WINDOWS PHONE

Amosse EDOUARD
MBDS Haiti, Août 2015

Environnement

- Windows 8 Pro 64 bit
- Windows Phone SDK + Visual Studio

Langages

Les Application Windows Phones peuvent être développées:

- C#, VB# et C++

- XAML

XAML

- Successeur de Silverlight
- Langage de balise basé sur du XML
- Utilisé pour définir les interfaces graphiques des applications

C#

- Nous préférons C# à VB dans ce cours
- Langage Orienté Objet fortement typé
- Très proche de la syntaxe Java
- Developpé et maintenu par Microsoft

C# et XAML

Les fichiers XAML sont liés à un fichier C# permettant d'implémenter et d'exploiter les objets décrits dans le XAML.

Un fichier XAML est transformé en code C# à la volée par l'IDE. Il y a donc en réalitée 2 fichiers .cs liés à un ficher XAML.

C# et XAML

Concrètement :

- 1 XAML → 1 classe C#

- 1 XAML → 2 fichiers physiques (*.cs)
  - 1 classe partial (partial)
  - 1 classe à proprement parlé

C#- Classe Partiel (Partial)

- Permet d'implémenter une classe dans deux fichiers distincts (mot clef "partial")
- 1 XAML est lié directement à une classe partielle, celle qui contiendra la logique applicative de l'application (→ édité par le developpeur)
- 1 fichier complémentaire généré et géré par l'outil de développement

C#- Classe Partiel (Partial)

- Considérons un fichier XAML App.xaml
- 1 classe partiel → App.cs
- 1 classe généré automatiquement par V.S → App.g.i.cs

Ca ne sert à rien de modifier le contenu de ce fichier, il sera regénéré par V.S

Architecture d'un projet

Dossier properties :

- WMAppManifest.xml
  - Déclaration des droits utilisés par l'application
  - Définit les propriées de l'application (ID, nom, …)
- AssemblyInfo.cs
  - informations de l'assembly

Architecture d’un projet

References:

Contient les librairies utilisées par l’application.

Assets :

Contient les données statiques de l’application (Images, vidéo, xml , ...)

Architecture d'un projet

Resources:

Il contient des dictionnaires de données (clef/valeur)

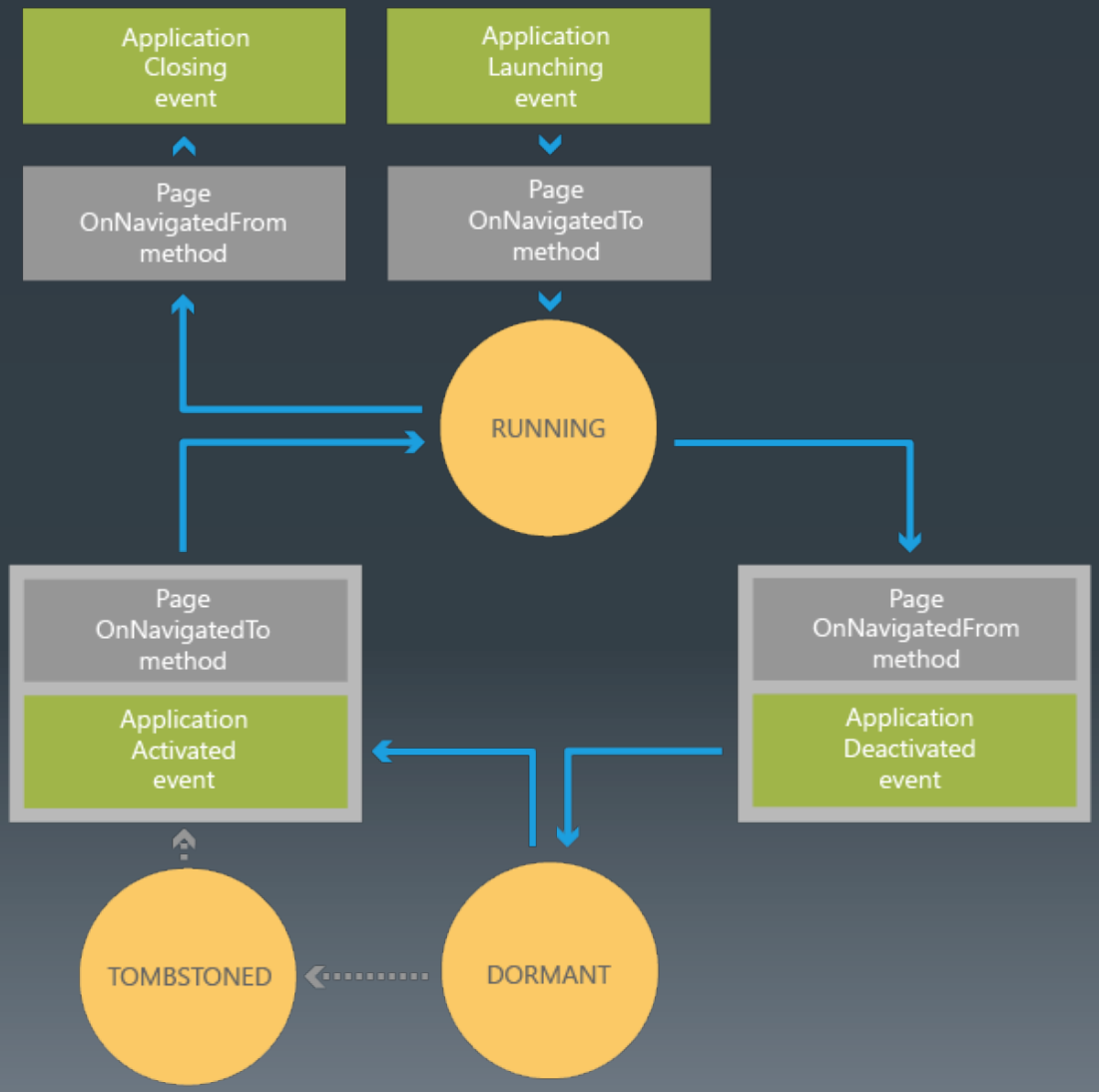
- Définit les styles de l'application (code de couleurs, style de texte, …)
- Gérer l'internationalisation

Architecture d'un projet

App.xaml:

- Lié à un fichier App.cs
- Gère les événements de l'application
- Gestion du cycle de vie de l'application

Cycle de vie d'une application



OnNavigationTo

- Méthode permettant de savoir quand un utilisateur arrive/revient sur une interface
- protected override void OnNavigatedTo(NavigationEventArgs e){}
- L'object NavigationEventArgs permet de déterminer le statut de la page

Etat Running

L’application est dans l’état running tant que l’ utilisateur n’a pas changé d’application.

L’état Running est perdu quand le téléphone se met en veille sauf si la méthode de détection de mise en veille (Idle Detection) a été surchargée.

OnNavigatedFrom

- Appelé lorsque l'utilisateur quitte l'écran actuel.
  - Sauvegarder l'état
  - NavigationEventArgs.NavigationMode

```
protected override void
OnNavigatedFrom(System.Windows.Navigation.NavigationEventArgs e){}
```

Etat dormant

- Quand l'utilisateur navigue en dehors de l'application.
  - Tous les threads sont bloqués
  - L'application reste en mémoire.
  - L'application peut être detruite (Tombstoned) si le téléphone a besoin de ressources

Etat Tombstoned

- L’ application a du être terminée:
  - Le téléphone a besoin de ressources
  - L'utilisateur l'a fermée
-
- L’app en re-devenant active aura accés à l’objet State qui, si il a été rempli permettra de restaurer l’état de l’application.

Evénement App Activated

- Appelé quand l'application reviens de l'état Dormant ou Tombstoned .
  - Vérifier IsApplicationInstancePreserved
    - true : l'app était Dormant et son état est intact
    - false : l'app était Tombstoned il faut restaurer (stateDictionary)

Evénement App Deactivated

- L'utilisateur quite l'application
  - Lance une autre application
  - Appuie sur le bouton start
  - Le téléphone passe en veille.
- Sauvegarder les variables de l'application , (Utiliser l'objet State)

Evénement App Closing

- L'utilisateur sort de l'application
  - Appuyer sur retour jusqu'à arriver sur l'écran d'accueil
  - Ferme l'application en l'enlevant de la pile
- Sauvegarder l'état de l'application si necessaire
  - En moins de 10 secondes

Interfaces Graphiques : Les pages

- Une application WP est consitituée d'une ou plusieurs pages
- Les pages sont déssinées dans des fichiers *.xaml
- Un visualisateur permet de voir le rendu en temps réél
- Les objets du fichier XAML sont manipulés dans le fichier .cs associé au .xaml

Interfaces Graphiques : Les pages

- La classe liée à un XAML est héritée de la super classe PhoneApplicationPage

- Cette classe dispose des événements liés à la page, qu'on peut surcharger au besoin

UI builder

- L'IDE met à votre disposition une boite a outil (onglet à gauche) facilitant l'ajout des composants standards boutons , champs texte, etc…
- UI builder vous permet de construire des interfaces en faisant du drag & drop
- En selectionnant un composant, une interface apparait à droite et vous permet d'ajouter des propriétés à l'objet

Architecture d'une page XAML

<phone:PhoneApplicationPage
x:Class="TPMBDS.Page1"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
xmlns:shell="clrnamespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc=
http://schemas.openxmlformats.org/markup-compatibility/2006
FontFamily="{StaticResource PhoneFontFamilyNormal}"
FontSize="{StaticResource PhoneFontSizeNormal}"
Foreground="{StaticResource PhoneForegroundBrush}"
SupportedOrientations="Portrait" Orientation="Portrait"
mc:Ignorable="d" shell:SystemTray.IsVisible="True">

Architecture d'une page XAML

Le Tag phone:PhoneApplicationPage est le tag racine d'un fichier XAML décrivant une page.

Outre les namespaces xmlns ,
on y trouve les attributs :
x:Class permettant le lien au fichier.cs de la page

FontFamily, Font Size, Foreground → définissant les paramètre de la police par défaut de la page

SupportedOrientations & Orientation → définissant l'orientation de l'interface

shell:SystemTray.IsVisible=bool (affiche ou
cache la barre de status en haut dutéléphone

Conteneur graphique GRID

- Les layouts en Windows phone fonctionne principalement avec des composants GRID qui définissent une grille ainsi que la largeur et la hauteur de chaque cellules.

- Les tailles peuvent être définies :
  - en pixel
  - avec le mode Auto (s'adapte au conteu)
  - avec le caractère * (s'étend au maximum)

Grid - Généralités

- Un Grid dispose ses éléments comme dans un tableau
- La propriété grid:RowDefinifions permet de définir les lignes
- La propriété grid:ColumnDefinitions permet de définir les colonnes

Grid- Généralités

```
<Grid.ColumnDefinitions>
<ColumnDefinition Width="*" />
<ColumnDefinition Width="*" />
<ColumnDefinition Width="*" />
</Grid.ColumnDefinitions>
```

```
<Grid.RowDefinitions>
<RowDefinition Height="*" />
<RowDefinition Height="*" />
<RowDefinition Height="*" />
</Grid.RowDefinitions>
```

Grid - Généralités

- Pour placer un élément dans un Grid, on doit spécifier la ligne ainsi que la colonne

```
<TextBlock Text="MBDS" FontSize="50"
Grid.Row="0" Grid.Column="0"/>
```

Les controles

- Plusieurs types de controles
- Tous les controles sont dérivés de la classe UIElement
- Permettent aux utilisateurs d'interagir avec l'application

Les controles

- Trois grandes catégories :
  - Les dérivés de ContentControl → Conteneurs d'autres objets
  - ItemsControl → Afficher une liste d'éléments
  - Control → Ne peuvent pas contenir d'autres objets ou afficher des listes (TextBlock, Button, …)

Les controles

- Le designer vous permet d'ajouter automatiquement les controles à votre interface

- Vous pouvez aussi utiliser Expression Blend pour enrichir vos interfaces

Les conteneurs

- Les conteneurs sont déstinés à contenir d'autres éléments
  - StackPanel
  - ScrollViewer
  - Canvas
  - …

StackPanel

- Conteneur destiné à positionner des éléments de façon linéaire
  - Horizontal
  - Vertical

ScrollViewer

- Cache automatiquement son contenu quand la taille de la vue est supérieure à celle de l'ecran
- Permet de scroller pour afficher les parties cachées

On peut placer un conteneur dans un autre…

Canvas

- Contrairement aux autres conteneurs, un canvas ne calcule pas la position des éléments qu'il contient
- Ils ne se redimensionnent pas en fonction de leur contenu
- Les éléments sont positionnés relativement par rapport au Canvas

Alignement

- Propriété permettant d'aligner les éléments dans leur conteneurs

- Types d'alignement
  - HorizontalAlignment
  - VerticalAlignment

- Valeurs
  - Stretch (étiré)
  - Left (à gauche)
  - Right (à droite)
  - Center (au centre)

Les marges (Margin)

- Permettent de définir les espacements entre les objets d'une interface
  - Par rapport à l'écran
  - Par rapport à d'autres objets

- Comme en CSS
  - Margin ="10" → 10 px (haut, gauche, droit, bas)
  - Margin ="10 20 30 40"
    - Haut :10 px, Gauche : 20 px
    - Bas : 30px et droit : 40px

Les resources

- Fichiers statiques embarqués dans l'application
  - Images
  - Sons
  - Vidéos
  - Styles

Les styles

- Les styles permettent de modifier l'apparence des controles
- Ils permettent de définir un ensemble de propriétés à appliquer sur un ou plusieurs éléments
- Sont des ressources
- Définis comme un dictionnaire

Les styles

- Les styles peuvent être définis
  - Au niveau de l'application
  - Au niveau d'une page

- Les styles sont associés aux controles via la propriété StaticResource définie pour tous les controles

Style Exemple

```
<phone:PhoneApplicationPage.Resources>
        <Style x:Key=”MonStyle" TargetType="TextBlock">
                <Setter Property="Foreground" Value=”Blue" />
                <Setter Property="FontSize" Value=”20" />
                <Setter Property="Margin" Value="0 10  0 10" />
                <Setter Property="HorizontalAlignment" Value="Center" />
        </Style>
</phone:PhoneApplicationPage.Resources>

<TextBlock Text="{StaticResource HelloWorld}"
Style="{StaticResource MonStyle}"/>
```

C#, XAML et les instances d'objects

Prenons l'exemple d'un Textblock déclaré comme suit :

```
<TextBlock x:Name="tb_text" TextWrapping="Wrap" />
```

si l'on veut modifier son texte dans une méthode de notre classe Page, il faudra faire par exemple :

```
private void addLine(String line)
{
tb_text.Text = tb_text.Text + line;
}
```

En effet dans le code behind caché l'IDE l'a instancié pour nous :

```
this.tb_text = ((System.Windows.Controls.TextBlock)
(this.FindName
("tb_text")));
```

C#, XAML et Evénements

Prenons l'exemple d'un Bouton :

<Button x:Name="btn_send" Grid.Column="2" Content="SEND" Tap="

btn_send_Tap"/>

L'attribut Tap sert à définir une méthode à appeler lors d'un Tap sur le bouton. Lors de la saisie de la valeur de l'attribut l'IDE propose la création d'un événement dans le code .

Code généré :

private void btn_send_Tap(object sender, System.Windows.Input.GestureEventArgs e){ …}

Navigation

- Passer d'une fenêtre à une autre
  - Automatiquement
  - En répondant à une action utilisateur
- Service assuré par NavigationService
  - Acceptant en paramètre l'URI relative de la nouvelle page

Navigation

Aller vers une nouvelle page

NavigationService.Navigate(new Uri(”/NouvellePage.xaml", UriKind.Relative));

Revenir sur la page précédente

NavigationService.GoBack();

Navigation – Paramètres

- Envoyer des paramètres

```
NavigationService.Navigate(new Uri("/
NouvellePage.xaml?nom="+txtNom.Text
+"&prenom="txtPrenom.Text,
UriKind.Relative));
```

- Récupérer les paramètres
  - Utilisation de la méthode (événement) OnNavigatedTo

Navigation - OnNavigatedTo

```
protected override void
OnNavigatedTo(System.Windows.Navigation.NavigationEventArgs e)
    {
            base.OnNavigatedTo(e);
            string nom, prenom;
            NavigationContext.QueryString.TryGetValue("nom", out nom);
            NavigationContext.QueryString.TryGetValue("prenom", out
prenom);
    }
```

Pour aller plus loin

- Gestion de données
  - DataBinding et Converters
  - MVVM
  - Requetes HTTP
  - Gestion de données locales
- Capteurs (GPS, Map)
- Gestion de l'orientation du téléphone
- Taches de fonds
- Réseaux Sociaux
- Publication de l'application sur le store

TP

- Dans le TP d'aujourd'hui nous allons concevoir les interfaces de l'application cliente Compte Bancaires.
- Gérer l'enchainement des écrans
- La semaine prochaine, nous allons synchroniser les données en utilisant le service Web