Concepteur Développeur en Informatique

Développer des composants d'interface

Présentation ASP Net Pages Maitres

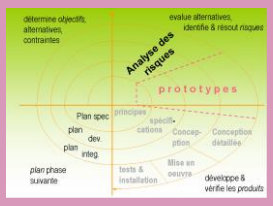
Accueil

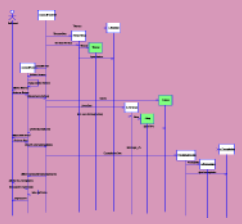
Apprentissage

PAE

Evaluation

Impossible d'afficher l'image.





Localisation : U03-E05-S03

SOMMAIRE

1. Introduction

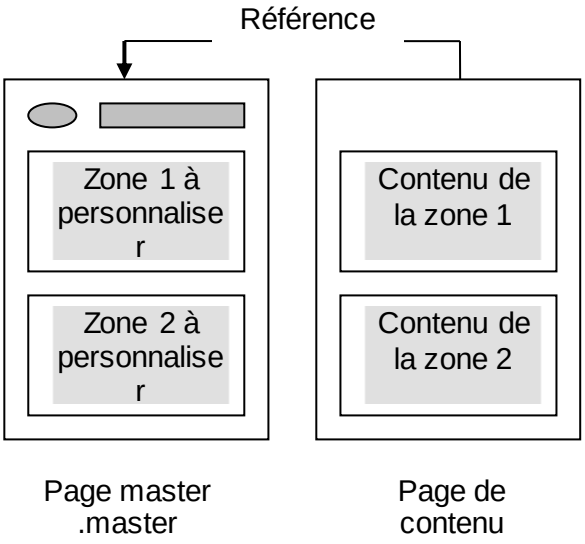
Ce document présente l'organisation des contenus grâce au recours aux pages maitre et pages de contenu.

Nous avons besoin de pouvoir afficher de nombreuses pages d'aspect similaire, partageant des graphiques, des éléments d'interface utilisateur et des menus de navigation ou des formulaires de recherche.

Pour organiser les contenus et assurer une cohérence de rendu sur l'ensemble du site, nous disposons des pages maîtres. Celles-ci nous permettent de réutiliser simplement un modèle de design de page.

1.1. Qu'est-ce qu'une page maître

Une page maître permet de définir l'aspect des pages de contenu qui dépendent d'elle.
Une page maître porte l'extension .master.
Une page maître organise l'architecture des contenus.
Elle comporte au moins un contrôle ContentPlaceHolder qui contiendra les pages de contenu en son sein.
Les pages de contenus qui seront affichées dans les ContentPlaceHolder sont des pages asp.x.



Un contrôle *ContentPlaceHolder* définit une région de la MasterPage susceptible d'être personnalisée dans une page dérivée.
Une MasterPage dépourvue de ces contrôles fonctionne comme une page ordinaire, avec la charge de traitement supplémentaire des MasterPages.

1.2. Créer une page Maître

Une MasterPage est similaire à une page ASP.Net ordinaire, à l'exception de la directive @Master, et de la présence d'un ou de plusieurs contrôles serveur ContentPlaceHolder.

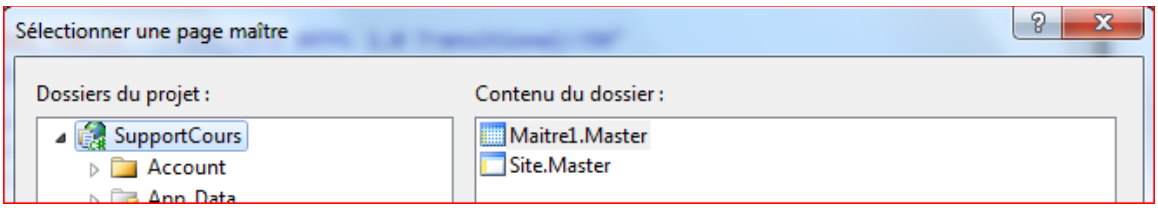
La directive @Master prend en charge quelques attributs, en grande partie ceux de la directive @Page.

```
<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="Maitre1.master.cs"
Inherits="SupportCours.Maitre1" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <asp:ContentPlaceHolder ID="head" runat="server">
    </asp:ContentPlaceHolder>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">

        </asp:ContentPlaceHolder>
    </div>
    </form>
</body>
</html>
```

Le code affiché ci-dessus correspond à celui généré par défaut lors de la création d'une nouvelle page maître.
A la différence d'une page classique aspx nous trouvons des zones réservées pour les contenus.
Une zone pour les contenus d'entête et une pour les contenus du corps de page.

1.3. Créer la page de contenu associée

Le plus simple est d'ajouter un nouveau document de type Web Form associé à une page maître. Il vous est alors proposé de choisir la page maître à associer.



Vous pourrez vérifier que la page de contenu ne contient que des éléments de contenu et aucun bloc HTML ou FORM. En fait ces derniers sont dans la page maître.

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Maitre1.Master"
AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inherits="SupportCours.WebForm1" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
</asp:Content>
```

Les éléments de contenu sont générés avec la référence des ContentPlaceHolder de la page maître.

Une page de contenu n'associe pas forcément un contrôle <asp:Content> à chaque contrôle <asp:contentplaceholder>.de sa masterpage.
En outre, la définition d'une MasterPage peut inclure des contrôles <asp:Content> associés à ses propres contrôles <asp:contentplaceholder>. Ce contrôle sera alors utilisé par défaut lorsque la page de contenu ne fournit pas le contrôle adéquat.

A l'intérieur d'un contrôle asp:Content, nous définirons tous les contrôles de la page de contenu :
Dans ce premier exemple simpliste, deux contrôles ont été déposés dans la zone de contenu, un label et une boite de texte.

```
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <asp:Label ID="Label1" runat="server" Text="Label">Votre Nom</asp:Label>
    <asp:TextBox ID="txtNom" runat="server"></asp:TextBox>
</asp:Content>
```

A noter : En code behind, les contrôles sont toujours accessibles via la référence d'une variable portant le nom de l'identifiant du contrôle. Mais côté client, en javascript, les contrôles seront accessibles via un nom généré constitué de l'association de l'Id du ContentPlaceHolder et de l'ID du contrôle. Ainsi, le contrôle TxtNom sera accessible via la référence ContentPlaceHolder1_TxtNom.

Extrait de la page générée.

```
<span id="ContentPlaceHolder1_Label1">Votre Nom</span>
<input name="ctl00$ContentPlaceHolder1$txtNom" type="text" value="Bost" id="ContentPlaceHolder1_txtNom" />
```

1.4. Imbriquer des pages maîtres

Il est possible de créer des pages maîtres imbriquées.
Visual studio pour propose de vous aider à générer celle-ci en choisissant comme modèle Page Maitre imbriquée.

1.5. Accéder aux contrôles définis sur la MasterPage

Deux options :

- Par le biais d'une propriété définie comme publique dans la MasterPage
- Par la méthode FindControl de la page maître

Il est possible d'accéder au contenu d'un contrôle défini sur la MasterPage, directement à partir d'une page de contenu.
En définissant sur la MasterPage , une propriété publique retournant la propriété Text du contrôle en question.

Par exemple, si nous souhaitons accéder au contrôle situé dans la master page défini ci-dessous :

```
<asp:Literal ID="lContenu" runat="server"></asp:Literal>
```

Il nous faudra déclarer une propriété avec des accesseurs publiques dans la page maitre.
On expose la propriété Text de txtSaisie, comme propriété publique ainsi :

```
public string StrSaisie
{
get { return txtSaisie.Text; }
set { txtSaisie.Text = value; }
}
```

Pour utiliser cette propriété:
On ajoute une directive @MasterType dans la page de contenu

```
<%@ MasterType VirtualPath="~/Maitre.master" %>
```

Qui nous permettra d'utiliser la propriété Master : cette propriété représente l'objet MasterPage compilé par l'environnement d'exécution.

```
protected void Page_Load(object sender, System.EventArgs e)
    {
    Master.StrSaisie = "Bonjour !";

    }
```

Vous pouvez aussi utiliser la méthode FindControl().
A noter : Si le contrôle est placé dans un conteneur enfant de la page, il faut d'abord récupérer une référence à ce conteneur puis effectuer une recherche du contrôle dans ce conteneur.

```
private void RechercherControle()
{
    ContentPlaceHolder masterConteneur;
    TextBox masterControle;
    masterConteneur = (ContentPlaceHolder)Master.FindControl("ContentPlaceHolder1");
    if (masterConteneur != null)
    {
        masterControle = (TextBox)masterConteneur.FindControl("txtNom");
        if (masterControle != null)
        {
            masterControle.Text = "Contrôle master mis à jour";
        }
    }
}
```