BERENGER Zoé
LANVIN Alexandre
Groupe PACT 2.1

Rapport de conception détaillé de l'application Android

SOMMAIRE

INTRODUCTION

Nous voulons programmer notre application à l'aide d'Android Studio. Sur ce logiciel, une page de l'application correspond à une activité qui elle-même correspond à un fichier .java.

Nous allons donc organiser notre algorithme autour de ces activités. Seules les méthodes essentielles au bon fonctionnement de notre application seront ici présentées ; certaines méthodes pourront par la suite être ajoutées pour enrichir ou personnaliser celle-ci.

Rappel du but de notre application :

Notre projet, nommé "e-Maisel", consiste en la mise en place d'une application Android permettant de faciliter la vie quotidienne des habitants de la maison des élèves. Plus précisément, le but est de permettre aux utilisateurs d'accéder aux machines-à-laver disponibles et, le cas échéant, de pouvoir les réserver. L'application doit de surcroît afficher le nombre de personnes dans la cuisine d'un étage défini par l'habitant de la Maisel qui peut aussi, de la même manière, voir quelles plaques sont occupées. Elle présente des fonctionnalités supplémentaires suivantes : l'affichage d'une notification dans les cas de contrôle de ménage ou de loyer à régler, le fait de pouvoir savoir si la personne a ou non du courrier et l'affichage des statistiques du site Babar. Ce projet s'inscrit dans le thème Campus du futur : il a en effet été pensé pour la Maisel, mais pourra aussi être adapté pour les futurs résidences des étudiants de Télécom.

1. Inventaire des méthodes propres à Android Studio

Ces méthodes seront utilisées sur toutes les pages de notre application.

- onCreate(Bundle savedInstanceState) : initialisation de l'activité.
- setContentView(int i) : appelle un layout (mise en page) en invoquant son identifiant de la forme R.layout.name_of_layout.
- findViewById(int i) : permet par l'intermédiaire de leur identifiant de récupérer les objets graphiques du layout.
- Intent(Context context, Class class) : crée un objet qui permet d'exécuter un lien entre le premier paramètre et le deuxième. On l'utilisera principalement pour lier deux activités entre elles.

Les prochaines méthodes sont propres aux objets graphiques du layout et permet d'écouter ces objets ou d'envoyer des erreurs :

- setOnEditorActionListener(TextView.OnEditorActionListener l) : permet d'avoir un listener qui est appelé dès qu'une action est effectué sur cet objet.
- onEditorAction(int actionCode) : définit l'action à effectuer lorsqu'une action sur l'objet écouté est effectuée.
- setOnClickListener(View.OnClickListener l) : permet d'avoir un listener qui est appelé dès qu'un appui est effectué sur cet objet.
- onClick(View view) : définit l'action à effectuer lorsqu'un appui sur l'objet écouté est effectué.
- setError(CharSequence error, Drawable icon) : dans un TextView, permet d'afficher l'icone sur le coin haut droit de la vue et d'afficher un message d'erreur.

2. LoginActivity

Cette page est la première page qui s'ouvre lors du lancement de l'application. Elle permet à l'utilisateur de s'identifier sur son compte et d'utiliser e-Maisel.

Objets :

- ❏ mLoginFormView (type View) : vue dans laquelle seront positionnés les objets suivants.
- ❏ mEmailView (type : AutoCompleteTextView) : champ dans lequel l'utilisateur rentre son adresse email.
- ❏ mPasswordView (type : EditText) : champ dans lequel l'utilisateur rentre son mot de passe.
- ❏ LIST_CREDENTIALS (type String[]) : tableau qui contient les noms d'utilisateurs et mots de passe connus de l'application.
- ❏ mSignInButton (type : Button) : bouton sur lequel l'utilisateur appuie pour finaliser son identification ou inscription.

Méthodes :

- attemptLogin() : essaie de se connecter à un compte utilisateur en fonction des paramètres rentrés dans mEmailView et mPasswordView en utilisant la classe UserLoginTask.
- isEmailValid(String email) : permet de vérifier que l'élément rentré est un email.
- isPasswordValid(String password) : vérifie que le mot de passe fait plus de 4 caractères.

Une classe UserLoginTask est de plus ajoutée afin d'implémenter notamment une méthode doInBackground(paramètres) qui connecte l'utilisateur si son email est reconnu ou ajoute ce dernier à la base de données si l'email n'y figure pas encore.

3. AccueilActivity

Cette page est la page centrale de l'application. Elle permet d'accéder aux différentes fonctionnalités que nous voulons proposer, qui sont implémentées dans d'autres activités et auxquelles on accède à l'aide d'un appui sur un bouton.

Objets :

- ❏ laverieButton, cuisineButton, babarButton, notificationButton, settingsButton (type Button) : boutons permettant d'accéder aux autres pages de l'application.
- ❏ accueilActionBar (type ActionBar) : une banière en haut de l'écran qui indique "Accueil"

Méthodes :

- Utilisation des objets OnClickListener, Intent, et de la méthode onClick(View view) pour ouvrir les autres pages de l'application lors de l'appui sur un des boutons créés.

4. LaverieActivity

Cette page se concentre sur la partie Laverie de notre application. Elle propose, grâce à un code couleur, de voir quelles machines sont disponibles, réservées ou utilisées ; l'utilisateur pourra de plus réserver une machine

Objets :

- ❏ button1, button2, button3, button4, button5, button6, button7, button8 (type Button) : boutons représentant les machines de numéro correspondant.
- ❏ reserveButton (type Button) : permet de réserver une des machines disponibles.
- ❏ statButton (type Button) : permet d'afficher la page des statistiques.
- ❏ laverieActionBar (type ActionBar): une banière en haut de l'écran qui indique "Laverie"
- ❏ LIST_AVAILABLE (type Int[]) : tableau des numéros des machines disponibles.

Méthodes :

- setChange(Attribut attribut) : permet de changer l'attribut d'une machine (disponible, réservée ou utilisée) et de changer la couleur du bouton en même temps.
- isAvailable(int i) : renvoie un booléen en fonction de si la machine n°i est disponible ou non.
- isReserved(int i) : renvoie un booléen en fonction de si la machine n°i est réservée ou non.

- makeReservation() : permet de réserver une des machines disponibles. Affiche un message si aucune machine n'est disponible. La réservation s'annule après 5 minutes.
- updateAvailable() : met à jour le tableau de machines disponibles.

5. CuisineActivity

Cette page traite de la partie relative à la cuisine de notre application. Elle permet de voir le nombre de personnes dans une cuisine donnée ainsi que le nombre de plaques disponibles. L'utilisateur peut de plus lancer un minuteur (ce qui concrètement est utile pour la hotte).

Objets:

- ❏ statButton (type Button) : permet d'afficher la page des statistiques.
- ❏ minuteurButton (type Button) : permet de lancer le minuteur.
- ❏ nbPersonnes (type int) : indique le nombre de personnes présentes dans la cuisine.

- ❏ nbPlaques (type int) : indique le nombre de plaques disponibles dans la cuisine.
- ❏ cuisineActionBar (type ActionBar): une banière en haut de l'écran qui indique "Cuisine".
- ❏ etage (type : EditText) : champ dans lequel l'utilisateur rentre l'étage qui l'intéresse.

Méthodes :

- updateAvailable() : met à jour le nombre de personnes présentes et de plaques disponibles.
- isEtageValid(int i) : indique si l'étage entré est valide (i.e. s'il est compris entre 2 et 9.)

6. NotificationActivity

Cette page affiche les notifications récupérées dans la base de données avec un adresse email propre à notre application et les affiche pour les personnes concernées (dépend de l'étage rentré dans Paramètres).

Objets:

- ❏ notificationActionBar (type ActionBar): une banière en haut de l'écran qui indique "Notifications".
- ❏ updateButton (type Button) : permet de mettre à jour les notifications.
- ❏ deleteButton (type Button) : supprime le message selectionné.
- ❏ LIST_MESSAGES (type String[]) : tableau qui contient les messages à afficher (taille maximum précisée).

Méthodes:

- update() : met à jour les notifications.
- etageIsValid(int i) : détermine si l'étage i est concerné par l'email.
- display() : affiche le message lorsque l'on clique dessus.
- delete() : supprime le message.

7. BabarActivity

Etant donné que le site Babar a été créé par des Télécommiens, nous rentrerons en contact avec eux afin de pouvoir si possible avoir accès à leur base de données et juste afficher un tableau avec les statistiques de l'utilisateur du site Babar. Il faudra pour cela auparavant que l'utilisateur ait rentré son nom et prénom dans les paramètres.

8. StatisticsActivity

Cette page est une page qui permet d'afficher des statistiques sur l'affluence dans la laverie ou la cuisine d'un étage donné sur par exemple 2 semaines. Ainsi l'on peut sélectionner un étage, et cela affichera un diagramme de l'affluence en fonction de l'heure de la journée.

Objets :

- ❏ LIST_ACTIVITY (type int[]) : tableau de l'activité de la journée en fonction de l'heure, l'heure étant l'indice des cases du tableau de 0 à 23, et l'activité notée entre 0 et 10.
- ❏ statActionBar (type ActionBar): une banière en haut de l'écran qui indique "Statistiques"
- ❏ okButton (type Button) : bouton qui ferme la page

Méthodes :

- GridLayout(Context context) : va générer une grille sur la page pour pouvoir faire un histogramme dedans à l'aide de colonnes de carrés
- drawRect(float left, float top, float right, float bottom, Paint paint) : permet de créer des rectangles que l'on insérera dans la grille.
- getActivity(int i) : renvoie un entier qui représente l'activité d'une certaine heure.

9. SettingsActivity

Ceci est la page des paramètres: elle permet à un utilisateur de changer son mot de passe, et d'accéder à des informations.

Objets:

- ❏ infoButton (type Button) : permet d'afficher la page des informations.
- ❏ mPreviousPasswordView (type : EditText) : champ dans lequel l'utilisateur rentre son ancien mot de passe.
- ❏ mNewPasswordView (type : EditText) : champ dans lequel l'utilisateur rentre son nouveau mot de passe.
- ❏ mNewPasswordAgainView (type : EditText) : champ dans lequel l'utilisateur rentre son nouveau mot de passe à nouveau pour le confirmer.
- ❏ forgottenPasswordButton (type Button) : envoie un mail à l'utilisateur avec son ancien mot de passe.
- ❏ deconnectionButton (type Button) : déconnecte l'utilisateur.
- ❏ settingsActionBar (type ActionBar): une banière en haut de l'écran qui indique "Paramètres".

Méthodes:

- isNewPasswordValid(String password1, String password 2) : vérifie que les deux mots de passe rentrés sont les mêmes et qu'ils font plus de 4 caractères.

10. InformationsActivity

Cette page affiche les informations quant à l'application.

Objets:

- ❏ informations (type TextView) : texte donnant les informations relative à l'application.
- ❏ infoActionBar (type ActionBar): une banière en haut de l'écran qui indique "Informations".