  margin-right: auto;
  font-weight: normal;
}
```

Les autres parties de l'en-tête se positionnent de manière absolue par rapport à l'image de fond :

```
.rotator
{
  position: absolute;
  right: 30px;
  top: 20px;
}
.nav
{
  position: absolute;
  top: 65px;
  right: 15px;
  color: #ffffff;
  font-size: 10px;
}
.menua
{
  position: absolute;
  right: 15px;
  top: 0px;
  color: #ffffff;
  font-size: 10px;
}
```



- ajoutez la zone centrale de la page :

```
<!-- Zone : Centre -->
<div class="page">
  <!-- Zone : Zone de navigation à gauche de la page -->
  <div id="sidebar">
    <!-- Zone : Espace membre -->
    <h1>Espace membre</h1>
    <div id="liensEspaceMembre"></div>
  </div>
  <!-- Zone : Services des annonces proposés par le site -->
  <h1>Nos Annonces</h1>
  <h2>Consultez toutes nos annonces et passez vos annonces en ligne</h2>
  </div>
  <!-- Zone : Zone contenant le contenu des pages enfants dans un contrôle
  <asp :contentplaceholder> -->
  <div id="content"></div>
</div>
```

- le contenu des classes dans la feuille de style est similaire à la zone d'en-tête à ceci près que les zones *sidebar* et *content* sont des *ID d'Elements* (précédés du signe #). Les IDs sont comme des classes, à la différence près qu'ils ne peuvent être utilisés qu'une seule fois dans un document html.

La classe *page* est la principale de cette zone centrale et définit le fond de l'écran : *body-repeat.gif*. L'image ne fait que 1 pixel de hauteur. En effet, le procédé consiste à faire en sorte que cette image très fine se répète sur l'axe des ordonnées de la page de façon dynamique en fonction du contenu qui sera placé dans le flux de celle-ci. Ainsi la hauteur de vos pages s'ajustera automatiquement en fonction de la quantité d'information qu'elles contiennent.

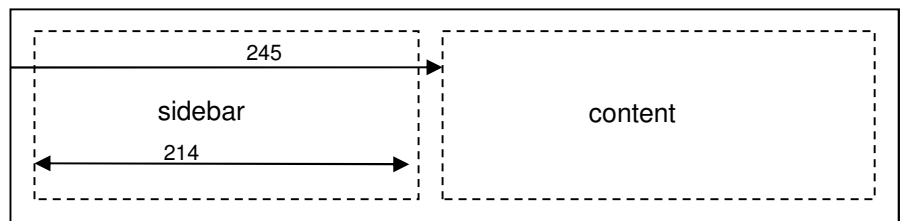
Essayez de produire ce même résultat en utilisant une structure à base de tableaux HTML plutôt que des balises <div>...prévoyez d'être occupé pendant un petit moment... !!

Zone de couleur soutenue sur laquelle on va positionner les éléments de menu du site et l'espace utilisateurs membres.



```
.page
{
  margin-left: auto;
  margin-right: auto;
  text-align: left;
  background-image: url(images/body-repeat.gif);
  background-repeat: repeat-y;
  position: relative;
  width: 712px;
  padding: 0px 25px;
}
```

Zone de corps de page



```
#sidebar
{
  float: left;
  width: 214px;
  height: 100%;
}
```

Indique le bord de la zone le long duquel le texte sera placé. Vous obtenez un **flux** de 214 pixels de large non limité en hauteur.

```
#content
{
  margin-left: 245px;
}
```

- enfin ajoutez la zone pied de page associée à la classe **footer** :

```
<!-- Zone : Pied de page -->
<div class="footer">
</div>
```

Là encore, vous pourriez définir une image de fond de façon à délimiter la zone de pied de page. Pour simplifier, nous n'en avons pas dessiné ici.

- Rester en mode *Source*.
- Faites un glisser déplacer à l'intérieur de la balise <div id= « content »> d'un contrôle *ContentPlaceholder* depuis la Boîte à outils > rubrique Standard.
- Renommez le contrôle : Main.

```
<div id="content">
    <asp:ContentPlaceholder ID="Main" runat="server">
    </asp:ContentPlaceholder>
</div>
```

- Ajoutez le panneau publicitaire : faites un glisser déplacer à l'intérieur de la balise <div class= « rotator »> d'un contrôle *AdRotator* depuis la Boîte à outils > rubrique Standard.

```
<div class="rotator">
    <asp:AdRotator ID="AdRotator1" runat="server"/>
</div>
```

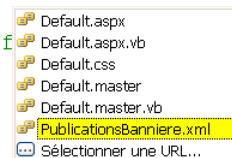
Ce contrôle serveur affiche des bannières de publicité à partir des informations lues dans un fichier XML respectant un formalisme particulier. Vous allez récupérer ici un fichier existant nommé *PublicationsBanniere.xml* dans le répertoire fourni.

- Faites un glisser déplacer du fichier *PublicationsBanniere.xml* depuis l'Explorateur de Windows sur la racine du projet dans l'Explorateur de solutions de VWD.
- Double cliquez sur le fichier *PublicationsBanniere.xml* pour l'ouvrir et regarder la structure d'un fichier de publicités.

Chaque balise <Ad> contient une bannière publicitaire avec ses informations, notamment l'url de l'image à afficher dans <ImageUrl> et le lien hypertexte à associer à l'image dans <NavigateUrl>. S'il y a plusieurs balises <Ad>, elles sont affichées de façon aléatoire à chaque requête sur la page (vous pouvez préciser le nombre de passage préférentiel dans une propriété <Impressions>).

- Revenez sur la page *MasterPage.master* en mode *Source* et rajoutez l'url vers le fichier d'informations publicitaires dans la propriété *AdvertisementFile* du contrôle.

```
<div class="rotator">
    <asp:AdRotator AdvertisementFile=
</div>
<!-- Zone : barre de navigation af
<div class="nav"></div>
</div>
```



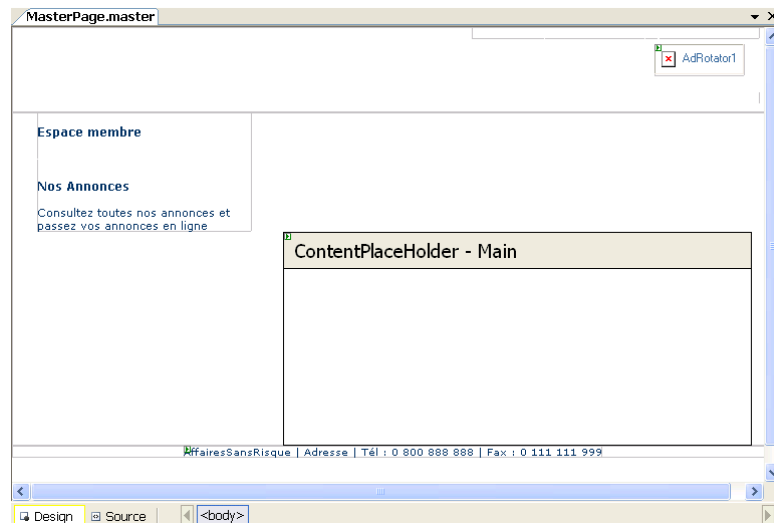
Soyez vigilant quant à la manière dont vous saisissez l'url des fichiers référencés dans la page maître parce que vous risquez d'avoir des surprises si votre page enfant n'est pas situé dans le même répertoire que votre page maître. Optez pour une url absolue avec le chemin complet du fichier, ou pour une url relative comme c'est le cas si vous laissez l'IntelliSense le faire pour vous :

```
<div class="rotator">
    <asp:AdRotator ID="AdRotator1"
    AdvertisementFile="~/PublicationsBanniere.xml" runat="server"/
</div>
```

- Ajoutez un texte simple en bas de page : faites un glisser déplacer à l'intérieur de la balise <div class=footer> d'un contrôle *Literal* depuis la Boîte à outils > rubrique *Standard*.
- Changez l'ID par défaut en *l1PiedDePage*.
- Ajoutez le texte comme suit dans la propriété *Text* :

```
<!-- Zone : Pied de page -->
<div class="footer">
    <asp:Literal ID="l1PiedDePage" runat="server"
    Text="AffairesSansRisque | Adresse | Tél : 0 800 888 888 | Fax : 0 111 111 999">
    </asp:Literal>
</div>
```

- Basculer maintenant en mode **Design** pour observer enfin le résultat de la construction de cette page Maître. Vous devez obtenir ceci :



Pour l'instant les différents flux sont vides et le designer ne vous montre pas le fond d'écran avec les images donc vous obtenez quelque chose d'assez peu convaincant. Mais n'ayez crainte, toute la structure de base de vos pages est bien en place. Vous allez maintenant créer des pages de contenu.

4. Essayez de lancer l'exécution de la page en faisant un clic droit sur *MasterPage.master* dans l'*Explorateur de solutions* > *Afficher dans le navigateur*. Vous constatez que l'option n'est pas disponible et qu'une page Maître ne peut pas s'exécuter seule (type de ressource interdite pour le serveur web). Elle n'a de sens que si une page de contenu en hérite, c'est pourquoi vous avez par contre l'option de menu *Ajouter une page de contenu* qui vous permettra de créer une page enfant



## 4 Créer des pages de contenu

### Contexte fonctionnel

Votre application contient déjà une page Default.aspx. Vous allez la modifier pour qu'elle hérite de la page Maître. Imaginons également que le pied de page de cette page par défaut, propose un texte différent qui suggère à l'utilisateur de remplir une enquête de satisfaction sur les services proposés par l'entreprise.



Nous vous proposons ensuite de créer une nouvelle page de contenu, nommée **QuiSommesNous.aspx** à partir de la page Maître. Son contenu est une description de l'entreprise AffairesSansRisque.



### 4.1 Transformer une page existante

Déroulement de l'exercice :

- Ouvrez la page Default.aspx en mode Source :
  - Double cliquez la page *Default.aspx* depuis **l'Explorateur de solutions**.
- Modifiez la directive `@Page` pour indiquer au runtime que votre page hérite maintenant de la page Maître *MasterPage.master* :
  - Ajoutez l'attribut `MasterPageFile` et pointez sur l'url de la page Maître.

`<%@ Page MasterPageFile=` `<%@ Page MasterPageFile="~/MasterPage.master"`

MasterPage.master  
Choisir un maître...

Vous pouvez aussi décider de faire le lien avec la page Maître directement dans le fichier de configuration *web.config* de sorte que l'héritage se fasse automatiquement au niveau de toutes les pages de l'application.

- Vous devez ensuite transformer la structure de la page qui n'a plus de balise `<html>` puisqu'elle va déjà être incluse dans celle de la page Maître :

- Supprimez tout le contenu HTML de la page en ne gardant que la directive `@Page` et le contrôle `Label` que vous aviez dessiné :
4. Une page enfant n'a le droit que de remplir les zones de contenu délimitées par les contrôles `ContentPlaceholder` de la page Maître. Dans une page enfant, cette zone de contenu dynamique est un contrôle `<asp:Content>` qui est relié à un contrôle `ContentPlaceholder` parent par l'attribut `ContentPlaceholderID` :
    - Autour du contrôle `Label`, ajoutez un contrôle `<asp:Content>` (depuis la Boîte à outils > rubrique Standard ou manuellement).

```

1  <%@ Page MasterPageFile="~/MasterPage.master" AutoEventWireup="false"
2
3  <asp:Content ID="Content1" ContentPlaceHolderID="Main"
4  Runat="Server">
5      <asp:Label ID="Label1" runat="server" Text="Label">
6      </asp:Label>
7  </asp:Content>
8  |

```

**Main** est l'ID du contrôle serveur `ContentPlaceholder` de votre page Maître **MasterPage.master**

Notez qu'une page enfant n'est pas obligée de mettre du contenu dans tous les contrôles `ContentPlaceholder` de sa page Maître. Et un contrôle **Content** ne peut pas être lié à plusieurs contrôles `ContentPlaceholder` en même temps.

5. Basculez la page en mode *Design* pour observer le résultat visuel :



Vous remarquez que le contenu défini par la page Maître est en lecture seule. Vous n'avez accès qu'au contenu du contrôle `Content1`, le reste du graphisme étant hérité de la page Maître. Pour changer le contenu de la page Maître, il faut modifier directement celle-ci. L'intérêt est double : cela simplifie la maintenance du site puisque vous savez que vos modifications sur la page Maître seront impactées en même temps sur toutes les pages enfants concernées ; cela minimise également la compilation puisque seule la partie dynamique des pages enfants sera remise en cause.

6. Sauvegardez la page
7. Exécutez la page :
  - Faites un clic droit en mode **Design** > **Afficher dans le navigateur**.


## 4.2 Accéder aux objets de la page maître depuis la page enfant

Déroulement de l'exercice :

Pour changer le pied de page défini dans la page maître dynamiquement depuis la page enfant, il faut que la page maître expose une propriété *PiedDePage* qui permette d'accéder au contrôle <asp:Literal> d'ID *ltlPiedDePage* de la zone pied de page.

```
<!-- Zone : Pied de page -->
<div class="footer">
    <asp:Literal ID="ltlPiedDePage" runat="server"
    Text="AffairesSansRisque | Adresse | Tél : 0 800 888 888 | Fax : 0 111 111 999">
    </asp:Literal>
</div>
```

1. Affichez la page de code de la page maître :

- Depuis l'*Explorateur de solutions*, cliquez sur *MasterPage.master* et sur l'icône  Afficher le code :

2. Ajoutez une propriété publique *PiedDePage* qui renvoie la valeur de la propriété *Text* du contrôle *ltlPiedDePage* :

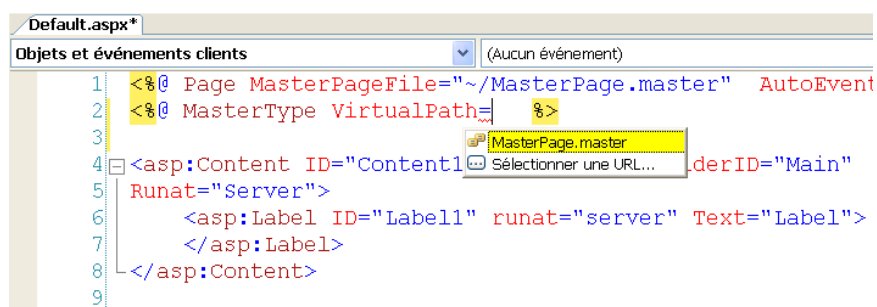
#### Code C# :

```
public partial class MasterPage : System.Web.UI.MasterPage
{
    public String PiedDePage
    {
        get
        {
            return ltlPiedDePage.Text;
        }
        set
        {
            ltlPiedDePage.Text = value;
        }
    }
}
```

3. Sauvegardez le fichier.

4. Pour accéder à cette propriété depuis la page enfant, il faut d'abord créer dans cette dernière une référence fortement typée sur la page maître correspondante :

- Double cliquez sur *Default.aspx* depuis l'**Explorateur de solutions**.
- En mode source, ajoutez la directive *@MasterType* à la page pour indiquer le type de la page maître dont la page hérite :



```
1 <%@ Page MasterPageFile="~/MasterPage.master" AutoEvent
2 <%@ MasterType VirtualPath="~/MasterPage.master" %>
3
4 <asp:Content ID="Content1" ContentPlaceHolderID="Main"
5   Runat="Server">
6     <asp:Label ID="Label1" runat="server" Text="Label">
7     </asp:Label>
8 </asp:Content>
9
```

- Sauvegardez la page. *La sauvegarde est très importante* si vous voulez que *Visual studio* mémorise ce typeage fort pour que l'*IntelliSense* vous aide dans le point suivant !

5. Ajoutez le code de modification du pied de page dans la page enfant :

- Affichez le *code behind* de la page *Default.aspx*.
- Positionnez le curseur juste après le remplissage du *Label1* dans la procédure *Page\_Load* de façon à changer le pied de page dès le chargement de la page.
- Accédez directement à la propriété *PiedDePage* en utilisant la propriété *Master* de la classe *Page* et saisissez le texte suivant :

"Merci de prendre quelques instants pour répondre à notre <a href=EnqueteSatisfaction.aspx>enquête de satisfaction</a>"

Lien hypertexte vers une page *EnqueteSatisfaction.aspx* que vous créerez à l'exercice 5.

Vous obtenez :

### Code C#

```
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Label1.Text = "Bienvenue sur le N°1 des sites de bonnes affaires s;
        //Personnalisation du Pied de page
        Master.PiedDePage="Merci de prendre quelques instants pour répondre;
    }
}
```

Sauvegardez le fichier.

6. Exécutez la page *Default.aspx* pour observer le changement dynamique de pied de page.

## Créer une nouvelle page de contenu

Déroulement de l'exercice :

1. Créez une nouvelle page de contenu :
  - Faites un clic droit sur le répertoire du projet dans l'*Explorateur de solutions* > *Ajouter un nouvel élément...*
  - Sélectionnez le modèle "Web Form" et cochez les deux cases à cocher. Nommez la page : *QuiSommesNous.aspx*.

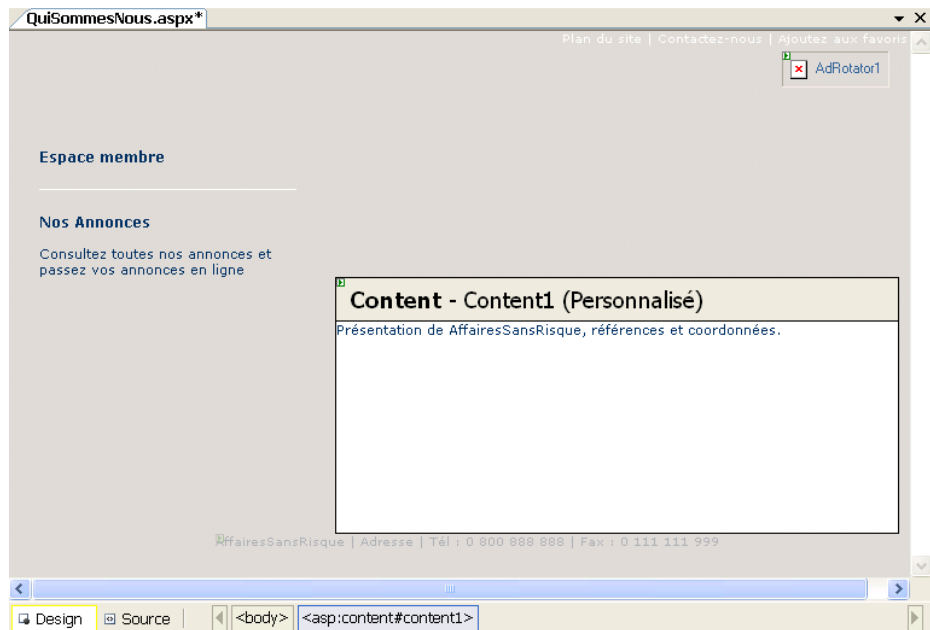
Vous constatez que vous pouvez écrire une page de contenu dans un langage différent de la page Maître dont elle hérite.

- Cliquez sur *Ajouter*.
- Dans la fenêtre *Sélectionner une page maître*, cliquez sur votre page *MasterPage.master*.

C'est dans cette fenêtre que vous pourriez préciser de quelle page maître vous souhaitez hériter dans le cas où vous en auriez défini plusieurs dans votre application.

- Cliquez sur OK.
- Observez la page générée en mode *Source* : vous obtenez directement une page avec un contrôle <asp :Content> lié au contrôle <asp :ContentPlaceHolder> d'ID=*Main* de la page Maître, comme précédemment pour la page *Default.aspx*.

2. Ajoutez du contenu à la page :
  - Basculez en mode *Design*.
  - Ajoutez en tapant directement dans le contrôle *Content1* la présentation de l'entreprise ou plus simplement un texte comme suit :



Vous pouvez faire un glisser déplacer de n'importe quel contrôle de la **Boîte à outils** sur le contrôle **Content1** pour dessiner la page.

3. Sauvegardez la page

4. Exécutez la page :

- Faites un clic droit en mode **Design > Afficher dans le navigateur**

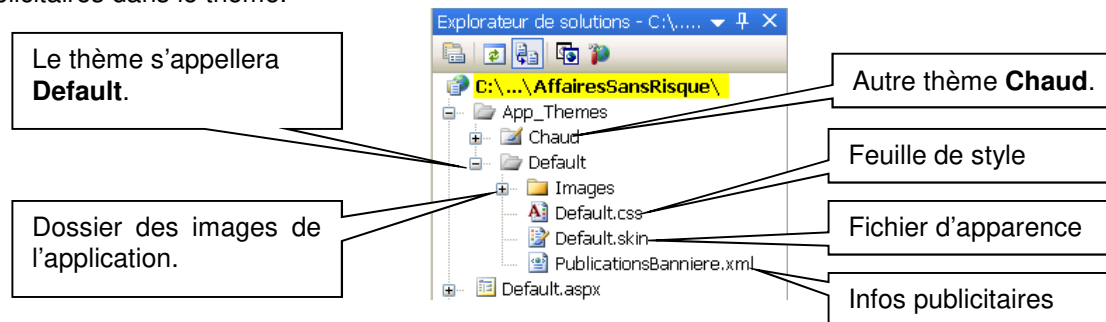
1. Notez que le pied de page est bien celui défini par défaut dans la page maître.

## 5 Utiliser un thème

### Contexte fonctionnel

Sur la base de la feuille de style utilisée précédemment, l'idée est de regrouper tous les éléments définissant la charte graphique de l'application dans un thème.

Pour l'instant, l'application n'utilise pas encore beaucoup de contrôles serveurs, mais nous verrons comment inclure également la définition de ceux-ci dans un fichier d'apparence dans le même thème. Ce sera le cas notamment du contrôle AdRotator de la page maître pour lequel vous allez déplacer les informations publicitaires dans le thème.



### 5.1 Isoler les éléments de la présentation dans un thème

Déroulement de l'exercice :

1. Créez un nouveau thème :
  - Faites un clic droit sur la racine de votre projet dans l'Explorateur de solutions > *Ajouter le dossier ASP > Thème.*
  - Nommez le thème : *Default* (ce sera le thème par défaut de votre application).

Que s'est-il passé ?

Visual studio a créé un répertoire très spécial avec le nom prédéfini *App\_Themes*. C'est là que ASP.NET s'attend à trouver tous les thèmes utilisés par votre application.

Un thème particulier est lui-même un répertoire (que vous avez appelé ici *Default*) dans lequel vous allez copier tous vos éléments graphiques, tels que la feuille de style et votre dossier *Images*.

2. Déplacez les éléments graphiques dans le dossier *Default* :
  - Faites un glisser déposer de la feuille de style *default.css* et du répertoire *Images* sous le dossier *Default*.

Vous venez de créer un thème de niveau application. Sachez que vous pouvez définir également des thèmes globaux de niveau machine (disponible pour toutes les applications) dans le répertoire d'installation du Framework >= 3.0.

3. Un fichier de configuration pour votre application doit exister. Si ce n'était pas le cas suivre la procédure suivante :
  - Depuis l'*Explorateur de solutions*, faites un clic droit sur la racine du projet > *Ajouter un nouvel élément... > modèle fichier de configuration Web.*
  - Cliquez sur *Ajouter.*
4. Appliquez le thème à l'ensemble de l'application depuis le fichier de configuration :
  - Depuis l'*Explorateur de solutions*, double cliquez sur le fichier *web.config* pour l'éditer.
  - Ajoutez à l'intérieur de la balise `<system.web>` la balise `<pages>` donnant le nom du dossier Thème que vous souhaitez appliquer à l'application.

```
<system.web>
  <!-- Indique le thème applicable à toute l'application-->
  <pages theme="Default"/>
```

A partir de là, le runtime ASP.NET sait qu'il doit scruter le répertoire Default pour retrouver toute référence à un élément qui traite de la représentation graphique de l'application. C'est le cas de la feuille de style, pour laquelle vous pouvez supprimer la référence dans le fichier MasterPage.master. ASP.NET sait où la trouver tout seul !

Testez l'exécution de la page *Default.aspx* pour vérifier que le *runtime* retrouve la feuille de style.

Vous pourriez aussi décider d'appliquer le thème qu'à certaines pages uniquement. Pour cela il suffit de rajouter à la directive de page *@ Page* l'attribut *theme*.

## 5.2 Appliquer un autre thème

---

Déroulement de l'exercice :

1. Récupérez un thème existant depuis le répertoire fourni intitulé *Chaud* :
  - Depuis l'explorateur de Windows, faites un glisser déplacer du répertoire *Chaud* dans le dossier *App\_Themes* du projet dans **l'Explorateur de solutions**.
2. Appliquez le thème à l'ensemble de l'application depuis le fichier de configuration :
  - Depuis *l'Explorateur de solutions*, double cliquez sur le fichier *web.config* pour l'éditer.
  - Changez l'attribut *theme* de la balise *<pages>* avec le nouveau nom de thème : *Chaud*.

```
<system.web>
  <!-- Indique le thème applicable
  <pages theme="Chaud"/>
```

3. Exécutez la page *Default.aspx* pour tester le nouveau thème.

Le thème est similaire. Seules les images de fond, les couleurs de police et le pied de page sont différents. Vous pourriez par exemple proposer à l'utilisateur de choisir le thème de son application et le mémoriser dans son profil (

## 5.3 Définir un fichier d'apparence de contrôles

Il est fort à parier que la représentation de vos contrôles serveur se retrouve également d'une page à l'autre. Par exemple, une grille de données aura la même représentation graphique (couleur de fond, taille des polices, bordures etc...) pour toute votre application.

Il est possible de « factoriser » également la représentation graphique des contrôles à l'intérieur d'un thème au même titre que celle des autres éléments, en définissant celle-ci dans un *fichier d'apparence de contrôles* d'extension *.skin*.

Déroulement de l'exercice :

- Regardons de plus près le contrôle serveur *AdRotator* que vous avez positionné dans la page maître Affichez la page *MasterPage.master* en mode *Design*.
- Faites un clic droit sur le contrôle *AdRotator1* > *Propriétés*

Remarquez que le contrôle possède une propriété *CssClass* donc il peut faire référence à une classe de votre feuille de style, exactement comme une balise HTML *<div>*.

Mais en prime, votre contrôle possède des propriétés pour les informations de style les plus courantes, avec un type fort, telles que *BackColor*, *BorderColor*, etc... En fait, elles représentent un sous ensemble des propriétés de style HTML standard et sont directement exposées par la classe de base *System.Web.UI.WebControls.WebControl*. L'intérêt est que vous bénéficiez pour celles-ci de l'IntelliSense et de la vérification de type faite au moment de la compilation...

1. Récupérez un fichier d'apparence prédéfini dans ce qui est fourni
  - Faites un clic droit sur le répertoire *Default* dans *App\_Themes* > *Ajoutez un élément existant...*

- Sélectionnez *Default.skin* puis cliquez *Ajouter*.

## 2. Affichez le contenu du fichier *Default.skin* pour observer son contenu.

Vous auriez pu bien sûr construire votre fichier d'apparence .skin à partir d'un fichier vierge (un modèle est disponible dans la boîte de dialogue **Ajoutez un nouvel élément**). Vous constatez qu'un fichier d'apparence contient tout simplement la définition usuelle des contrôles dans le flux des pages web du type : `<asp :NomDuContrôle>`. A la différence près que le fichier d'apparence ne doit pas comporter d'ID pour les contrôles comme c'est le cas dans vos pages (l'ID n'a de sens que pour nommer l'instance d'un contrôle à l'intérieur d'une page).

Lorsque ASP.NET traite une requête sur une page contenant des contrôles serveur, s'il trouve une définition des contrôles dans le fichier d'apparence, il applique systématiquement la représentation que vous y avez décrite.

- Descendez dans le fichier d'apparence jusqu'à la définition du contrôle *GridView* qui vous servira dans l'atelier sur l'accès aux données. Vous remarquez qu'il y a deux définitions pour ce même contrôle, marquées d'un attribut *SkinId* différent pour les différencier. On parle alors **d'apparence nommée**. C'est ce qui vous permettra d'appliquer un style distinct selon le contexte de page dans lequel vous utiliserez le contrôle.

```
<asp:GridView SkinId="tableMaitre" CssClass="gridcontent" runat="
</asp:GridView>
<asp:GridView SkinId="tableDetails" CssClass="gridcontent" runat=
  <RowStyle BackColor="#e1eff7" ></RowStyle>
</asp:GridView>
```

- Pour appliquer un style particulier au contrôle, il suffira de préciser l'attribut **SkinId** au moment de la définition du contrôle dans la page comme le montre l'exemple ci-dessous. Si vous n'indiquez aucun *SkinId*, c'est la définition par défaut (c'est-à-dire celle ne précisant aucun *SkinId*) du fichier d'apparence qui s'applique.

```
<asp:GridView SkinId="tableMaitre" ID="GridView1" runat="server"
  <Columns>
    <asp:BoundField DataField="Ann_Telephone" HeaderText="T&#
    <asp:BoundField DataField="Ann_Email" HeaderText="Email"
    <asp:BoundField DataField="Ann_Ville" HeaderText="Ville"
    <asp:HyperLinkField DataNavigateUrlFields="Ann_Id" DataNa
      HeaderText="D&#233;tails" Text="D&#233;tails..." />
  </Columns>
</asp:GridView>
```

Sachez également que vous pouvez définir à l'intérieur d'un même thème autant de fichiers d'apparence que vous le souhaitez. Ceux-ci peuvent contenir deux définitions différentes pour un même contrôle à condition de préciser des *SkinId* distincts. Vous pouvez alors classer les contrôles dans ces fichiers d'apparence différents par zones fonctionnelles de votre site web, ou par catégories de contrôles par exemple...

3. Le contrôle *AdRotator* définit dans le fichier d'apparence le chemin d'accès au fichier contenant les informations publicitaires. Il n'est donc plus utile de le définir dans la page *MasterPage.master*. Et vous pourriez imaginer utiliser un autre contrôle *AdRotator* ailleurs dans l'application avec un autre fichier d'information, auquel cas il suffirait de définir un *SkinId* distinct pour les deux définitions du contrôle dans votre fichier d'apparence.
  - Affichez la page *MasterPage.master* en mode *Source*.
  - Supprimer l'attribut *AdvertisementFile* dans la définition du contrôle *AdRotator*.
4. Déplacez le fichier d'informations publicitaires *PublicationsBanniere.xml* dans le répertoire de votre thème de façon à le rapprocher des images auxquelles il fait référence :
5. Exécutez la page **Default.aspx** pour vérifier que le runtime retrouve bien le chemin du fichier d'informations publicitaires grâce au fichier d'apparence.



6. Vous pouvez empêcher ASP.NET d'appliquer la représentation donnée dans le fichier d'apparence pour un contrôle (ou pour la page) en désactivant l'utilisation des thèmes pour le contrôle (ou pour la page) avec l'attribut *EnableTheming* positionné à *false*.
- Ajoutez l'attribut *EnableTheming=false* sur le contrôle *AdRotator*.

```
<!-- Zone : panneau publicitaire -->  
<div class="rotator">  
    <asp:AdRotator ID="AdRotator1" EnableTheming="false" runat="server"/>  
</div>
```

- Re-testez la page *Default.aspx* en exécution. Le contrôle ne sait plus où sont les informations publicitaires.

Pour que le message de bienvenue apparaisse comme ci-dessus sur toute la largeur de la partie dynamique, supprimez également l'application des thèmes au contrôle `<asp:label>` correspondant dans la page **Default.aspx**.

```
<asp:Label ID="Label1" runat="server" Text="Label"  
    EnableTheming="false"></asp:Label>
```

- Supprimez l'attribut *EnableTheming=false* dans le contrôle *AdRotator* pour réactiver l'utilisation du thème.