



Développement d'application cross- platform avec Xamarin

Julien Mialon : mialon.julien@gmail.com

Valentin Jubert : valentin.jubert@outlook.fr

Au sommaire...

Séance 1

- Écosystème mobile
- Natif ou Cross-Platform?
- Xamarin et ses outils
- C# & .NET
- Xamarin.Forms
- DevOps (CI / CD / CT) + AppCenter

Séance 2

- Xamarin.Forms avancé
- OAuth2 password flow
- Push notifications
- Cache
- Architectures
- Publication sur les stores

Au sommaire...

QCM le 05/03 (environ 30 minutes)

TD/mini-projet à rendre pour le 03/04



Xamarin.Forms

MVVM



Bindings

```
public class HomeViewModel : ViewModelBase
{
    private string _title;
    private ObservableCollection<Todo> _todos;

    public string Title
    {
        get => _title;
        set => SetProperty(ref _title, value);
    }

    public ObservableCollection<Todo> Todos
    {
        get => _todos;
        set => SetProperty(ref _todos, value);
    }
}
```

Bindings

- Une fois le contexte définit, toutes les **PROPRIÉTÉS PUBLIC** du contexte deviennent disponibles pour le binding

```
<Label Text="{Binding Title}" />
```



```
<Label Text="{Binding _title}" />
```



MVVM : Converters

- Comment faire pour adapter les valeurs provenant du ViewModel à ce qu'on veut faire dans la View ?
- Exemples:
 - Enum à convertir en texte ou en image
 - Cacher un élément si une liste est vide

MVVM : Converters

- 1) On peut changer la propriété dans le ViewModel ?
 - Oui mais ce n'est pas la meilleure des solutions, notre ViewModel ne doit pas s'adapter à la View, c'est l'inverse qui doit se produire
- 2) Utiliser les converters afin de transformer les valeurs du ViewModel
 - Oui !

MVVM : Converters

```
public class SeasonToImageConverter : IValueConverter
{
    public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
    {
        Season season = (Season) value;
        switch (season)
        {
            case Season.Winter: return "winter.jpg";
            case Season.Spring: return "spring.jpg";
            case Season.Summer: return "summer.jpg";
            case Season.Autumn: return "autumn.jpg";
            default: throw new ArgumentOutOfRangeException();
        }
    }

    public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
    {
        throw new NotImplementedException();
    }
}
```

MVVM : Converters

```
<ContentPage.Resources>
    <local:SeasonToImageConverter x:Key="SeasonToImageConverter" />
</ContentPage.Resources>

<Image Source="{Binding Season, Converter={StaticResource SeasonToImageConverter}}"
        HeightRequest="100"
        Aspect="AspectFit" />
```

MVVM : Converters

```
<Label Text="{Binding Price, StringFormat='Prix: {0:0.00}€'}" />
```



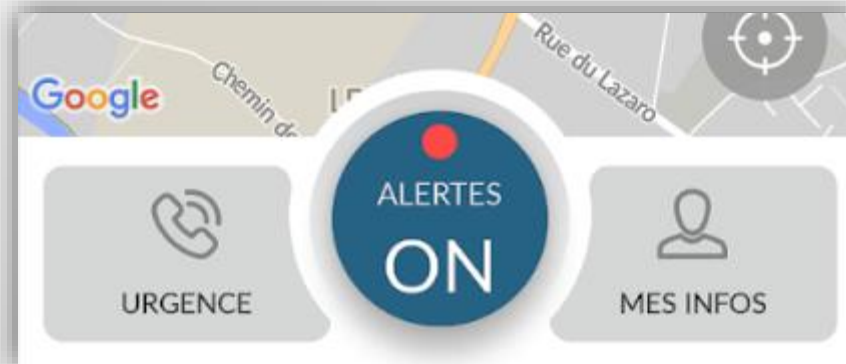
Aa Bb Cc Dd

$2 + 2 = \dots$

Questions ?

Custom Renderer

- Comment faire pour réaliser des vues très complexe ?



Custom Renderer

- Les custom renderers permettent de définir une abstraction « partagée » pour des contrôles avec des rendus natif
- C'est de cette manière que sont réalisés tous les control du framework

Custom Renderer

```
public class MyEntry : Entry
{
}
```

```
<ContentPage ...
  xmlns:local="clr-namespace:CustomRenderer;assembly=CustomRenderer"
  ...>
  ...
  <local:MyEntry Text="In Shared Code" />
  ...
</ContentPage>
```


Custom Renderer

```
[assembly: ExportRenderer(typeof(MyEntry), typeof(MyEntryRenderer))]  
namespace CustomRenderer.Android  
{  
    class MyEntryRenderer : EntryRenderer  
    {  
        public MyEntryRenderer(Context context) : base(context)  
        {  
        }  
  
        protected override void OnElementChanged(ElementChangedEventArgs<Entry> e)  
        {  
            base.OnElementChanged(e);  
  
            if (Control != null)  
            {  
                Control.SetBackgroundColor(global::Android.Graphics.Color.LightGreen);  
            }  
        }  
    }  
}
```

Custom Renderer

- Avantages :
 - Aucune limite par rapport à une application native
- Inconvénients :
 - Vous devez écrire le code pour Android et pour iOS

Skia Sharp

- Dessin 2D
- C'est ce que Flutter utilise pour faire tout ses contrôles
- C'est gérer par google
- Votre limite devient votre imagination

Skia Sharp



Skia Sharp

Affinez vos garanties



Tiers

Tiers+

Tous risques



Aa Bb Cc Dd

$2 + 2 = \dots$

Questions ?



DÉMO

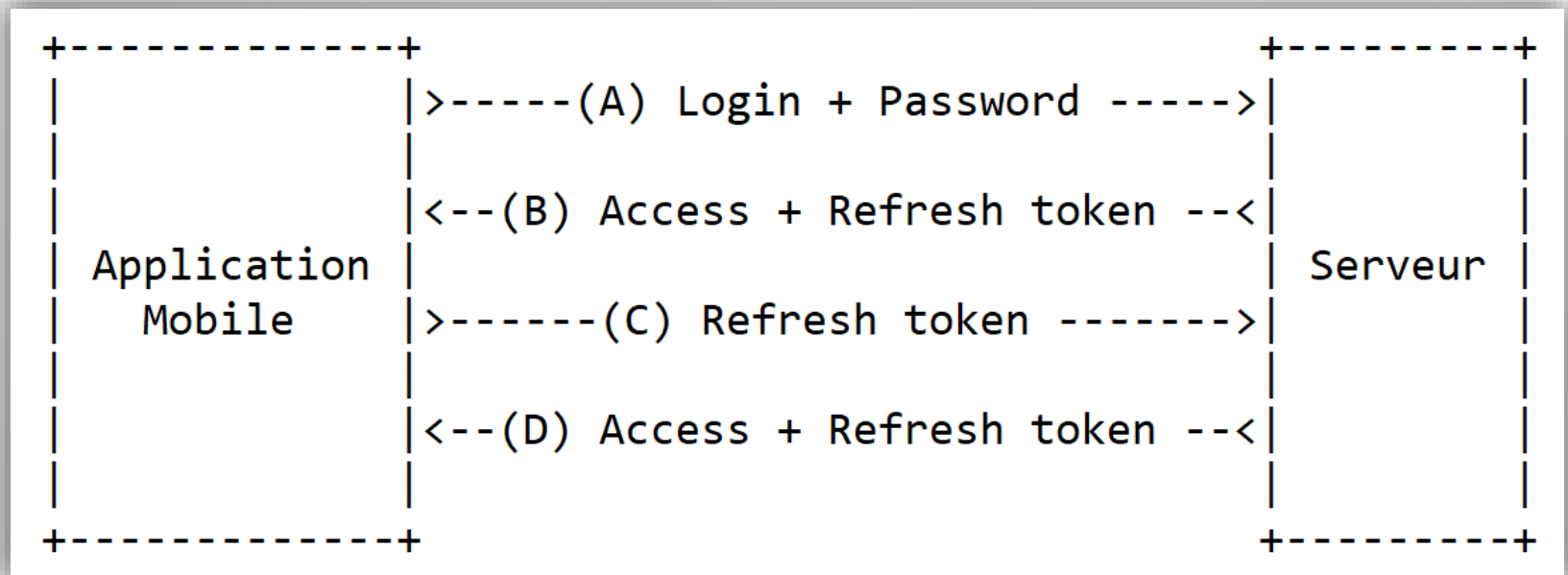
— live



OAuth2

- L'authentification dans une application mobile nécessite un certain niveau de sécurité
- Votre téléphone peut être volé, copié, ...
- Le but est que dans ce cas, vos login/mot de passe ne soit pas compromis

OAuth2



OAuth2

- Le login + le mot de passe n'est nécessaire qu'à la première étape => pas nécessaire de le stocker
- L'access token a une durée de vie limité (en général < 1 jour)
- Le refresh token permet de récupérer un nouvel access token et peut être invalidé afin de ne plus permettre la connexion

OAuth2

- Inconvénients :
 - Il faut envoyer l'access token dans toutes les requêtes HTTP
 - Il faut refresh l'access token quand il est expiré ou presque expiré

POST

`{{api_root}}/oauth2/login`

Params

Authorization

Headers (1)

Body ●

Pre-request Script

Tests ●

● none

● form-data

● x-www-form-urlencoded

● raw

● binary

JSON (application/json) ▼

```
1 {  
2   "email": "{{oauth2_login}}",  
3   "password": "{{oauth2_password}}",  
4   "grant_type": "Password",  
5   "client_id": "{{oauth2_client_id}}",  
6   "client_secret": "{{oauth2_client_secret}}"  
7 }
```

Body

Cookies

Headers (7)

Test Results

Pretty

Raw

Preview

JSON ▼



```
1 {  
2   "data": {  
3     "access_token": "5179d0a4deb648fa936ba4781c20f59a",  
4     "refresh_token": "9753a98d411e41a7b968c3bd867f685a",  
5     "expires_in": 86399,  
6     "token_type": "Bearer"  
7   },  
8   "is_success": true,  
9   "error_code": null,  
10  "error_message": null  
11 }
```

▶ front /me

GET



{{front_root}}/me

Params

Authorization

Headers (1)

Body

Pre-request Script

Tests ●

KEY

VALUE



Authorization

Bearer 5179d0a4deb648fa936ba4781c20f59a

Key

Value

POST

`{{api_root}}/oauth2/refresh`

Params

Authorization

Headers (1)

Body ●

Pre-request Script

Tests ●

☐ none☐ form-data☐ x-www-form-urlencoded☒ raw☐ binary

JSON (application/json) ▼

```
1 {  
2   "refresh_token": "{{refresh_token}}",  
3   "client_id": "{{oauth2_client_id}}",  
4   "client_secret": "{{oauth2_client_secret}}"  
5 }
```

Body

Cookies

Headers (7)

Test Results

Pretty

Raw

Preview

JSON ▼



```
1 {  
2   "data": {  
3     "access_token": "182e35f6dbbe47d689e6a101d5408171",  
4     "refresh_token": "abf7f924bbfe4e71825ac02c722fb7c4",  
5     "expires_in": 86399,  
6     "token_type": "Bearer"  
7   },  
8   "is_success": true,  
9   "error_code": null,  
10  "error_message": null  
11 }
```



Aa Bb Cc Dd

$2 + 2 = \dots$

Questions ?



Push notifications

Push notifications



Twilio Blog

18:32

How to add push notifications to your Andr..

360 Hello 360 • now ^

April Bowman

Checkout the sunset in Berlin



data updated

7:24 PM

allow sync immediatly

APPROVE

REJECT

Push notifications

CRÉATION

Création
manuelle

Génération
automatique

ENVOI

Serveur
d'envoi

TRANSMISSION

serveurs de push

iOS
APNS

Android
GCM

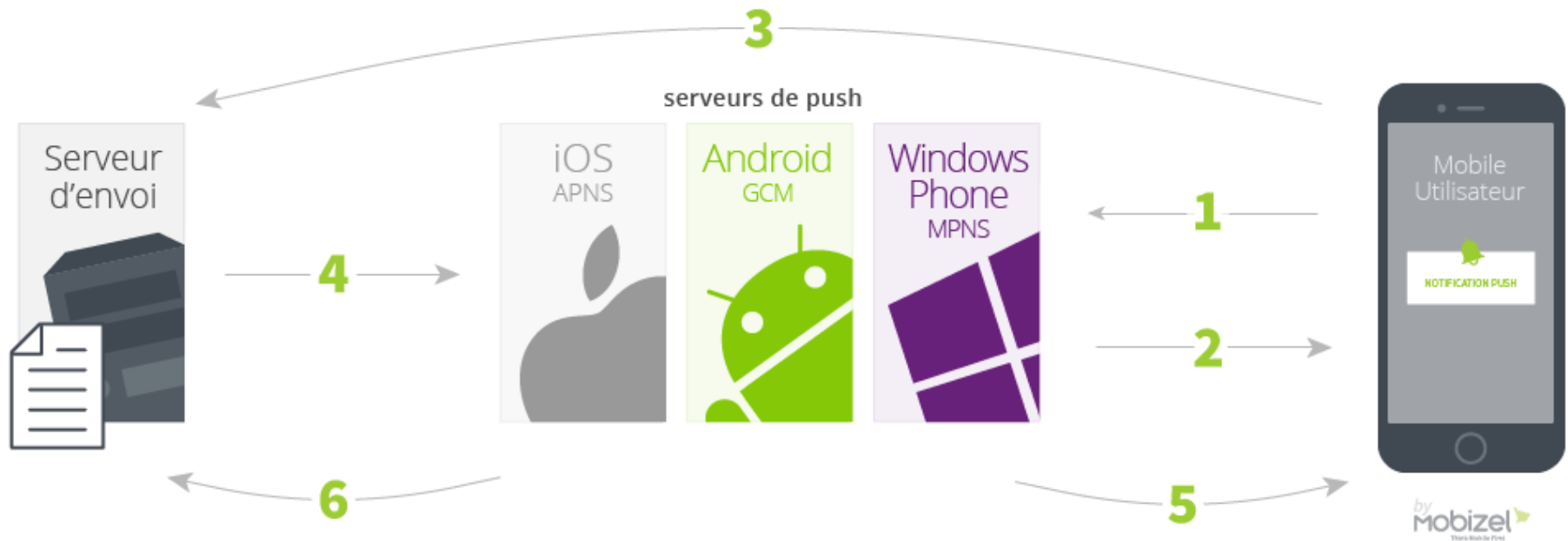
Windows
Phone
MPNS

RÉCEPTION

Mobile
Utilisateur

NOTIFICATION PUSH

Push notifications





Cache

Cache

- Pourquoi faire du cache ?
 - Mode offline
 - Interruption du réseau
 - ...



- <https://github.com/jamesmontemagno/monkey-cache>

Cache

```
async Task<IEnumerable<Monkey>> GetMonkeysAsync()
{
    var url = "http://montemagno.com/monkeys.json";

    //Dev handle online/offline scenario
    if(!CrossConnectivity.Current.IsConnected)
    {
        return Barrel.Current.Get<IEnumerable<Monkey>>(key: url);
    }

    //Dev handles checking if cache is expired
    if(!Barrel.Current.IsExpired(key: url))
    {
        return Barrel.Current.Get<IEnumerable<Monkey>>(key: url);
    }

    var client = new HttpClient();
    var json = await client.GetStringAsync(url);
    var monkeys = JsonConvert.DeserializeObject<IEnumerable<Monkey>>(json);

    //Saves the cache and pass it a timespan for expiration
    Barrel.Current.Add(key: url, data: monkeys, expireIn: TimeSpan.FromDays(1));
}
```



Architectures

Architectures

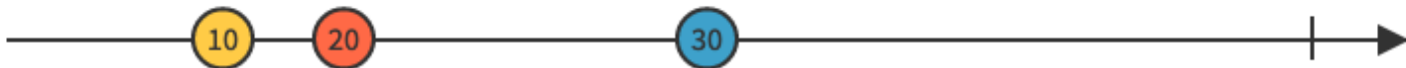
- MVVM : Model-View-ViewModel
- MVC : Model-View-Controller
- Rx : Reactive Extensions
- Components

Rx

- Basé sur le fait de réagir à des événements
- Un flux d'événement se produit et on s'abonne dessus



```
map(x => 10 * x)
```



Rx

```
static void Main()
{
    var oneNumberPerSecond = Observable.Interval(TimeSpan.FromSeconds(1));

    var lowNums = from n in oneNumberPerSecond
                  where n < 5
                  select n;

    Console.WriteLine("Numbers < 5:");

    lowNums.Subscribe(lowNum =>
    {
        Console.WriteLine(lowNum);
    });

    Console.ReadKey();
}
```

Rx

- Pour en savoir plus : <http://reactivex.io/>
- Rx.Net : <https://github.com/dotnet/reactive>
- RxUI : <https://github.com/reactiveui/ReactiveUI>

Components

- Architecture provenant du web (React par exemple)
- Utilisé surtout sur React.Native, Flutter
- Possible sur Android & iOS

Components

- Nécessaire de découper son application en composant
- Exemple un composant bouton, un composant map, un composant détail d'un lieu...
- Chaque composant peut utiliser des composant enfant
- Chaque composant doit être auto-géré

Components

- Avec Xamarin :
 - Un ComponentViewModel => partagé iOS/Android
 - Un ComponentView
- ComponentViewModel => génère un état pour la vue (du json simple par exemple)
- ComponentView reçoit l'état et met à jour l'affichage en fonction

Components / Flutter

```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Welcome to Flutter',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Welcome to Flutter'),
        ),
        body: Center(
          child: Text('Hello World'),
        ),
      ),
    );
  }
}
```



Publication sur les stores

Publication Android

- Google Play (ou autre store privé : Amazon, Huawei, ...)
- Votre application doit être signée avec un keystore
- Vous devez ouvrir un compte google play (25\$ à vie)
- Uploader votre application, remplir sa fiche et la publier
- Temps de publication :
 - variable si review manuelle
 - Moins de 4h autrement

Publication iOS

- AppStore
- Vous devez ouvrir un compte développeur Apple (99\$/an)
- Vous devez signer votre application avec un certificat Apple
- Uploadé votre application et remplir sa fiche
- Validation manuelle par les équipes d'Apple
- Temps de publication : environ 24h



Aa Bb Cc Dd

$2 + 2 = \dots$

Questions ?