Globalisation et accessibilité

1 Introduction

Jusqu'à présent nous avons vu de nombreux outils qui vont nous permettre de créer une application Web des plus fournies. Ceci étant, une application aussi complète soit-elle n'a que peu de valeur sur un marché international si elle n'est disponible que dans une seule langue. De plus il est généralement préférable de créer une application donc le fonctionnement sera similaire pour tous les navigateurs et dont l'affichage ne sera pas perturbé d'un navigateur à l'autre.

Au cours de ce chapitre nous verrons comment rendre une application Web multilingue et accéssible d'une manière générale.

2 Une Application Multilingue

Pour rendre une application multilingue il existe différents moyens, par exemple, la première chose qui vient à l'esprit est de créer autant de page web qu'il y a de langue supportées par notre application. Par exemple :

Default.aspx	:	Page par défaut en anglais
Default.fr.aspx	:	Page en français
Default.es.aspx	:	Page en espagnol
Default.ge.aspx	:	Page en allemand

Néanmoins ce procédé présente deux gros désavantages :

- En cas de maintenance ou de mise à niveau, le temps d'intervention sera multiplié car il faudra intervenir sur chacune des pages du site.
- Pour un lourd projet, cette méthode peut faire perdre pas mal de performance à notre application.

Nous allons donc voir quels sont les outils qu'ASP.NET met à notre disposition pour créer une application multilingue de façon rapide et légère.

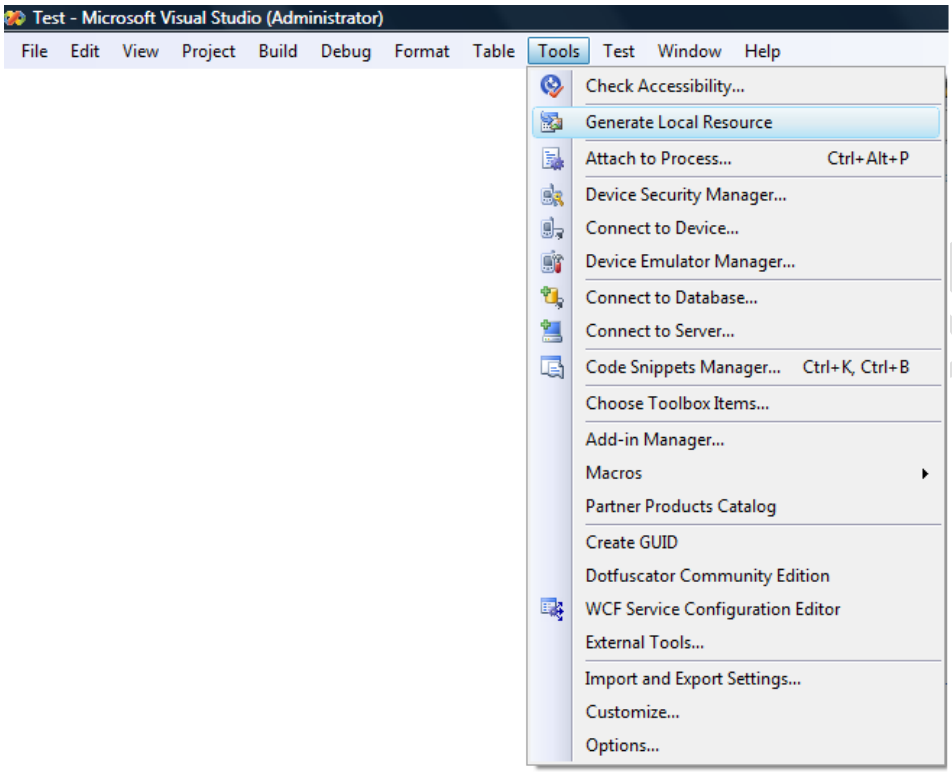
2.1 La localisation

Le principe de la localisation est de générer un fichier qui va contenir une référence de chacun de nos contrôles de notre page web ainsi qu'une propriété qui contiendra du texte. Ce fichier va s'adapter à la langue de notre navigateur en fonction de celles qu'on aura prévu. Dans le cas où nous avons affaire à un navigateur qui ne prévoit pas notre langue, alors c'est le document de référence par défaut qui sera utilisé.

Il est recommandé d'effectuer les étapes suivantes chaque fois que vous avez fini de coder une page de votre application Web.

2.1.1 Générer un fichier de ressources locales

Une fois que vous avez fini de coder votre page, dans le menu cliquez sur outil puis Générer la ressource locale.



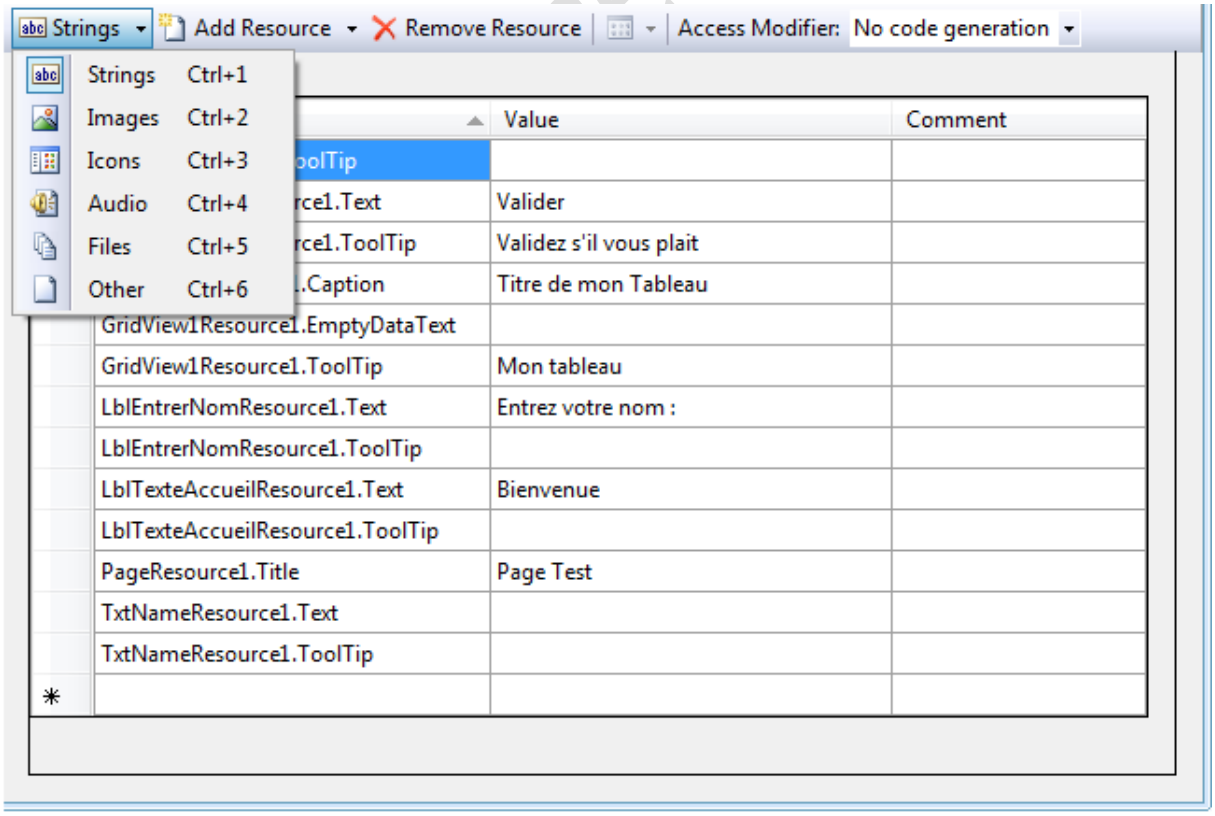
Si ce n'est pas déjà fait vous devrez créer le dossier *App_LocalResources*, à l'intérieur de ce dossier sera généré un fichier *.resx* portant le même nom que votre page : *default.apsx.resx* pour la page par défault par exemple. A l'intérieur de ce fichier sont répertoriés tous les contrôles que vous avez ajoutés dans votre page Web.

2.1.2 Editer le fichier de ressource locale

Une fois que vous avez généré ce fichier ouvrez le, voici comment il se présente :

Name	Value	Comment
AccueilResource2.ToolTip		
BtnValidNomResource1.Text	Valider	
BtnValidNomResource1.ToolTip	Validez s'il vous plait	
GridView1Resource1.Caption	Titre de mon Tableau	
GridView1Resource1.EmptyDataText		
GridView1Resource1.ToolTip	Mon tableau	
LblEntrerNomResource1.Text	Entrez votre nom :	
LblEntrerNomResource1.ToolTip		
LblTexteAccueilResource1.Text	Bienvenue	
LblTexteAccueilResource1.ToolTip		
PageResource1.Title	Page Test	
TxtNameResource1.Text		
TxtNameResource1.ToolTip		
*		

On peut remarquer que beaucoup propriétés de nos contrôles sont récupérées. La génération de ressource locale récupère toutes les propriétés texte de nos contrôles, mais aussi les propriétés des contrôles de type audio, images, icones, fichier, etc. tels que l'url par exemple.



Ainsi, vous pouvez compléter entre autre les propriétés que vous avez pu omettre.

On pourra constater que la propriété suivante a été ajoutée à vos contrôles concernés :

```
meta:resourcekey="NomDuControleResource1"
```

2.1.3 Mettre en place plusieurs langue pour votre application

Lors du chargement de la page, notre application détecte automatiquement la langue du navigateur. Il est alors possible de charger la langue adéquate en fonction des informations données par le navigateur. Cette fonctionnalité est simple à mettre en place, et se base sur la norme ISO 639-1. Pour ajouter une langue à votre application web, copiez votre fichier de ressource locale dans votre répertoire *App_LocalResources* puis renommez-le de la façon suivante :

(*nom de la page*).aspx.(*nom de la langue selon la norme ISO 639-1*).resx

Exemple :

Default.aspx.fr.resx

Puis rééditez les propriétés pour les mettre dans la bonne langue.

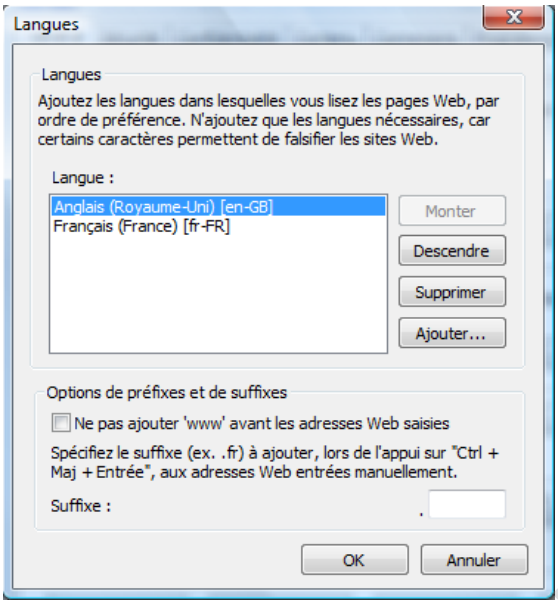
Vous pouvez consulter la liste des codes ISO 639-1 pour chaque langues à cette adresse :

http://fr.wikipedia.org/wiki/Liste_des_codes_ISO_639-1

Voici un aperçu de votre autre fichier de ressource locale en anglais : Default.aspx.en.resx

Name	Value	Comment
AccueilResource2.ToolTip		
BtnValidNomResource1.Text	Validate	
BtnValidNomResource1.ToolTip	Please valid	
GridView1Resource1.Caption	My table's title	
GridView1Resource1.EmptyDataText		
GridView1Resource1.ToolTip	My Table	
LblEntrerNomResource1.Text	Enter your name :	
LblEntrerNomResource1.ToolTip		
LblTexteAccueilResource1.Text	Welcome	
LblTexteAccueilResource1.ToolTip		
PageResource1.Title	Testing Page	
TxtNameResource1.Text		
TxtNameResource1.ToolTip		

Pour vérifier le bon fonctionnement de cette manœuvre, lancer votre application avec Internet Explorer, allez dans le menu Outils puis Option Internet. Dans l'onglet Général, choisissez Langues.



Faites monter l'anglais en premier (ajoutez-le si besoin). Validez et fermez le menu Option Internet. Actualisez la page. Vous pourrez constater que le texte en français auparavant est passé en anglais grâce à notre fichier Default.aspx.en.resx.

Bonjour à tous

Bienvenue

Entrez votre nom :

Valider

Titre de mon Tableau

id	nom	prenom	adresse	poste
2	BEDE	Nicolas	TOULOUSE	Developpeur
3	RONSIN	Cédric	TOULOUSE	Manager
4	DOLLON	Julien	TOULOUSE	Fondateur

Langue Française dans Option Internet

Hello you !

Welcome

Enter your name :

Validate

My table's title

id	nom	prenom	adresse	poste
2	BEDE	Nicolas	TOULOUSE	Developpeur
3	RONSIN	Cédric	TOULOUSE	Manager
4	DOLLON	Julien	TOULOUSE	Fondateur

Langue Anglaise dans Option Internet

Bien que cette méthode soit très pratique et légère, elle n'est effective que sur un seul fichier. Et pour avoir une traduction complète de notre site, il faudra créer autant de fichier *.resx* que nous avons de pages et de langues à supporter.

Cela dit, on peut fréquemment voir que certain contrôle sont identiques sur toute les pages. Le texte affiché aussi ne change pas. Comment faire pour traduire ce contrôle une bonne fois pour toute sans avoir à rééditer chaque fois nos fichier *.resx* ?

2.2 La globalisation

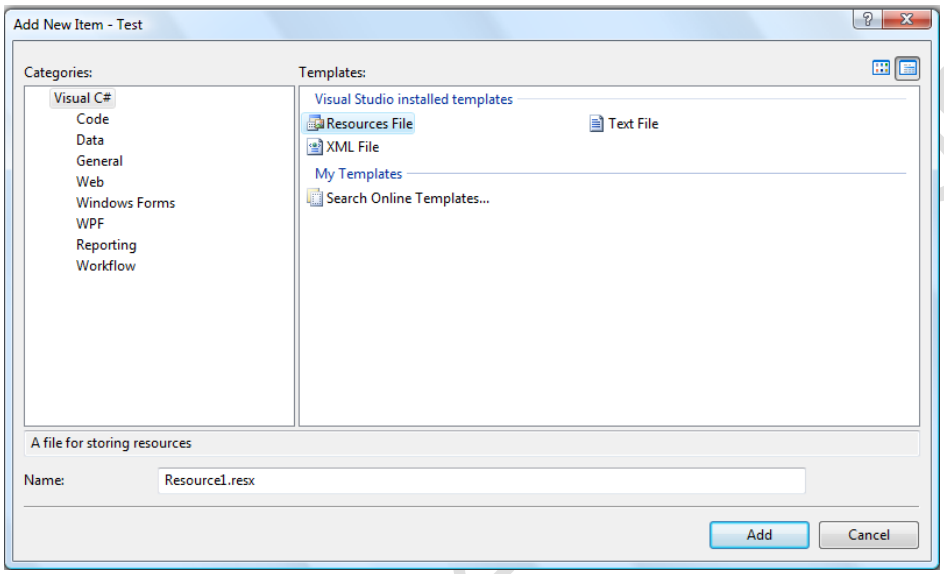
La globalisation nous permet de créer nos propres références et d'y faire appel dans n'importe quelle propriété attendant un string. Tout comme la localisation, et de la même façon, nous pourrons traduire ces références dans autant de langues que l'on souhaite.

2.2.1 Créer une ressource globale

Pour créer un fichier de ressource globale, il faut commencer par créer le dossier qui va contenir nos ressources.

Faites un clic droit sur votre solution puis Ajouter, Ajouter un dossier ASP.NET, App_GlobalResources.

Maintenant créons le fichier de ressource globale : faites clic droit sur votre dossier nouvellement créé, puis Ajouter, Ajourer un nouvel élément. Choisissez Fichier ressource :



Il nous faut maintenant le remplir. Voici comment se présente ce fichier.

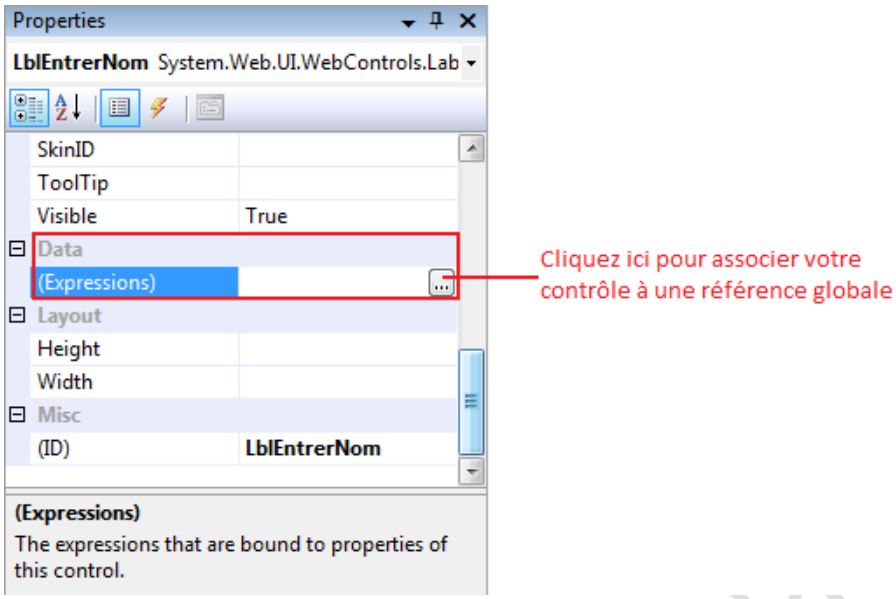
	Name	Value	Comment
▶	Bonjour	Bonjour à tous	
*			

Notre fichier de ressource globale est en fait un tableau contenant 3 colonnes :

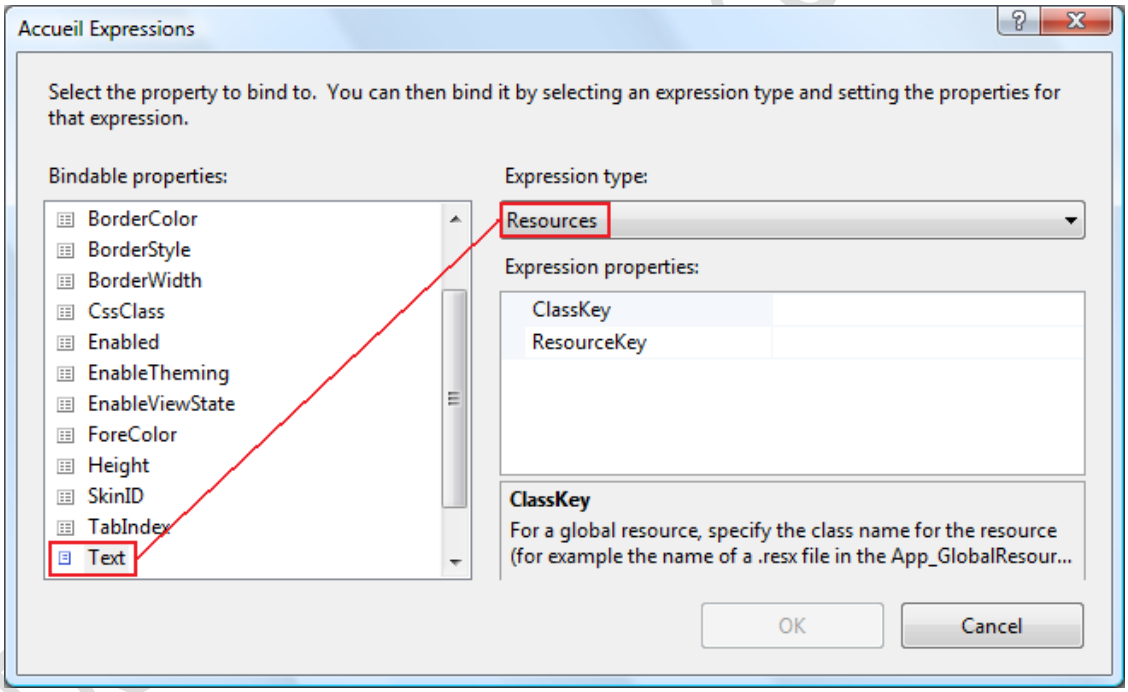
- Une colonne *Name* : qui nous permettra de définir le nom de notre référence ;
- Une colonne *Value* : qui nous permettra d'y associer une valeur de type string ;
- Une colonne *Comment* : qui nous permettra de commenter notre référence.

2.2.2 Lier une référence à la propriété d'un contrôle

Pour faire appel à ce fichier, créez une référence comme le montre l'aperçu ci-dessus, puis regardez les propriétés d'un de vos contrôles dont vous souhaitez voir afficher la valeur, un Label par exemple.



Il vous sera alors demandé sur quelle propriété vous désirez lier une expression (dans le cas présent nous choisirons la propriété *Text* pour y lier une référence) :



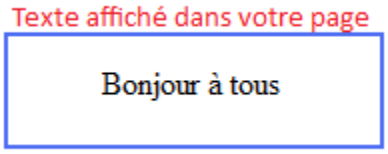
Dans la partie *ClassKey* mettez le nom de votre ressource global sans l'extension *.resx* (exemple : *Resource1*), puis dans la partie *ResourceKey* mettez le nom de la ressource que vous avez édité dans le fichier de ressource global (dans l'exemple vu plus haut on pourrait mettre *Bonjour*).

Une fois que vous avez validé cette action, du code ASPX est généré à notre place :

ASPX

```
<asp:Label ID="Accueil" runat="server"
   Text="<%$ Resources:Resource1, Bonjour %>"></asp:Label>
```

Désormais c'est la valeur correspondant à la ressource *Bonjour* du fichier *Resource1.resx* qui sera affichée.



2.2.3 Définir plusieurs langues pour fichier de ressource globale

Le mode opératoire pour créer d'autre langue est en tout point similaire avec les fichiers de ressources locales.

Il vous suffira de copier votre fichier de ressources globales et d'y ajouter le nom de la langue (selon la norme ISO 639-1) juste avant l'extension et traduire tout les champs *Value* dans la langue souhaité.

Votre nom de fichier doit être écrit de la façon suivante :

(*nom de votre fichier de ressources globales*).(*nom de la langue en ISO 639-1).resx*

Exemple :

Resource1.en.resx pour avoir un fichier de ressource qui sera chargé si la langue choisi par le navigateur est l'anglais.

Tout comme pour le fichier de ressources locales, si le navigateur réclame une langue qui n'est pas prise en charge, c'est le fichier de ressources globales par défaut qui sera chargé automatiquement (ici ce sera *Resource1.resx*).

2.2.4 Lier la propriété d'un contrôle à une ressource globale dynamiquement

Contrairement aux ressources locales, il est possible de lier la propriété d'un contrôle à une ressource globale via le *CodeBehind*.

Il suffit pour cela de suivre l'exemple suivant :

C#

```
Label lblLiasonResource = new Label();
lblLiasonResource.Text = Resources.Resource1.Bonjour;
form1.Controls.Add(lblLiasonResource);
```

VB.NET

```
Public Sub Page_Load(ByVal sender As Object, ByVal e As EventArgs) Handles Me.Load
    Dim lblLiasonResource As Label = New Label()
    lblLiasonResource.Text = Resources.Resource1.Bonjour
    form1.Controls.Add(lblLiasonResource)
End Sub
```

2.3 Le choix des cultures

Nous avons remarqué que lors de la création nous nous basions sur la norme ISO 639-1 pour nommer nos fichiers.

Lors de la génération de nos ressources locales vous avez surement du remarquer que la directive Page avait été modifié :

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Page1.aspx.cs"
Inherits="Test.Page1"
Culture="auto" UICulture="auto" %>
```

Des propriétés ont été rajoutées dont voici un explicatif :

- `Culture="auto"` : cette propriété définie la culture d'un point de vue monétaire, historique, etc., ainsi le format de date pourra changer ou le sigle monétaire. Pour modifier cet élément il faut prendre en compte la langue parlée selon la norme ISO 639-1 mais aussi les exigences régionales (fr-CAN pour français du Canada).
- `UICulture="auto"` : Défini quel fichier de ressource locale ou globale sera chargé au démarrage du navigateur, prenant en paramètre soit `auto`, soit une langue selon la norme ISO 639-1.

Les valeurs `auto` récupère la langue du navigateur par défaut, cela dit il peut arriver que la langue du navigateur ne soit pas celle de l'utilisateur (par exemple lorsqu'un touriste se rend dans un Web Café), il faut alors donner la possibilité de choisir sa langue de façon simple.

L'exemple qui suit nous montre comment effectuer cela :

ASPX

```
<form id="form1" runat="server">
<div>
    <%-- Voici une page contenant des contrôles dont les propriétés
         sont liées à des ressources globales ou locales --%>
    <asp:Label ID="Accueil" runat="server"
       Text="<%$ Resources:LocalizeText, Bonjour %>"
        meta:resourcekey="AccueilResource1"></asp:Label>
    <br />
    <br />
    <asp:DropDownList ID="lstCulture" runat="server"
       meta:resourcekey="lstCultureResource1">
    </asp:DropDownList>
    <asp:Button ID="btnSetCulture" runat="server" Text="Changer de langue"
       meta:resourcekey="btnSetCultureResource1" />
</div>
</form>
```

C#

```
protected void Page_Load(object sender, EventArgs e)
{
    // On crée notre liste de langue supporté
    if (lstCulture.Items.Count == 0)
    {
        ListItem itCultureFr = new ListItem("Français", "fr");
        ListItem itCultureEn = new ListItem("English", "en");
        lstCulture.Items.Add(itCultureFr);
        lstCulture.Items.Add(itCultureEn);
     }
}
// On surcharge la méthode InitializeCulture
protected override void InitializeCulture()
{
    if (Request.Form["lstCulture"] != null)
    {
        // On redéfini la valeur de UICulture
        UICulture = Request.Form["lstCulture"];
    }
    base.InitializeCulture();
}
```

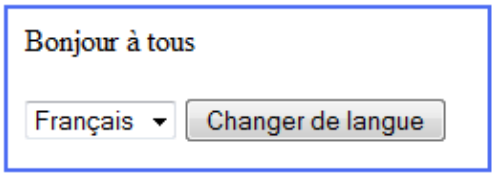
VB.NET

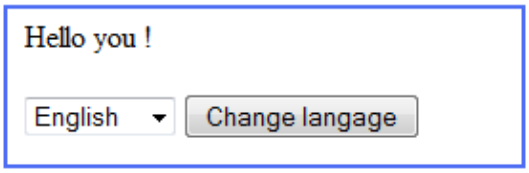
```
Partial Public Class _Default
    Inherits System.Web.UI.Page

    Public Sub Page_Load(ByVal sender As Object, ByVal e As EventArgs) Handles
Me.Load
        ' On crée notre liste de langue supporté
        If lstCulture.Items.Count = 0 Then
            Dim itCultureFr As ListItem = New ListItem("Français", "fr")
            Dim itCultureEn As ListItem = New ListItem("English", "en")
            lstCulture.Items.Add(itCultureFr)
            lstCulture.Items.Add(itCultureEn)
        End If
    End Sub

    ' On surcharge la méthode InitializeCulture
    Protected Overrides Sub InitializeCulture()
        If Request.Form("lstCulture") <> Nothing Then
            ' On redéfini la valeur de UICulture
            UICulture = Request.Form("lstCulture")
        End If
        MyBase.InitializeCulture()
    End Sub
End Class
```

Voici un aperçu de notre application (il faut bien entendu avoir généré nos ressources locales et globales auparavant).





3 Accessibilité

Cette notion est très importante lorsqu'on fait un site Web. En effet tout le monde n'utilise pas forcément le même navigateur pour afficher votre site. Des normes comme le W3C sont là pour permettre une cohérence entre les différents sites et ainsi permettre un affichage similaire pour tout les navigateurs existant : navigateur, navigateur portable (pour téléphone par exemple), terminal et autres encore comme la lecture de texte pour les non voyants… Pour augmenter l'accessibilité de votre application, il va donc falloir coller aux normes du Web. Cependant, il existe des astuces pour améliorer encore l'accessibilité. Elles sont plus spécifiques, c'est-à-dire que faire telle chose ciblera un certain type de visiteur sur votre site (par exemple on peut améliorer la lecture du site pour les logiciels de lecture de texte mais cela ne profitera pas aux autres visiteurs, ou encore d'autres naviguent sans souris et font donc tout au clavier).

3.1 L'accessibilité en ASP.NET

ASP.NET possède des contrôles qui ont été pensés pour l'accessibilité de votre site Web. Pour cela les contrôles serveur Web possèdent des propriétés telles *qu'AccessKey* qui permet de définir la touche

de raccourci pour ce contrôle. Il y a aussi *SkipLinkText* qui est une aide aux utilisateurs de logiciel de lecture d'écran. Cette propriété attend qu'on lui spécifie un string. Ce string correspondra à l'attribut Alt d'une balise image. La particularité de cette image est qu'elle contient un lien vers une autre partie de la page : juste après le contrôle. Aussi cette image possède une taille de 0, c'est-à-dire qu'elle ne sera pas visible par les utilisateurs normaux bien que l'image existe quand même dans le code. Ceci permet aux utilisateurs ciblés de passer le contrôle. Cette propriété existe sur le contrôle Menu, Wizard …

On pourrait aussi personnaliser l'accessibilité de notre application Web en utilisant des *TabIndex* qui définissent dans quel ordre les contrôles auront le focus lors d'un déplacement par la touche Tab.

3.2 Améliorer l'accessibilité

3.2.1 Visuelle

L'accessibilité visuelle est définie par un affichage clair et compréhensible de votre site Web quel que soit la situation et le navigateur ou périphérique utilisé. Ainsi si on met un tableau il faudra décrire ce qu'il représente ainsi que ses colonnes et lignes, si on met des images il faudra donner un titre à chacune d'elle etc.

Voici les points dont il faut tenir compte lorsque l'on fait un site Web :

Donner un titre aux images, ainsi qu'aux tableaux

Il peut arriver qu'une image ne soit pas chargée pour diverses raisons comme un lien cassé, un problème de connexion, l'utilisateurs peut aussi avoir désactivé le chargement des images ou encore son navigateur ne les prend pas en charge. C'est pour cela qu'il faut une alternative à l'image pour qu'en n'importe quelle circonstance l'utilisateur sache ce que représente l'image. Il existe aussi bien en HTML qu'en ASP.NET une propriété pour cela : respectivement *Alt* et *AlternateText*. Il est aussi possible que cette image ne soit pas importante et que vous vouliez donc que les lecteurs d'écran et autres ne le prennent pas en compte. La balise image possède en ASP.NET une propriété *GenerateEmptyAlternateText* qui permet de générer une description de l'image vide et donc d'être ignoré lors de la lecture du site.

En ce qui concerne les tableaux, en ASP.NET, il faut passer par une propriété sur la balise Table qui se nomme *Caption* pour définir son titre. En HTML il va falloir faire un balise *<caption></caption>* avec le titre. En plus d'expliciter ce que contient le tableau, c'est très utile pour un utilisateur de lecteur d'écran puisqu'il comprendra avant même d'entendre les données du tableau ce qu'il contient.

Faire attention aux couleurs utilisées

Vérifier que votre site reste facilement lisible. Des couleurs trop dans le même ton ou qui font mal aux yeux (par exemple parce qu'une couleur ressort trop en clair) vont gêner la lisibilité du site. De plus cela peut nuire aux personnes qui ont des problèmes que ce soit avec les faibles contrastes ou de simple vision (quand un site est dur à lire, les yeux forcent plus que la normale). Il n'est pas négligeable qu'un site agréable à lire sera forcément plus visité que s'il ne l'était pas : on ne perd

rien à faire un effort sur les couleurs.

Faire attention aux unités

Vous le savez, lors du développement du site Web vous allez devoir donner des tailles. Que ce soit pour spécifier la hauteur/largeur d'un bloc, sa position par rapport à un bord, la taille du texte et autres. Lorsque vous faites cela, vous le faites sur votre ordinateur. Il faut toujours penser que les autres n'ont pas le même matériel que vous. Ainsi peut être que votre site ne supportera pas un affichage sur une plus grande/ petite résolution. Peut être que votre site ne supportera pas que l'utilisateur augmente la taille du texte. Tout cela est autant de choses à prendre en compte.

Il va donc falloir vérifier que votre site peut être affiché sur différentes résolution et différentes taille de texte sans problème. C'est-à-dire que le site va rester lisible quoi qu'il arrive (évidemment sans aller à l'extrême : utiliser une police très grande déformera à coup sur votre site de façon illisible et vous n'y pourrez rien).

Surtout faire attention aux unités en pourcentage (et le positionnement) !

3.2.2 Formulaire

En ce qui concerne l'accessibilité des formulaires, on pourrait aussi parler de navigabilité. En effet le but est de permettre à une personne n'ayant pas de souris de remplir ce formulaire. Il n'y a qu'une solution : le clavier. Il faut donc rendre vos formulaires navigables par le clavier. Voyons quels sont les points à suivre pour l'accessibilité des formulaires :

Aide à la navigation

Il va falloir que vous définissiez la navigation dans vos formulaires. C'est-à-dire que vous permettiez à quelqu'un n'utilisant pas de souris de se déplacer dedans. Pour ce faire vous pouvez utiliser la propriété *TabIndex* des contrôles ASP.NET ou encore *AccesKey*. La première propriété permet de définir l'ordre de focus lors de l'utilisation de la touche Tab (évidement il faut le mettre dans l'ordre du formulaire) et la seconde définie une touche pour accéder au contrôle (raccourci avec Alt + Shift + touche définie). Il existe aussi la propriété *defaultfocus* sur les balises *form* qui permettent de choisir l'ID du contrôle qui aura le focus par défaut et donc dès que l'utilisateur va charger la page.

Une dernière chose utile pour aider à la navigation est la propriété *defaultbutton* sur les *form* et *panel*. Il prend comme valeur l'ID du bouton qui sera choisit par défaut si l'utilisateur appui sur la touche Entrer. Ainsi la personne n'a pas besoin d'aller en bas du formulaire pour valider.

Remarque : On peut mettre un raccourci sur un Label. Généralement pour que le raccourci soit visible, on utilise une lettre du texte définissant le contrôle en souligné (cette lettre représentant la lettre de raccourci).

Exemple :

Default.aspx

```
<form id="form1" runat="server" defaultfocus="TextBox3">
<div>
    <asp:TextBox runat="server" ></asp:TextBox>
    <asp:TextBox ID="TextBox3" runat="server"></asp:TextBox>
    <asp:TextBox runat="server" AccessKey="p">Alt + Shift + P</asp:TextBox>
</div>
</form>
```

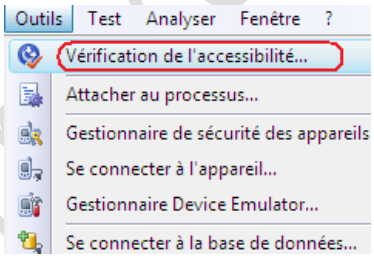
Faire des liens explicites et décrits

Les lecteurs d'écran permettent à leurs utilisateurs d'accéder à un lien grâce à la prononciation vocale du texte du lien en question. C'est pour cette raison que vous devez faire des textes explicites. Par exemple ne mettez pas "ici" ou encore "par là" et autres texte que nous avons tendance à mettre. Un bon lien serait plutôt : "Lien vers l'index du site", "site de programmation" ou encore "Vers galerie". Il est important de faire des liens qui puissent être utilisé uniquement en prononçant son texte.

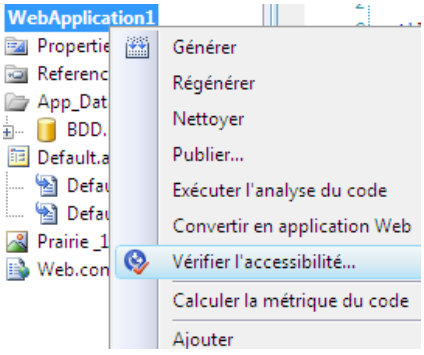
3.3 Tester l'accessibilité

Visual Studio vous donne la possibilité de vérifier l'accessibilité de votre site Web. Pour cela il y a deux manières de procéder mais qui amène aux mêmes résultats.

Soit vous faites Outils > Vérification de l'accessibilité.



Soit vous faites un clic droit sur le nom de votre projet et vous faites Vérifier l'accessibilité.



Les erreurs d'accessibilité seront affichées dans la liste d'erreurs.

4 Conclusion

Tous les utilisateurs n'utiliseront pas la même langue, aussi faudra-t-il penser à toucher le plus de monde possible en utilisant la globalisation pour implémenter d'autres langages.

Beaucoup ne se servent pas d'un système de pointage tel que la souris. Certains utilisent des claviers ou des systèmes vocaux. Avant de créer votre application, il est nécessaire de se demander à quel public est destiné votre site, à qui il s'adresse, pour définir vos propres règles lors de son développement. Et bien que le respect des normes d'accessibilités ne soit pas obligatoire, il rendra votre site visitable par un plus grand nombre de personne.

Les notions d'accessibilités sont nombreuses et ne peuvent être toutes abordées dans ce document, si vous désirez créer un site respectant un maximum de ces normes, n'hésitez pas à consulter d'autres sources dédiées à ce sujet.