Enoncé complémentaire 3

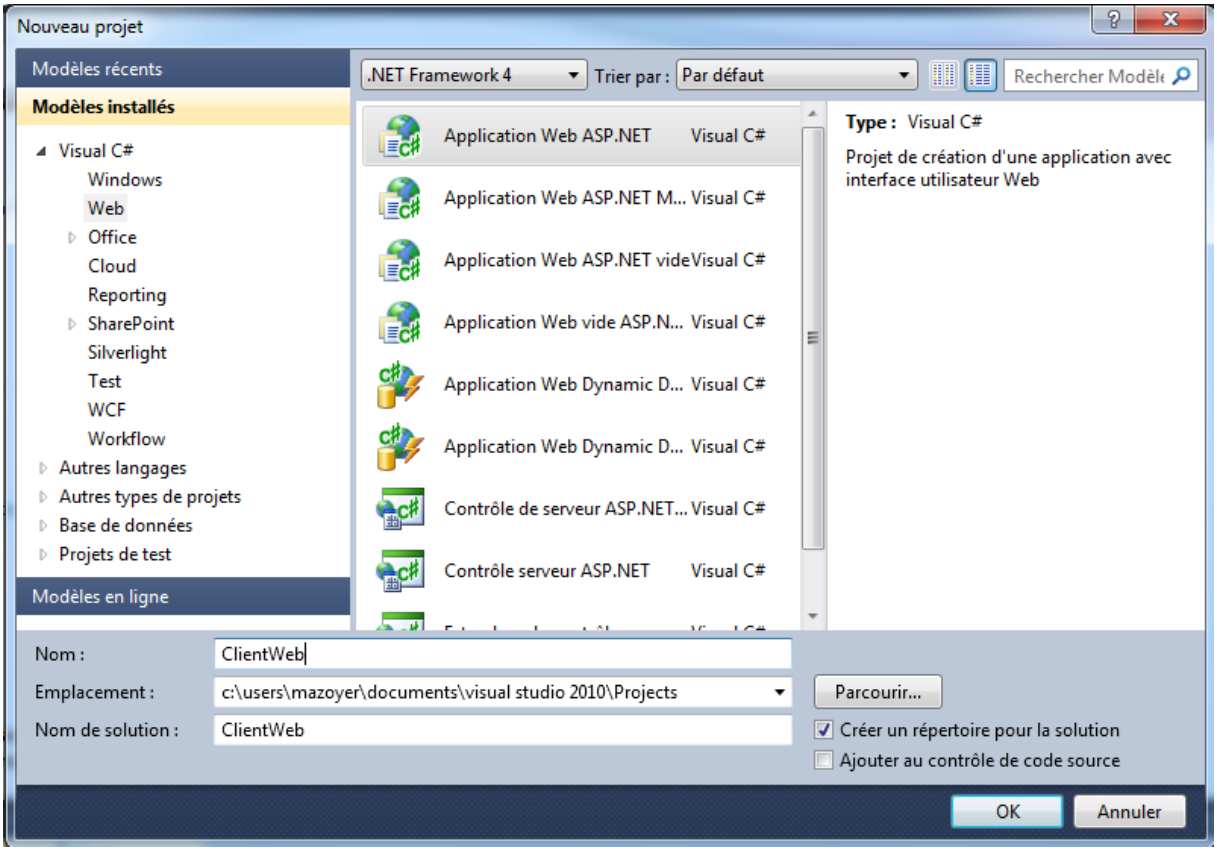
Tips :

1. Récupérer et visualiser une image dans une page web ASP.NET

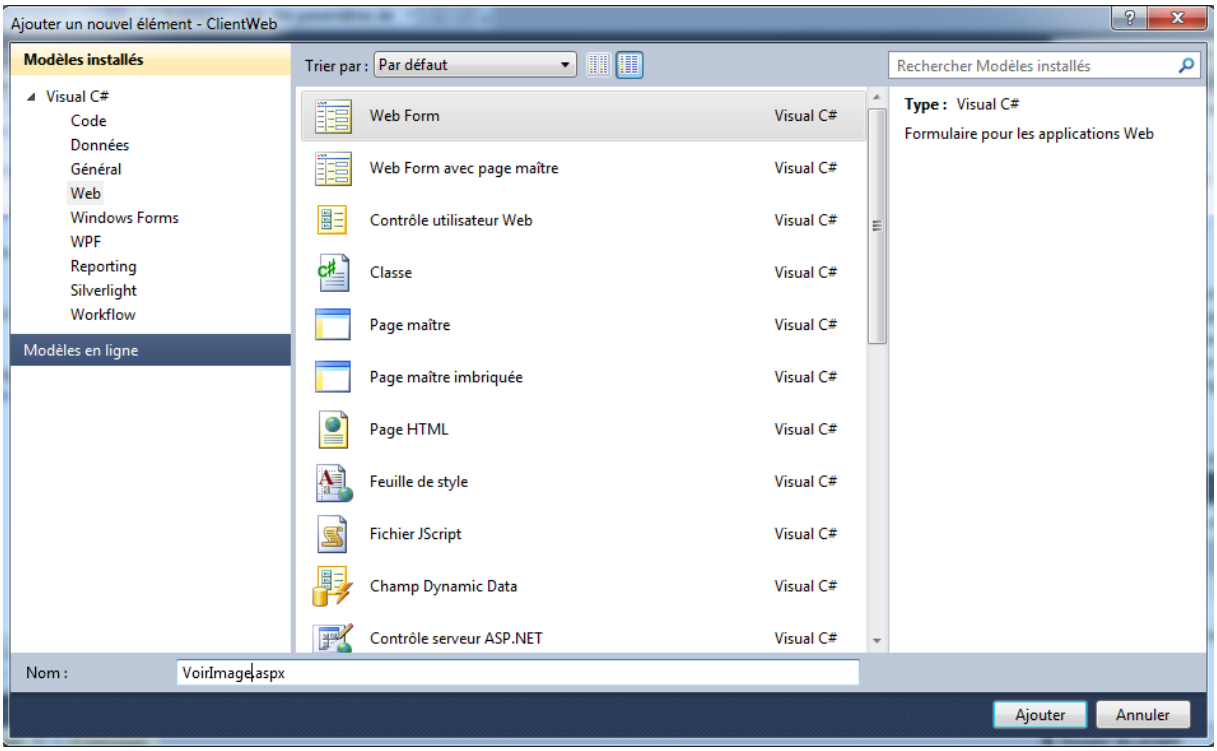
ASP.NET est un ensemble de technologies web basées sur le framework .NET qui permettent de créer des applications web dynamiques.

1.1. Création de la page VoirImage.aspx

Créer un nouveau projet de type application Web ASP.NET :



On y ajoute une page de type Web Form :

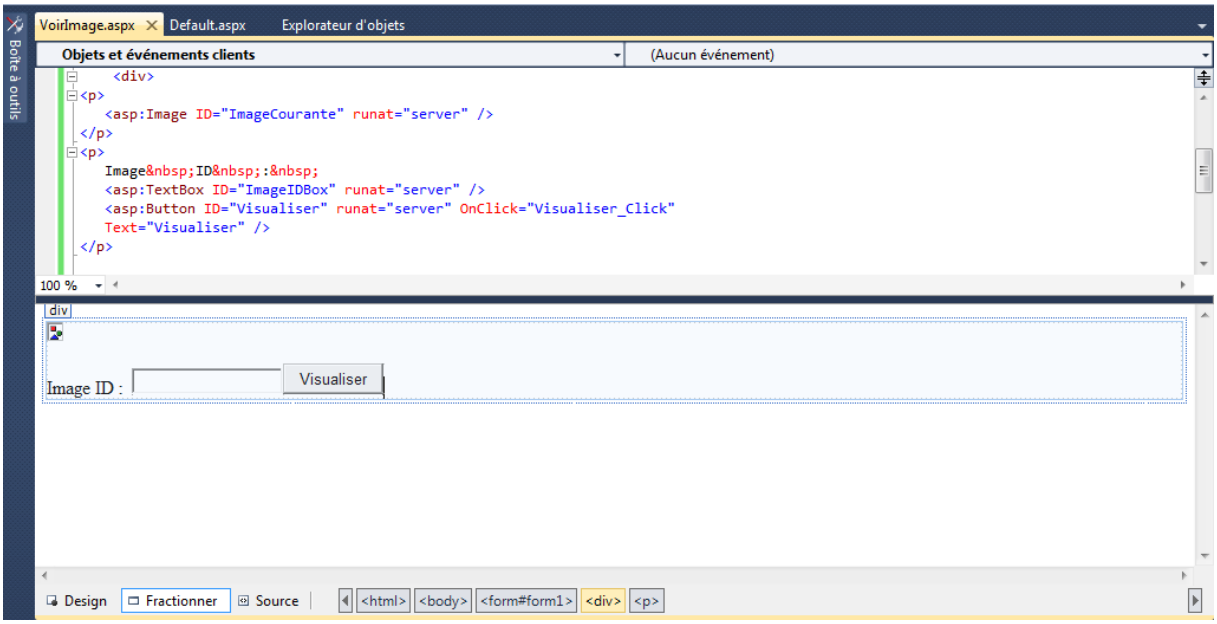


Dans la page créée, on ajoute ce code dans la div du form1 :

```
<p>
   <asp:Image ID="ImageCourante" runat="server" />
</p>
<p>
   Image ID : 
   <asp:TextBox ID="ImageIDBox" runat="server" />
   <asp:Button ID="Visualiser" runat="server" OnClick="Visualiser_Click"
   Text="Visualiser" />
</p>
```

Notre page web sera traitée sur le serveur afin de générer une page html pour le navigateur. Cette page contient du code html ainsi que des composants ASP.Net. Une liste des composants ASP.NET ainsi que leur description est disponible ici : ditch.ftp-developpez.com/AspNet2.pdf. Il est possible de choisir la fenêtre Design afin de visualiser les composants de notre page en mode graphique. Deux façons s'offrent au programmeur pour rajouter des nouveaux composants : ajouter leur code dans le mode Source ou en faire un glisser-déposer à partir de la boîte à outil dans le mode Design.

Le mode Fractionner est un mode contenant les fenêtres Design et Source :



On a ici un composant qui affichera notre image (ImageCourante), une TextBox (ImageIDBox) où l'utilisateur tapera l'ID de l'image qu'il veut visualiser et un bouton (Visualiser) qui lancera la récupération de l'image sur le serveur. Plusieurs attributs, tel que Text, qui permet de spécifier le contenu de notre bouton et OnClick, qui contient le nom de la méthode (Visualiser_Click) qui sera exécutée sur le serveur lorsque l'utilisateur cliquera sur notre bouton. Nous allons maintenant implémenter le code de cette méthode. Pour cela, on va éditer le fichier VoirImage.aspx.cs. Ce fichier contient pour l'instant la méthode Page_Load qui sera exécuté lors du chargement de notre page. Nous rajoutons donc la méthode Visualiser_Click :

```
protected void Visualiser_Click(object sender, EventArgs e)
{
      ImageCourante.ImageUrl = "Image.aspx?ImageID=" + ImageIDBox.Text;
}
```

Dans cette méthode, nous pouvons changer les propriétés de nos différents composants ASP.NET définis dans notre page. Ici, on veut changer l'URL de l'image de notre composant ImageCourante. Dans notre cas, notre image n'est pas une ressource statique présente sur le serveur, mais doit être récupérée sur notre base de données.

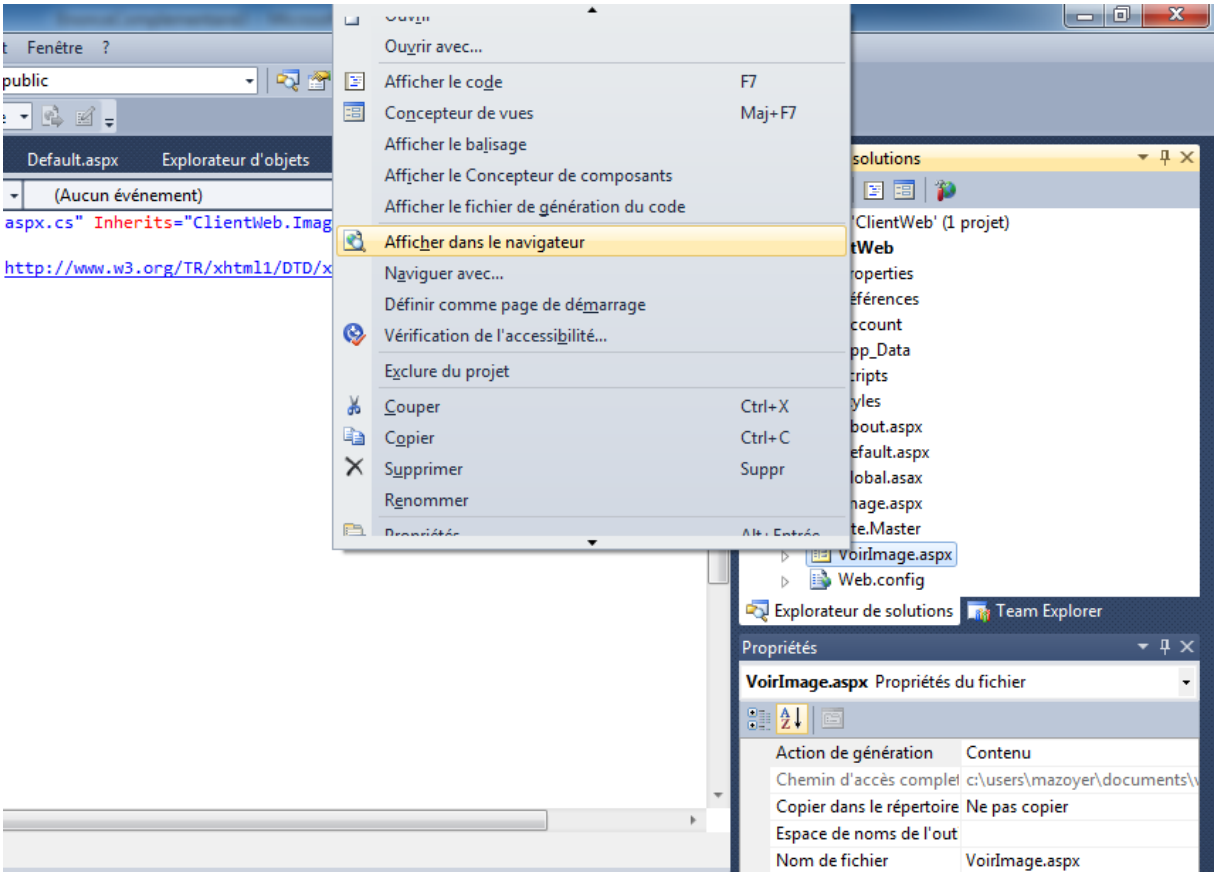
1.2. Création de la page Image.aspx

Nous allons donc créer une nouvelle Web Form, Image.aspx, qui va nous permettre de récupérer notre image. Nous allons juste modifier le fichier Image.aspx.cs de notre page :

```
protected void Page_Load(object sender, EventArgs e)
{
        // On récupére la valeur du paramètre ImageID passé dans l'URL
        String id = Request.QueryString["ImageID"];
        // Si ce paramètre n'est pas nul
        if (id != null)
        {
            // on récupére notre image là où il faut
            Byte[] bytes = XXXXXX.getImage(id);
            // et on crée le contenu de notre réponse à la requête HTTP
            // (ici un contenu de type image)
            Response.Buffer = true;
            Response.Charset = "";
            Response.Cache.SetCacheability(HttpCacheability.NoCache);
            Response.ContentType = "image/jpeg";
            Response.BinaryWrite(bytes);
            Response.Flush();
            Response.End();
        }
}
```

L'URL http://hostname:port/Image.aspx?ImageID=image1 va donc renvoyer un fichier image comme s'il s'agissait d'une ressource statique. (NB : Ici, pour des soucis de clarté, nous avons créé une nouvelle page Image.aspx mais il était aussi possible de réutiliser la page VoirImage.aspx).

Nous pouvons maintenant tester nos pages avec un clique-droit sur VoirImage.aspx->Afficher dans le navigateur :



3. Utilisation d'une variable de session

On crée une nouvelle page Coucou.aspx. Dans la div de notre form 1 on rajoute le code suivant :

```
<p>
     Salut
     <%
          // Si la variable de session user est non nulle,
          // on "écrit" sa valeur dans la page HTML que l'on génère
          if (Session["user"] != null)
          {
              Response.Write(Session["user"]);
          }
          else
          {
              Response.Write("inconnu");
          }
     %>
</p>
<p>
     Nom :
     <asp:TextBox ID="UserTextBox" runat="server" />
     <asp:Button ID="UserBouton" runat="server" OnClick="Authentifier_Click"
     Text="Ok" />
</p>
```

On remarque ici les balises <% et %> qui permettent d'insérer du code source C#. Ce code sera exécuté sur le serveur au moment de la génération de la page. On remarque aussi l'utilisation de Session["user"], qui contient la variable de session user. Les variables de session sont liées à un utilisateur et sont stockées sur le serveur le temps de la navigation de cet utilisateur. Si la valeur de la variable de session user n'est pas défini, le mot inconnu sera donc ajouté au code html généré par le serveur. Lors du premier appel de la page, la page html envoyé au client contiendra donc la phrase « Salut inconnu ».

Nous allons maintenant implémenter la méthode Authentifier_Click qui permettra modifiera la valeur de la variable de session user avec la valeur rentrée par l'utilisateur lorsque celui-ci cliquera sur le bouton UserBouton. Voici le code de cette méthode :

```
protected void Authentifier_Click(object sender, EventArgs e)
{
      Session["user"] = UserTextBox.Text;
}
```