

Table des matières

1	Introduction	1
2	Application	1
3	Maquette	2
4	Installation de l'environnement	2
5	Création du projet	3
6	Création d'une première Activité	4
6.1	Les fenêtres de l'environnement	5
6.2	Interface utilisateur	5
6.3	Layouts	5
6.3.1	LinearLayout	5
6.3.2	RelativeLayout	6
6.3.3	TableLayout	6
7	Fichier de ressources	6
8	Compilation	6
8.1	Emulateurs Android	7
8.2	Chargement sur un appareil physique	7
9	Lien entre deux Activités	7
10	Méthodes événementielles	7
11	Layout dynamique	7
12	Persistance de données	7
13	Utilisation d'un sensor	7

1 Introduction

Ce document a pour objectif de mettre en évidence les différentes étapes dans la réalisation d'une application Android.

Il s'agit d'un document pour des informaticiens, c'est-à-dire que nous partons du principe que la personne qui le lit, a des notions de développement et tout particulièrement en C#.

Même si MS annonce la fin de cette plateforme avec l'arrivée de .Net Maui, nous utiliserons l'environnement Visual Studio C# avec Xamarin.

2 Application

Pour mettre en évidence les différentes étapes à réaliser dans un projet, nous réaliserons une application spécifique : **Un gestionnaire de tâches**.

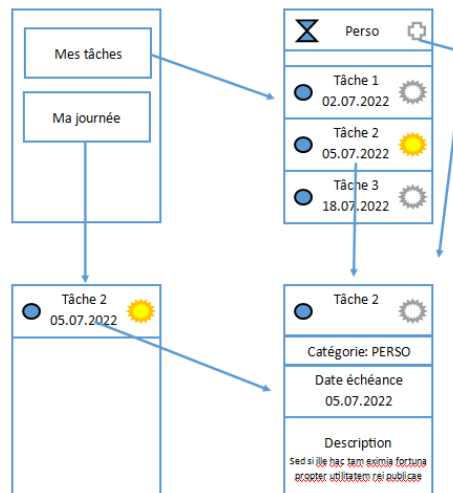
Cette application doit avoir au minimum les fonctionnalités suivantes :

- Pouvoir créer et visualiser des tâches.
- Une tâche est caractérisée au minimum par son titre, sa description et sa date d'échéance.
- Il doit être possible de mettre une tâche dans une catégorie.
- Il doit être possible de sélectionner des tâches pour en faire une liste « Ma journée », c'est-à-dire choisir les tâches à réaliser aujourd'hui.

De plus, cette application doit permettre d'aborder les thèmes suivants :

- Création d'une première Activité.
- Utilisation d'au moins un fichier de ressources, comme « Strings ».
- Lien entre deux Activités.
- Méthodes événementielles.
- Layout dynamique (c'est-à-dire création dans le code C#
- Persistance de données.
- Utilisation d'un sensor.

3 Maquette



Quatre interfaces sont prévues. La première permet de choisir si l'on veut aller sur l'écran de toutes les tâches ou sur l'écran des tâches de la journée.

L'écran de toutes les tâches aura un filtre en haut permettant de choisir si l'on veut afficher toutes les tâches ou uniquement celles d'une catégorie. En dessous, une liste scrollable permettra de parcourir les tâches. Sera affiché le titre, la date d'échéance (s'il y en a une) une checkbox pour définir que la tâche est terminée et une checkbox pour définir si la tâche doit se trouver dans la liste des tâches du jour. Il contiendra aussi un icône « + » qui permettra de créer une nouvelle tâche.

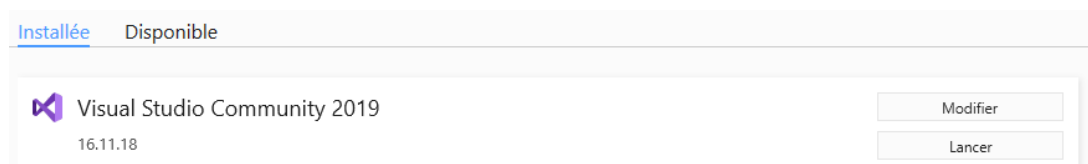
L'écran des tâches du jour aura une liste scrollable qui permettra de parcourir les tâches. Sera affiché le titre, la date d'échéance (s'il y en a une) une checkbox pour définir que la tâche est terminée et une checkbox pour définir si la tâche doit se trouver dans la liste des tâches du jour, qui permettra de la désélectionner si désiré.

C'est ici que nous utiliserons le sensor. L'idée étant que si on secoue l'appareil, toutes les tâches disparaîtront de la liste du jour (désélection de la checkbox de toutes les tâches). Ce qui permettra de recommencer une nouvelle journée à zéro.

Dans les deux interfaces, en cliquant sur le centre de la tâche l'utilisateur sera dirigé sur l'écran des détails d'une tâche. Ici il sera possible de voir les informations complètes de la tâche mais aussi possible de la modifier.

4 Installation de l'environnement

Une fois Visual Studio installé, il faut y intégrer l'environnement Xamarin. Pour cela, il faut lancer Visual Studio Installer et Modifier l'installation de Visual Studio.



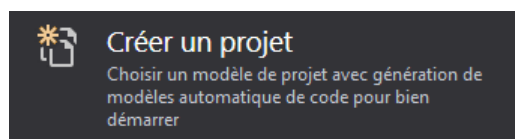
Le package « Développement mobile en .Net » doit être sélectionné.



Le développement pour Android reposant sur le SDK Java et sur le SDK Android, il est possible de le configurer dans Visual Studio. Menu *Outils* → *Options* → *Xamarin* → *Paramètres Android*.

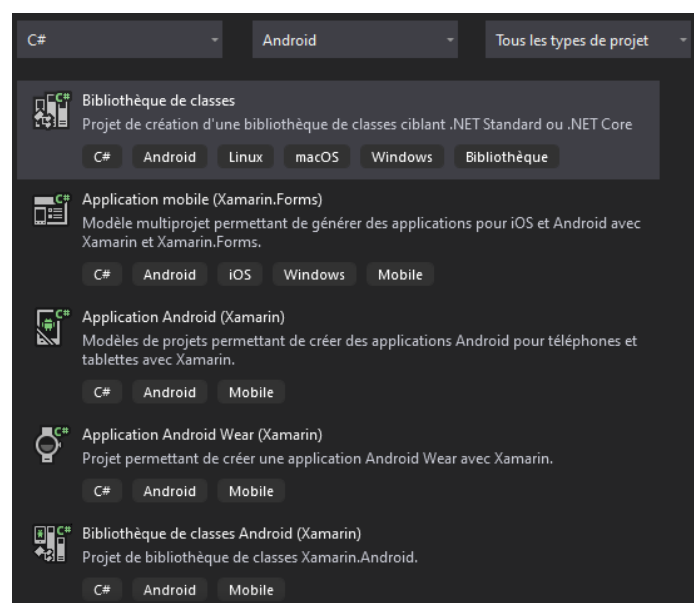
5 Création du projet

Dans la fenêtre de démarrage de Visual Studio, choisir « Créer un projet ».



Sélectionner les bons filtres : C#, Android et Tous les types de projet.

Choisir un projet de type « Application Android (Xamarin) ».

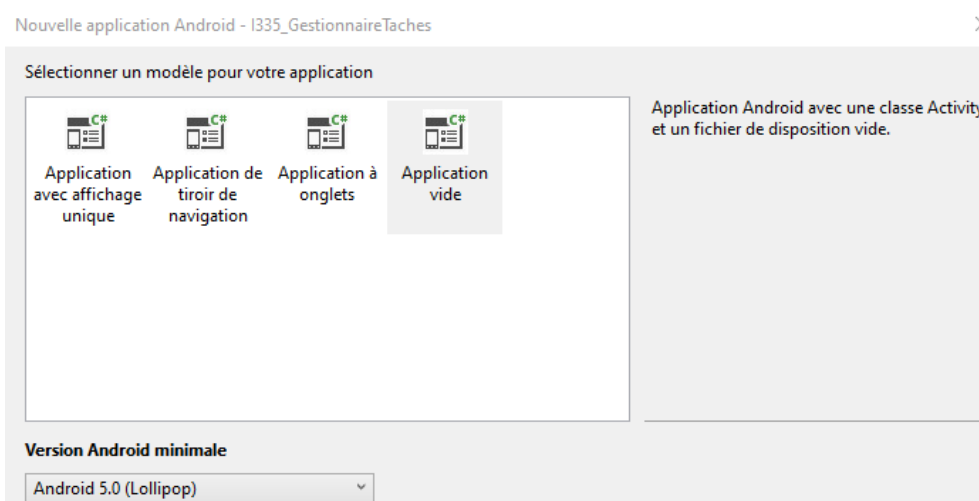


Dans la configuration du projet, donner un titre pertinent et choisir l'emplacement où sera stocké le projet, créer le projet.



Dans la sélection du modèle à utiliser, dans notre cas nous choisirons « Application vide » pour répondre à la contrainte de l'explication de la création d'une première Activité.

Choisir la version d'Android minimale. Pour cette partie, plus nous choisissons une récente, plus nous aurons des fonctionnalités récentes à disposition. La solution semble être de choisir la version la plus récente. Cependant cela aura comme conséquence que notre application ne pourra tourner que sur une portion minimale des supports Android existants sur le marché. En effet peu de personnes ont un support Android de dernière génération.



6 Création d'une première Activité

En créant le projet, la première activité (vide) est créée. Celle-ci est composée de deux fichiers principaux :

- MainActivity.cs
- Activity_main.xml

MainActivity.cs contient le code C# et les méthodes événementielles. Il s'agit de la partie C# de la première activité. C'est aussi l'activité principale puisque c'est celle qui est lancée à l'ouverture de l'application.

Activity_main.xml contient les balises xml qui permettent de créer l'interface utilisateur qui sera affichée.

6.1 Les fenêtres de l'environnement

L'environnement s'ouvre avec différentes fenêtres à disposition.

La partie centrale contient la fenêtre de développement. C'est d'ailleurs « MainActivity.cs » qui est affiché.

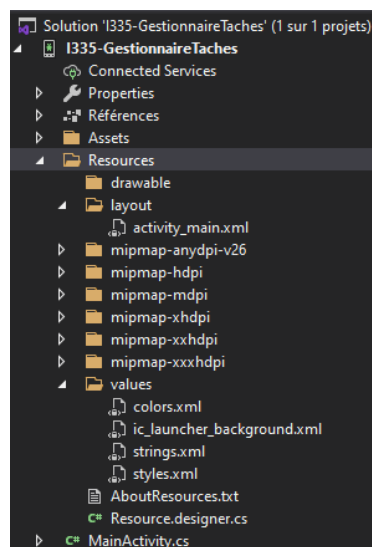
A gauche, on peut trouver aussi la « Boîte à outils ». Pour le moment elle est vide, mais elle contiendra le « Widgets » lorsque nous serons sur la partie interface utilisateur.

A droite, nous trouvons « l'explorateur de solutions » qui nous permettra de sélectionner les différents fichiers comme par exemple, le code C#, les balises xml ou les fichiers de ressources.

On peut trouver enfin la fenêtre des « propriétés ». Elles aussi est vide pour le moment mais comme pour la fenêtre de la « Boîte à outils », les propriétés des Widgets s'afficheront lorsque nous serons sur la partie interface utilisateur.

6.2 Interface utilisateur

Pour mettre en place l'interface utilisateur, il faut ouvrir le fichier xml correspondant. Pour cette première Activité, il s'agit du fichier « Activity_main.xml ». Pour ceci. Il faut aller dans « l'explorateur de solutions » est aller dans le répertoire « Resources » et « Layout ».



Lorsque le fichier est sélectionné, la fenêtre correspondante est partagée en deux. A gauche se trouve une fenêtre de création graphique où il est possible de mettre des Widgets en Drag/Drop. A droite, nous trouvons le correspondant en balises xml.

6.3 Layouts

Si l'on regarde du côté des balises, on peut voir que le premier Widget est un Relative Layout.

Les layouts sont des boîtes qui permettent de mettre en communs plusieurs Widgets et les traiter ensemble.

Il existe différents types de Layout, chacun ayant ses spécificités.

6.3.1 LinearLayout

Ce type de mise en page permet d'organiser une liste d'éléments de façon horizontale ou verticale. Chaque nouvel élément venant automatiquement se placer en dessous ou à droite de l'élément précédent.



6.3.2 RelativeLayout

Ce type de mise en page permet d'organiser les éléments les uns en fonction des autres, donc de façon relative. Il est ainsi possible de spécifier qu'un champ de texte soit au-dessus d'un champ de saisie ou qu'une image soit centrée dans son composant parent.

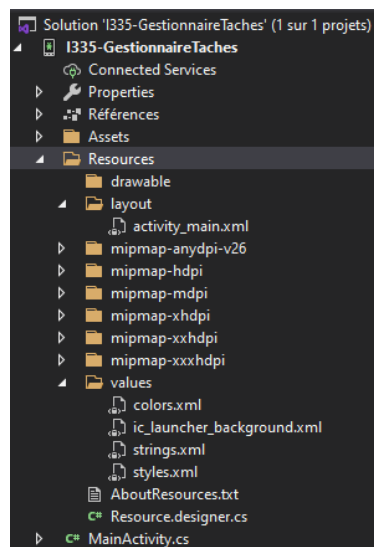
6.3.3 TableLayout

Lorsque l'on souhaite afficher des éléments sous la forme d'un tableau, avec lignes et colonnes, il faut utiliser la classe TableLayout. Elle se compose d'une liste de lignes (TableRow) pouvant avoir une ou plusieurs cellules associées.


7 Fichier de ressources

Pour afficher les différents fichiers de références, il faut aller dans « l'explorateur de solutions » est aller dans le répertoire « Resources ».

Pour trouver le fichier contenant les références de chaînes de caractères, il faut aller ensuite dans le répertoire « values ». Ici se trouve le fichier « strings.xml ».



8 Compilation

Pour lancer la compilation, il faut appuyer sur . Cependant auparavant il est nécessaire de définir où l'on veut que le programme s'exécute. En effet, le nombre d'appareils Android est immense. Chacun avec ses spécificités propres, ses tailles et définitions d'écran. Il est donc plus qu'intéressant de faire tourner son application sur plusieurs supports.

A ce niveau deux choix s'offrent à nous. Soit de charger l'application sur un ou plusieurs vrais appareils, soit de le charger sur un ou plusieurs émulateurs.

8.1 Emulateurs Android

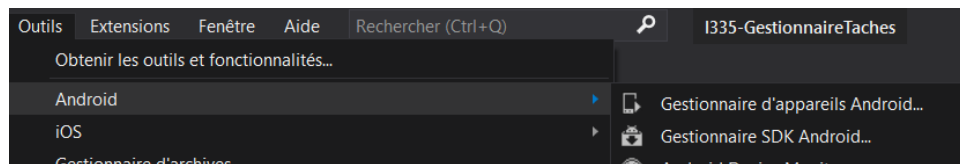
Il existe un très grand nombre de périphériques exécutant Android, que ce soit des téléphones, des tablettes, des « phablettes », etc. Donc un maximum de tests serait le mieux pour valider l'application.

La fragmentation d'Android est telle qu'il est peu concevable d'acheter un modèle de chaque version d'Android (avec, pour chaque version, un modèle/constructeur différent). Le mieux est d'utiliser des émulateurs qui permettront de simuler les différents environnements.

Depuis la version 25.2.3 des Android SDK Tools, l'application Android SDK Manager de Google est déprécié à la faveur de la version Xamarin Android SDK Manager. C'est celle qui est installée dans notre environnement.

Menu outils → Android → gestionnaire SDK Android pour charger les différents OS désirés.

Menu outils → Android → Gestionnaire d'appareils Android pour créer ses émulateurs.



8.2 Chargement sur un appareil physique

L'avantage de ce système est de pouvoir tester réellement son application, surtout si celle-ci utilise des sensor qu'il est difficile voire impossible de tester avec un émulateur.

Pour ceci, il faut brancher un appareil, le configurer pour qu'il soit en mode debug. Celui-ci apparaîtra ensuite dans la liste des appareils lors de la compilation.

9 Lien entre deux Activités

10 Méthodes événementielles

11 Layout dynamique

12 Persistance de données

13 Utilisation d'un sensor