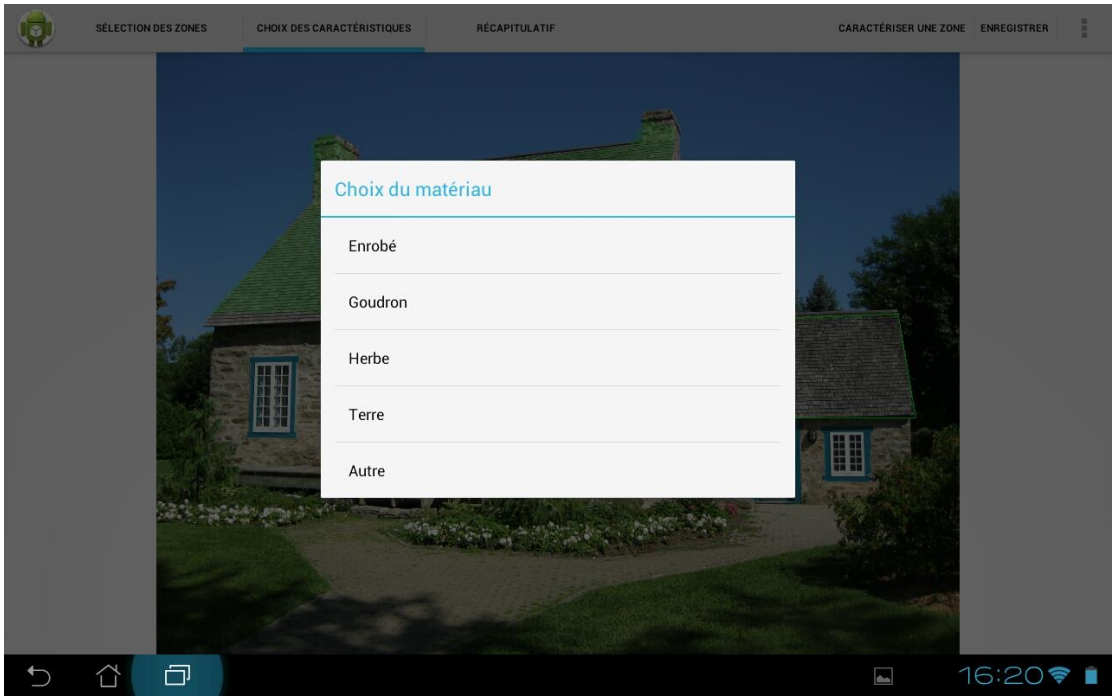
PROJET ANDROID D'EXTRACTION DE CARACTERISTIQUES DE FAÇADES ET SOLS SUR SITE A PARTIR D'IMAGES



Membres Projet

Patrick RANNOU
Jonathan COZZO

Equipe pédagogique

Vincent TOURRE
Myriam SERVIERES

EI3 INFO | Projet d'Application | 2012 – 2013

Sommaire

Introduction

Dans le cadre de notre dernière année d'étude dans l'option Informatique à l'École Centrale de Nantes, notre groupe de deux étudiants a réalisé un projet d'application en collaboration avec les enseignants-chercheurs Vincent Tourre et Myriam Servières. Il était question du développement d'une **application Android d'extraction de caractéristiques de façades et sols sur site à partir d'images**.

Ce rapport présente l'ensemble de notre travail, en commençant par l'élaboration du cahier des charges précis, puis la modélisation de l'application développée, l'étude de son interface puis de son fonctionnement. Nous aborderons enfin de possibles améliorations futures qu'il nous semble pertinent de mettre en place pour poursuivre le développement de l'application.

1. Présentation du projet

1.1. Existence d'un besoin

Pour générer des simulations microclimatiques, les chercheurs dans le domaine de la thermique ont besoin de renseigner sur les modèles 3D utilisés tout un ensemble de caractéristiques ; par exemple le type de matériaux, leurs couleurs, le pourcentage de vitrage, la position et la longueur des balcons, le type de sol, etc. Ces renseignements manquants peuvent être directement acquis sur site et intégrés comme informations ou annotations sur des images géo-localisées.

C'est dans cette optique que le souhait de développer une application utilisable sur tablettes Android a vu le jour. Son but est de permettre l'extraction, au moins manuelle, d'informations pour annoter une image géo-localisée qui pourra alors être projetée sur un modèle 3D en fonction de la position et l'orientation de l'appareil.

1.2. Cahier des charges

Le cahier des charges, document traitant des spécifications fonctionnelles et techniques pour la réalisation du projet, a été rédigé par l'équipe étudiante et validée par l'équipe pédagogique.

Il a été décidé que l'application réalisée présente trois grands aspects :

- Récupération de la position GPS ;
- Extraction, enregistrement et réouverture des caractéristiques sur les façades et sols de l'image : type de matériaux, couleur, pourcentage de vitrage, nombre d'étages du bâtiment, position suivant l'axe vertical et longueur des balcons ;
- Affichage graphique de ces caractéristiques sur l'image.

L'intégralité du cahier des charges est disponible en Annexe 1 – Cahier des charges.

2. Modélisation de l'application

A partir du cahier des charges, nous avons pu réaliser des diagrammes UML afin de décrire formellement le fonctionnement de notre application et les différentes interactions entre l'utilisateur et le logiciel. Nous présenterons ici les principaux points permettant de comprendre le fonctionnement du programme. Pour plus de détails, sont disponibles en annexes, le guide utilisateur (Annexe 4 – Guide d'utilisation) ainsi que les diagrammes de classes (Annexe 5 – Diagrammes de classe).

2.1. Diagrammes de cas d'utilisation

2.1.1. Présentation générale de l'application

Au lancement, le programme propose à l'utilisateur de charger une image préenregistrée dans la mémoire de l'appareil ou de prendre une photo grâce à l'appareil photo interne à la tablette.

Une fois l'image validée par l'utilisateur, l'écran principal de l'application s'affiche et lui propose d'effectuer un traitement de l'image par la délimitation de façades et la définition de leurs caractéristiques (voir partie 2.2.1 Présentation du traitement de l'image pour plus de détails).

Il est permis à l'utilisateur de visualiser, à tout moment, l'ensemble des informations qu'il a annoté sur l'image et également de les sauvegarder (dans un fichier XML).

Le diagramme de cas d'utilisation générale de l'application ci-dessous synthétise le fonctionnement général du programme.

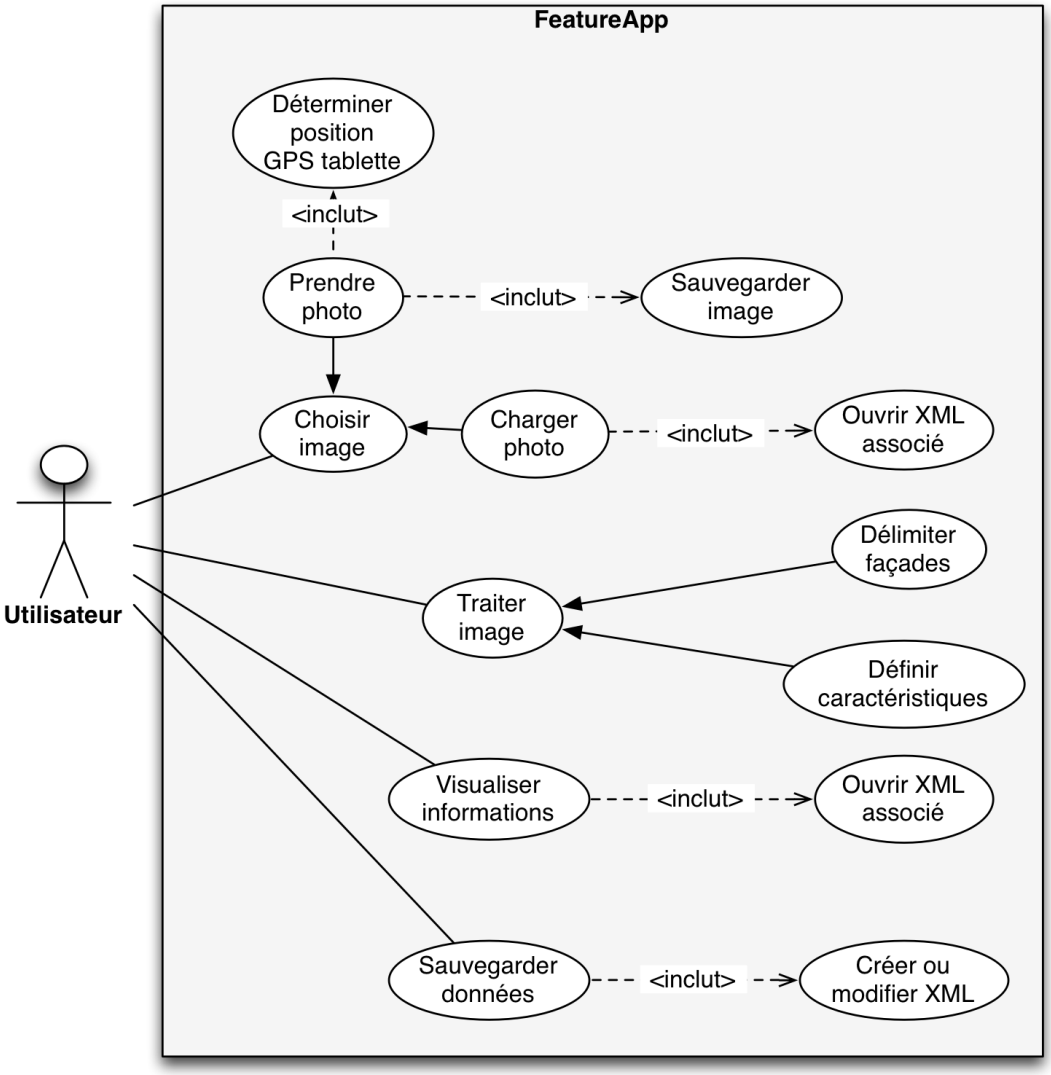


IMAGE 2.1 - CAS D'UTILISATION GENERALE DE L'APPLICATION

2.1.2. Présentation du traitement de l'image

Les fonctionnalités de traitement d'image représentent la partie la plus importante de l'application puisqu'elles ont pour but de répondre au besoin même pour lequel il a été décidé de développer ce programme.

On peut regrouper ces fonctionnalités en deux grands ensembles.

La délimitation de façade
L'utilisateur peut, à l'aide de son doigt ou d'un stylet pour plus de précision, délimiter une façade en cliquant simplement sur les sommets de cette dernière. Une fois le contour fermé, la façade est validée et prête à être caractérisée.

Il est également possible d'ajouter d'autres façades à tout moment, mais aussi de les grouper ou dégrouper suivant la volonté de l'utilisateur.

La définition des caractéristiques

Lorsqu'une façade a été délimitée, il est possible d'en définir ses caractéristiques : en y cliquant dessus, un premier choix doit être fait par l'utilisateur pour distinguer la nature de la façade (Façade, Sol ou Toit) puis son type de matériau (choix dans une liste ou ajout à la main, par le clavier virtuel de la tablette).

Il est ensuite possible de choisir la couleur du matériau, d'indiquer si la façade est un balcon (et le cas échéant, sa hauteur estimée) ainsi que le nombre d'étages constituant le bâtiment.

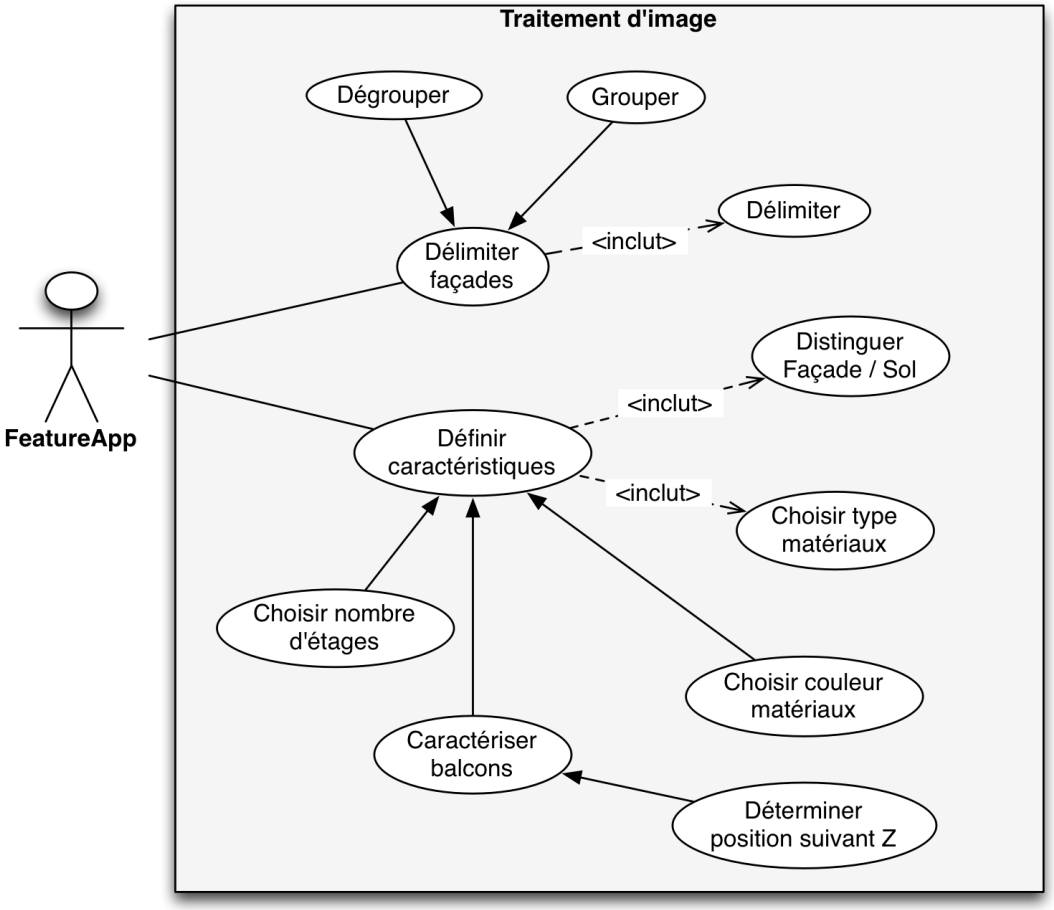


IMAGE 2.2 - CAS D'UTILISATION DU TRAITEMENT DE L'IMAGE

2.2. Diagrammes d'états-transition

2.2.1. Vue générale du fonctionnement

La navigation entre les différents modes de fonctionnement de l'application se fait par le passage d'une « Activité » à l'autre (spécificité des applications Android) par l'intermédiaire de boutons.

Ainsi, au démarrage de l'application, celle-ci se trouve dans l'activité « WelcomeActivity » et, après le choix de l'image à traiter effectué par l'utilisateur, le programme se dirige vers l'activité principale « MainActivity » qui sera celle où il réalisera le traitement de l'image.

L'appuie sur le bouton « Enregistrer » crée ou met à jour le fichier XML associé à l'image et conservant l'ensemble des informations annotées à l'image.

L'appuie sur la flèche de retour affiche un texte de confirmation et ramène au début de l'application, permettant ainsi à l'utilisateur de choisir une nouvelle image.

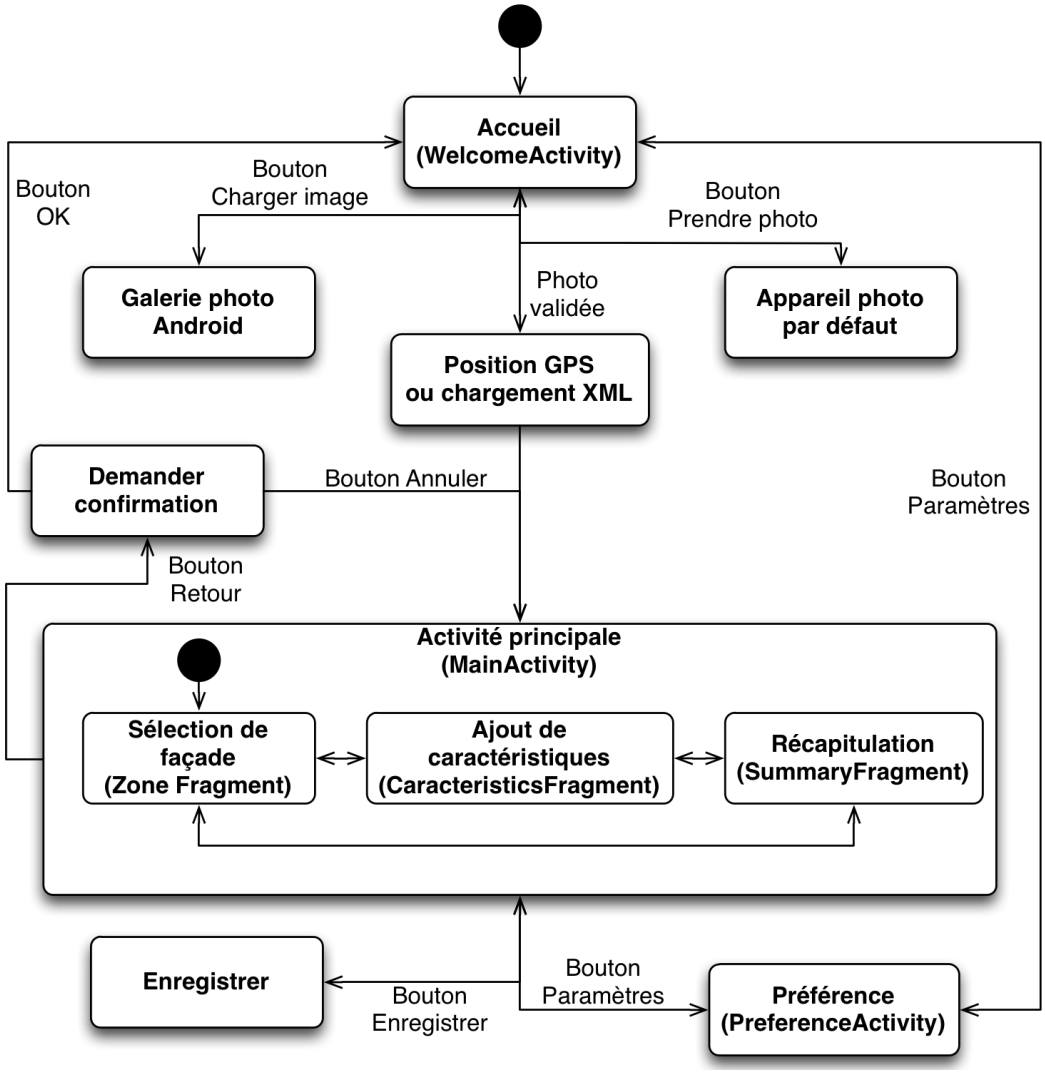


IMAGE 2.3 - ETATS TRANSITIONS DU FONCTIONNEMENT GENERAL

2.2.2. Détail de la sélection de façade

Une fois dans l'activité « MainActivity », l'utilisateur peut réaliser un travail de sélection de façade avec, comme nous l'avons présenté précédemment, la possibilité d'ajouter, grouper, séparer et supprimer des zones. Chacune de ces actions est réalisable par un simple clic sur un bouton pour entrer dans le mode correspondant.

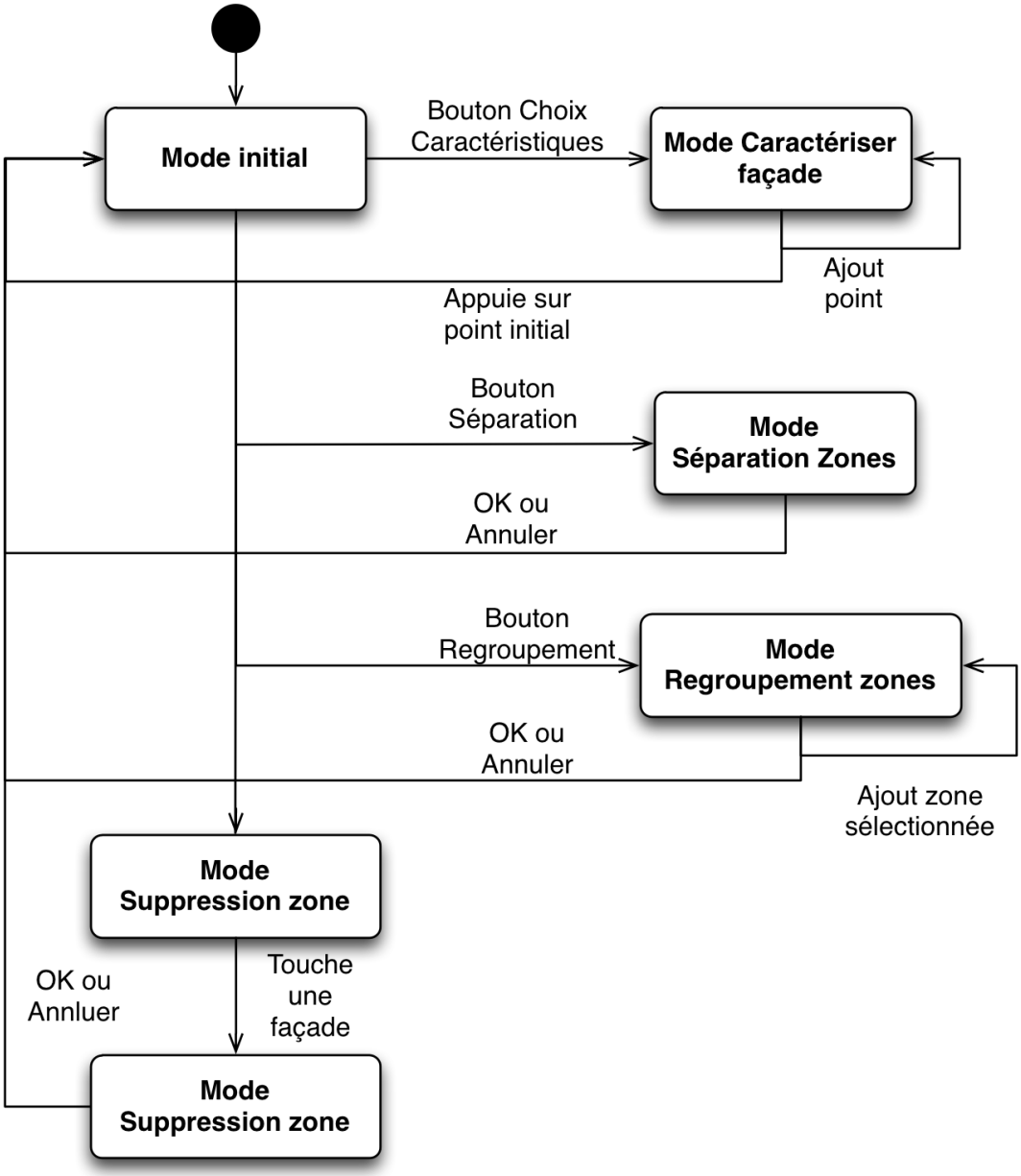


IMAGE 2.4 - ETATS TRANSITIONS DE LA SELECTION DE FAÇADES

2.2.3. Détail de la caractérisation de façade

Une fois une façade créée, il est possible de lui ajouter des informations à partir du mode de caractérisation de façades. Les différentes informations que l'application prend en compte sont le type de façade, son matériau, sa couleur, si elle est de type balcon ainsi que le nombre d'étage du bâtiment de la photo.

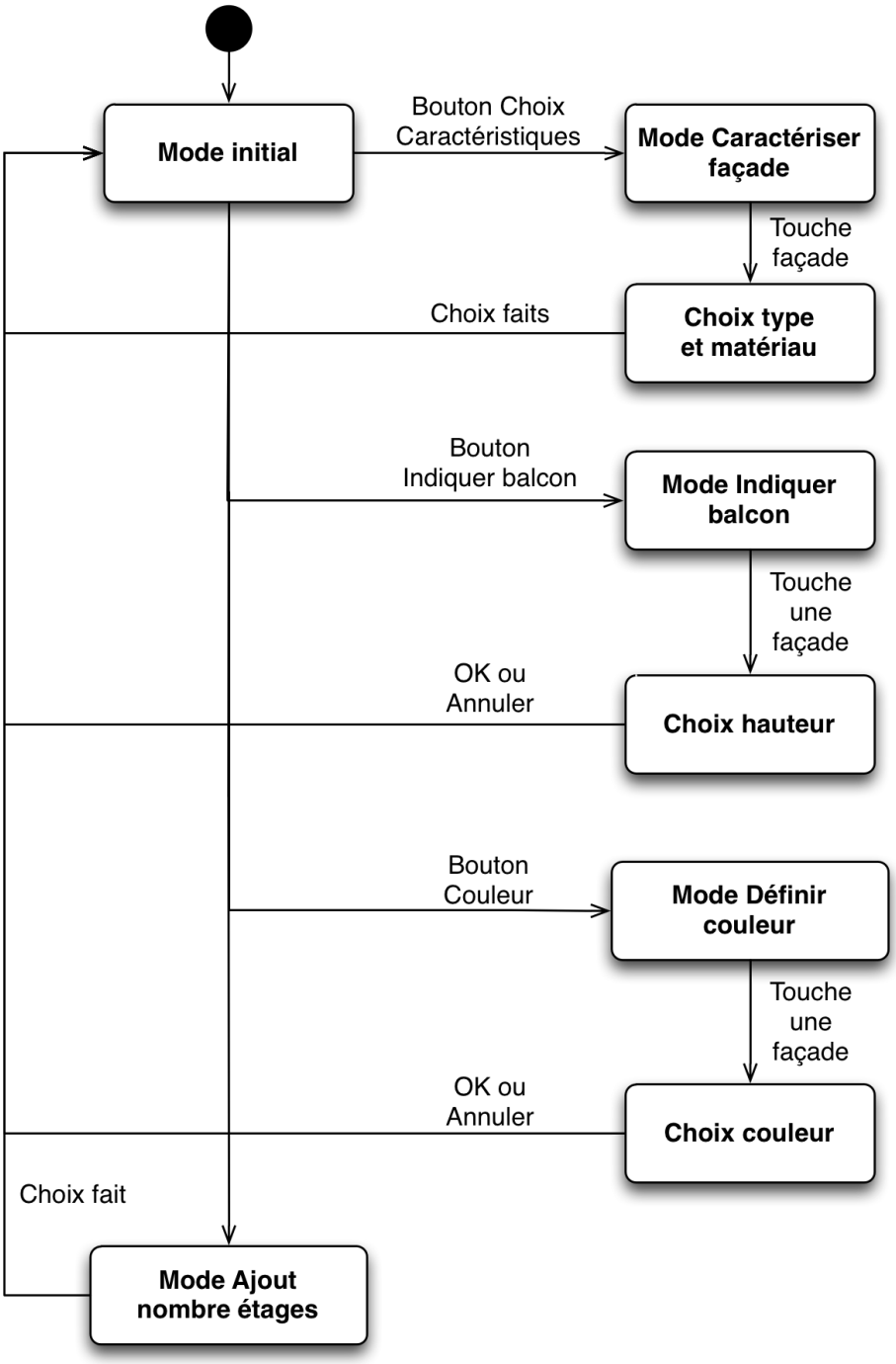


IMAGE 2.5 - ETATS TRANSITION DE LA CARACTERISATION DE FAÇADE

2.2.4. Détail de l'affichage des informations

Enfin, l'onglet « Récapitulatif » permet à l'utilisateur de visualiser en détail toutes les informations qu'il a ajoutées à l'image.

Il lui est possible d'afficher les informations particulières à une façade en y cliquant dessus mais aussi à la totalité de l'image en cliquant sur le bouton « Afficher les informations globales ».

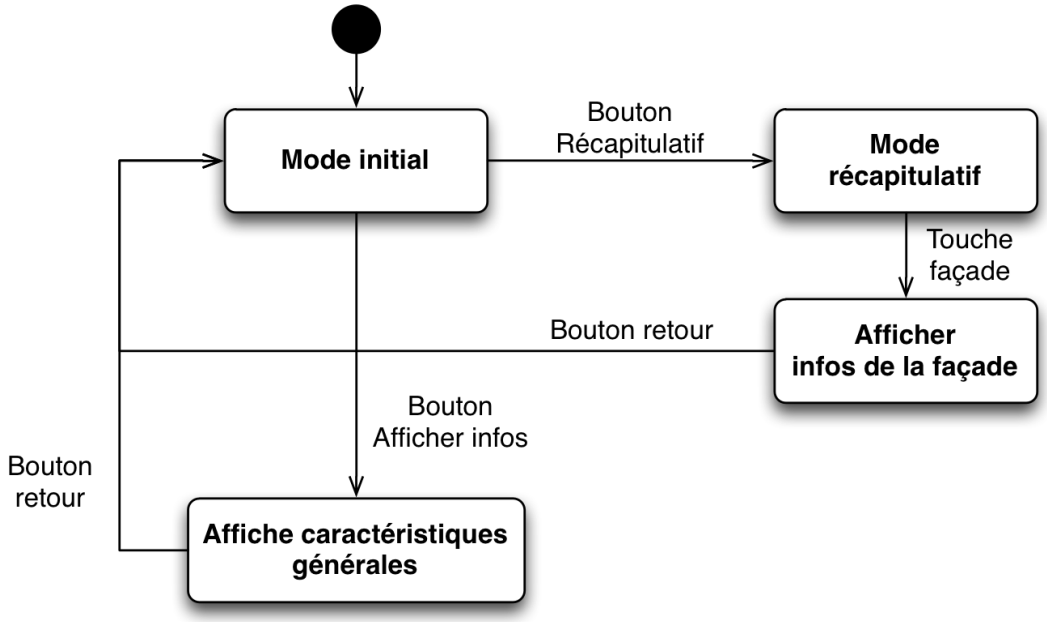


IMAGE 2.6 - ETATS TRANSITIONS DU RECAPITULATIF

3. Interface de l'application

Concernant le design de l'interface utilisateur de l'application, le choix a été fait de rester **proche des recommandations** fournies par le guide de design d'Android[1]. En effet, cela permet de garder **une cohérence et une unité** avec les autres applications utilisées par l'utilisateur. Par ailleurs, cela **facilite également l'utilisation** de FeatureApp car l'utilisateur est déjà habitué à une interface similaire et comprend donc plus rapidement le fonctionnement général (emplacement des boutons, des paramètres, fonctionnement des onglets…).

Il faut noter que l'interface utilisée est celle proposée à partir de la version 3.0 d'Android (première version à destination des tablettes). L'application ne sera donc pas compatible avec les versions d'Android antérieures.

3.1. Utilisation de "l'action bar"

Nous avons choisi **d'utiliser « l'action bar »** pour présenter les menus de notre application. Il s'agit de la barre de menu proposée par Android et qui affichée en haut de l'application[2]. Elle est constituée de plusieurs boutons personnalisés, qui sont, **selon leur importance et la place disponible à l'écran**, affichés dans cette barre ou dans un menu déroulant.

Par ailleurs, pour bien distinguer les différentes parties de notre application (sélection de façades, caractérisation de celles-ci et affichage des informations), nous avons choisi **d'utiliser des onglets**. On a donc trois onglets, qui, lorsqu'ils sont sélectionnés, **modifient les boutons affichés** dans l'action bar.

IMAGE 3.1 – ACTION BAR DE L'APPLICATION

IMAGE 3.2 – ACTION BAR AVEC LE MENU DEROULANT

Un autre des avantages d'utiliser l'action bar est **l'adaptation automatique à la taille de l'écran**. Cela est particulièrement intéressant lors du passage du mode paysage au mode portrait. En effet, dans ce cas, si l'espace n'est pas suffisant pour afficher tous les onglets, ils sont présentés sous la forme d'une liste déroulante.

[1] Cf. bibliographie (1)
[2] Plus de détails : bibliographie (2)

IMAGE 3.3 - ACTION BAR EN MODE PORTRAIT

De plus, l'action bar peut être modifiée temporairement pour faire apparaître **un menu contextuel**. Cette possibilité a été utilisée pour implémenter **le regroupement de zones** : quand l'utilisateur appuie sur le bouton correspondant le menu contextuel s'ouvre et permet à l'utilisateur de choisir plusieurs zones avant de valider son choix (ou d'annuler l'action).

IMAGE 3.4 - MENU CONTEXTUEL POUR LE REGROUPEMENT DE ZONES

3.2. La page principale

Concernant la page principale, nous affichons bien sûr la photo sur toute la page. Des lignes représentent les différentes zones (tracées par l'utilisateur) et sont de couleurs différentes selon l'état de celles-ci (rouge pour les non terminées et vert pour les autres). C'est sur cette image que l'utilisateur effectue les opérations de sélection de zones (les zones sélectionnées sont remplies de manière semi-transparente).

IMAGE 3.5 - PAGE PRINCIPALE DE L'APPLICATION

3.3. Les boites de dialogue

Pour permettre à l'utilisateur d'entrer des informations ou de confirmer (ou pas) une action, nous avons utilisé **les boites de dialogues** d'Android. Cela permet d'interagir **sans** pour autant **ouvrir une autre activité**. Elles permettent également, tout comme l'action bar, de conserver une homogénéité avec les autres applications Android.

Lorsqu'une boite de dialogue s'ouvre, elle est située en centre de l'écran et l'arrière-plan est légèrement grisé. Elle peut être quittée soit avec les boutons de celle-ci, soit avec un appui hors de la boite de dialogue.

IMAGE 3.6 - BOITE DE DIALOGUE (SUPPRESSION D'UNE ZONE)

Il existe plusieurs boites de dialogue classiques qui sont déjà implémentées, que l'on peut donc utiliser facilement, notamment pour la sélection dans une liste, l'entrée d'une chaine de caractère ou encore le choix d'un entier.

IMAGE 3.7 - BOITE DE DIALOGUE (CHOIX DU MATERIAU)

IMAGE 3.8 - BOITE DE DIALOGUE (INSERTION DU NOMBRE D'ETAGES)

Pour la boite de dialogue permettant de choisir la couleur d'une zone, nous avons utilisé une boite de dialogue distribuée sous la licence Apache 2.0 car il n'existait pas de boite de dialogue prédéfinie[3].

[3] Cf. bibliographie (4)

IMAGE 3.9 - BOITE DE DIALOGUE (CHOIX DE LA COULEUR)

3.4. Les préférences utilisateur

De plus, nous avons choisi d'ajouter une **page de réglage des préférences** pour permettre à l'utilisateur de personnaliser certains paramètres de l'application. Pour l'instant, un seul paramètre est modifiable : la précision de la détection de la fin du tracé d'une zone. En effet, les utilisateurs peuvent souhaiter une précision différente selon leur matériel (utilisation d'un stylet ou non…).

Cependant, nous avons utilisé **la page de gestion des préférences fournie par Android**, ce qui signifie qu'il sera très simple d'ajouter de nouveau paramètres ainsi que de les classer par catégories. De plus, cela **respecte le guide sur l'interface graphique** d'Android.

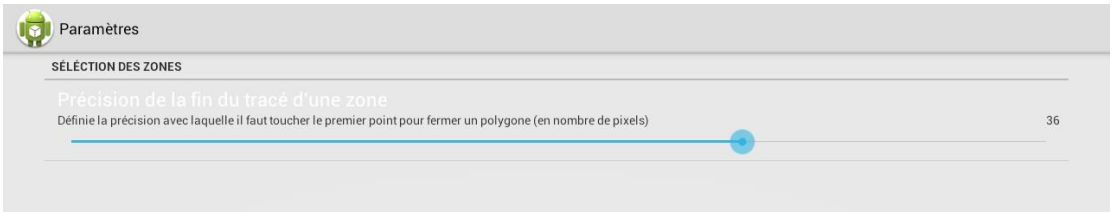


IMAGE 3.10 - PAGE DE GESTION DES PARAMETRES

La page de préférence est celle standard d'Android, cependant, pour pouvoir y afficher une barre de défilement pour choisir un entier, nous avons utilisé des fonctions spécifiques (Cf. Bibliographie (5)).

4. Fonctionnement de l'application

4.1. Organisation générale de l'application

Comme pour toutes applications Android, nous avons dû utiliser des activités pour représenter les différents écrans de l'application[4]. Il y a donc une activité pour l'écran d'accueil, une pour les préférences et une pour la page principale. Nous avons également utilisés des fragments[5] pour créer les différents onglets dans la page principale (un fragment par onglet).

4.1.1. Organisation des paquets

Les différentes classes de l'application ont été réparties dans plusieurs paquets pour une meilleure lisibilité et compréhension :

- *Activities :* Contient toutes les activités de l'application ;
- *Dialogs :* Contient l'implémentation de toutes les boites de dialogue ;
- *Fragments :* Regroupe les fragments de l'application ;
- *Utils :* Regroupe certaines fonctions utilitaires (pour afficher l'image, trouver la position GPS,…) ;
- *Utils.Colorpicker :* Contient les fonctions permettant l'affichage de la boite de dialogue de choix des couleurs ;
- *Zones :* Contient les fonctions permettant la mémorisation des informations entrées par l'utilisateur (zones, ensembles de zones, balcons…).

4.1.2. Diagrammes de classe

Pour une meilleure lisibilité, on sépare les diagrammes de classes en plusieurs sous-diagrammes, avec des versions simplifiées pour certains. Les diagrammes complets sont disponibles en annexe (Cf. Annexe 5 – Diagrammes de classe).

Diagramme des activités et des fragments

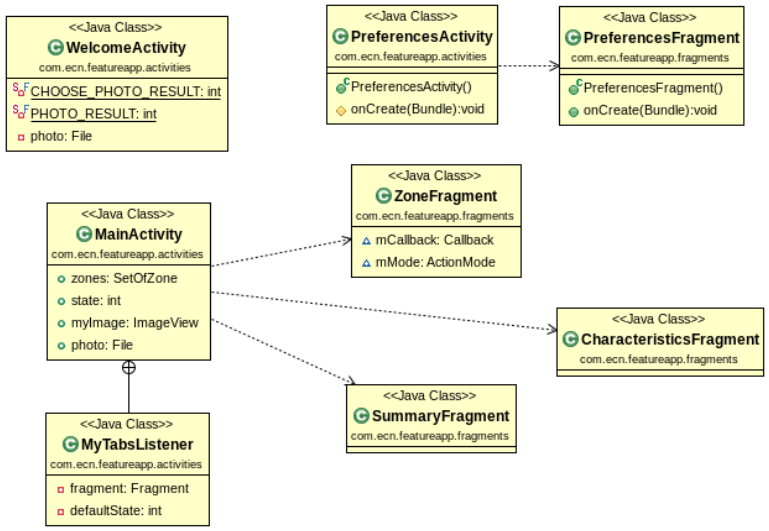


DIAGRAMME 1 - DIAGRAMME DE CLASSE SIMPLIFIE (ACTIVITES ET FRAGMENTS)

[4] Pour plus de détails sur le fonctionnement des activités : cf. bibliographie (6)
[5] Cf. bibliographie (7)

L'activité principale utilise les différents fragments. Par ailleurs, il y a aussi une classe *MyTabsListener* qui permet de réagir au toucher sur les différents onglets et de lancer le fragment correspondant.

On peut également remarquer que dans les différentes classes, on redéfinie certaines méthodes de la classe comme *onCreate* par exemple.

Diagramme sur la gestion des zones

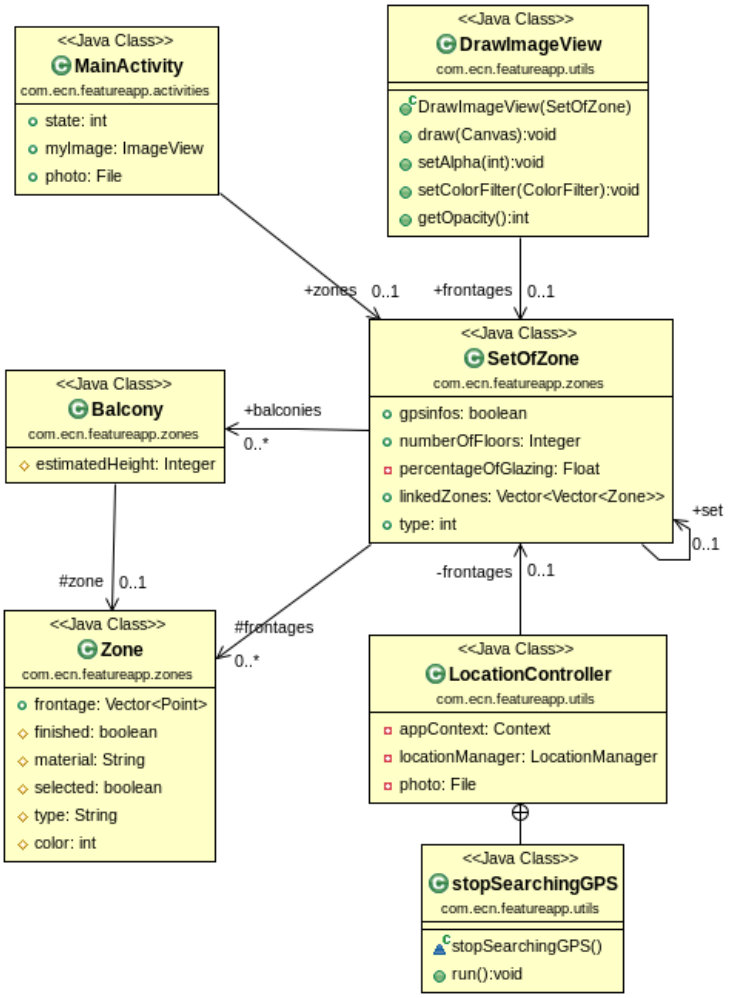


DIAGRAMME 2 - DIAGRAMME DE CLASSE SUR LA GESTION DES ZONES (SIMPLIFIE)

Pour gérer les informations entrées par l'utilisateur, on a d'abord une classe *SetOfZone* qui regroupe les informations sur toutes les zones et sur l'image en général (nombre d'étages…). Cette classe contient notamment un vecteur de la classe Zone. Cette dernière contient les informations sur une zone en particulier (la liste de ses points, son type, son matériau…).

La classe *DrawImageView* permet de dessiner les zones ajoutées par l'utilisateur sur l'image. Elle contient dont une référence vers un élément de la classe *SetOfZone*.

4.2. Création et gestion de l'interface graphique

4.2.1. Récupération des photos

Il y a deux façons pour déterminer l'image qui sera traitée :

- Choisir une photo dans la galerie photo de la tablette ;
- Prendre une nouvelle photo (qui sera enregistrée dans le dossier featureapp sous un nom créé automatiquement en fonction de la date et de l'heure).

Pour réaliser le choix de la photo, nous avons utilisé le code de l'application Ombre[6] en y apportant plusieurs améliorations. Tout d'abord, selon le choix de l'utilisateur**, on crée et on appelle un *Intent*** pour permet de récupérer ou de prendre une photo. Cela a pour effet de lancer **les applications par défaut du terminal** mobile réalisant ces opérations. Lorsque l'utilisateur a choisi ou pris une photo, la classe *WelcomeActivity* est relancée avec la méthode *onActivityResult*. Cette méthode vérifie que l'opération s'est bien déroulée et si une photo a bien été récupérée, elle lance l'activité principale.

Un point important à prendre en compte est que la galerie par défaut d'Android affiche également **les photos stockées en ligne** et liées au compte Google de la tablette. Il a fallu **prendre en compte ce cas-là** car sinon l'application s'arrête brutalement si l'utilisateur choisi une photo en ligne. Nous avons choisi de ne pas gérer ce cas pour l'instant car cela nécessitait de nombreux développement, que ce soit pour récupérer la photo ou pour l'enregistrer. Il y a cependant un message demandant à l'utilisateur de choisir une photo disponible en local[7].

4.2.2. Récupération de la position GPS

La récupération de la position GPS a lieu dans la classe *LocationController* (**qui implémente *LocationListener***). La recherche de la position prend fin automatiquement lorsqu'elle a été déterminée et enregistrée. De plus, un message d'erreur est affiché si le GPS n'est pas activé sur la tablette.

Cependant, il faut aussi prendre en compte **la situation où le GPS est activé mais où il n'arrive pas à déterminer la position** (à l'intérieur d'un bâtiment par exemple). Dans ce cas, laisser la recherche de position active userait trop la batterie et n'aurait que peu d'intérêt. C'est pourquoi un *timer* a été utilisé et il arrête la rechercher si aucune position n'a été trouvée après 20 secondes, tout en affichant un message.

L'enregistrement des coordonnées GPS a lieu dans les EXIF de l'image. Pour cela, on utilise la classe *ExifInterface* fournie par Android. Cependant, les coordonnées récupérées avec la tablette sont en degrés décimaux alors que l'on doit utiliser des coordonnées en « Degrés, Minutes, Secondes ». Il faut donc penser à **convertir les coordonnées géographiques** avant[8].

4.2.3. Affichage de l'image et des zones

Tout d'abord, il est important de connaître une spécificité importante d'Android : chaque application dispose d'une **limite concernant la quantité de mémoire** qu'elle peut utiliser. Si elle la dépasse, cela engendre une erreur et l'application ferme inopinément. Cela est important car cette limite peut être facilement dépassée **si on charge des images trop grandes**. Pour afficher une image, il est donc conseiller de **les redimensionner avant[9]** (de façon à ne pas charger une image ayant beaucoup plus de pixels que l'écran). Pour notre application, nous avons limité la tailler maximale de l'image à 1000 pixels.

[6] Cf. bibliographie (8)
[7] Pour l'implémentation, la référence (9) a été utilisée.
[8] Concernant la recherche de la position GPS, les références (10), (11) et (12) sont utiles.
[9] Code disponible (Cf. bibliographie (13)).

Concernant l'affichage des zones, la classe *DrawImageView* est utilisée. E**lle implémente la classe *Drawable*** et redéfinit la fonction *draw*. A présent celle-ci dessine les polygones représentant les zones (elle a en paramètre l'objet de type *SetOfZone* utilisé par l'application). Quand l'utilisateur modifie les zones, l'application modifie directement l'objet de type *SetOfZone* puis **appelle la méthode *invalidate*** sur l'objet *DrawImageView* qui a pour effet de **redessiner les zones**[10].

4.2.4. Gestion de la rotation de l'écran

Un point important à connaître lors du développement sous Android est que une activité est **automatiquement détruite puis ré-ouverte lors d'un changement de configuration** (typiquement une rotation de l'écran)[11]. Cela entrainerait une disparition des modifications effectuée par l'utilisateur. Pour contourner ce problème, on **peut passer un objet** (ici de type *SetOfZone*) entre l'ancienne et la nouvelle version de l'activité grâce aux fonctions *onRetainNonConfigurationInstance* et *getLastNonConfigurationInstance*. Par conséquent, seules les modifications stockées dans l'objet de type *SetOfZone* seront conservées lors d'une rotation de l'écran. Par exemple, si l'utilisateur commence à écrire un nom de matériau personnalisé et qu'il tourne l'écran avant de l'avoir validé, ce nom disparaîtra et il devra l'écrire à nouveau. Cependant, nous avons choisi de ne pas régler ce cas (ni celui similaire qui intervient lors du choix d'une couleur) car cela dégraderait les performances de l'application et n'aurait que peu d'intérêt (l'utilisateur changeant rarement l'orientation de son appareil tout en écrivant un mot).

On peut également noter qu'Android détruit également l'application si elle reste trop longtemps en arrière-plan et que le système a besoin de plus de mémoire. Ce cas-là n'est pas gérer par l'application et les modifications effectuées disparaissent donc.

4.2.5. Utilisation des fragments

Pour pouvoir créer différents onglets dans l'application, nous avons utilisé des fragments. Dans *MainActivity*, une classe (*MyTabsListener*) se charge de repérer les appuis sur les différents onglets et remplace le fragment actuel par celui correspondant à l'onglet choisi. Le changement de fragment change uniquement les menus affichés et éventuellement l'état par défaut de l'application (action lors d'un appui sur une zone). Les actions correspondant aux boutons du menu sont donc définies dans les différents fragments.

4.3. Gestion des zones

4.3.1. Gestion générale des différentes zones

Toutes les informations entrées par l'utilisateur sont stockées dans un objet de type *SetOfZone*. Cette classe contient notamment un vecteur d'éléments de type *Zone*, le nombre d'étages du bâtiment et la liste des balcons. L'objet de type *Zone* contient lui les coordonnées des points (en pixels) du polygone représentant cette zone ainsi que son type, son matériau…

Un objet de type *Balcony* est constitué de la zone associée ainsi que de la hauteur estimée du balcon. Cette classe n'a que peu d'attributs pour l'instant mais elle sera amenée à évoluer

[10] Cf. bibliographie (14)
[11] Cf. bibliographie (15)

dans les prochaines versions de l'application selon les informations concernant le balcon que l'on souhaite pouvoir stocker.

4.3.2. Fonctions de manipulation des zones

De nombreuses méthodes ont été implémentées pour manipuler ces zones et ces ensembles de zones. Voici les principales :

Calcul de la surface des zones

Le calcul de la surface des zones est indispensable pour trouver la zone la plus petite[12] ainsi que pour calculer le pourcentage de vitrage.

L'algorithme est celui détaillé par Darel Rex Finley[13]. Il faut noter que celui fournit des résultats non intuitifs pour des polygones qui se croisent eux-mêmes. Cependant, cela ne pose pas de problème pour notre application.

Vérification d'appartenance d'un point à une zone

La vérification d'appartenance d'un point à une zone est également très importante. Il y a tout d'abord la méthode *isInsideFrontage* (dans *SetOfZone*) qui retourne le numéro de la plus petite zone contenant ce point (ou -1 si aucune zone ne le contient). Cette méthode utilise une méthode *containPoint* (dans *Zone*) (cf. bibliographie (17) pour l'algorithme).

Calcul du pourcentage de vitrage

Le calcul du pourcentage de vitrage dépend évidemment des informations entrées par l'utilisateur. Cependant, il n'est pas nécessaire de le recalculer à chaque changement. En effet, le changer uniquement lors de l'enregistrement des données et de leur affichage est suffisant et évite des opérations inutiles.

Pour l'instant, le pourcentage est calculé à partir de l'ensemble des aire en matériau « verre » sur l'ensemble des aires des façades (sans prendre en compte les façades qui sont en verre). En effet, on considère que les parties vitrées sont forcément à l'intérieur d'une autre façade (typiquement un mur).

4.4. Import et export XML

Afin de réaliser une sauvegarde des données annotées à une image, nous avons décidé d'écrire l'ensemble de ces informations dans un fichier XML associé à l'image. Ce choix permet par là même d'assurer une réouverture possible d'une image traitée tout en conservant le travail de caractérisation qu'avait réalisé l'utilisateur.

L'écriture d'informations dans un fichier XML est appelée la sérialisation d'objet. Dans l'autre sens (lecture XML vers instanciation d'un objet), on parle de désérialisation.

4.4.1. La librairie XStream

Par souci de simplicité et de respect de délais, nous avons choisi d'implémenter une librairie préexistante permettant ces deux actions de sérialisation et désérialisation. Nous avons opté

[12] Utile pour savoir quelle zone l'utilisateur souhaite sélectionner lorsqu'il appuie sur un endroit contenant plusieurs zones.
[13] Cf. bibliographie (16)

pour la librairie XStream[14], libraire Java open source sous licence BSD (Berkeley Software Distribution license)[15].

Cette librairie permet de créer un fichier XML à partir de tout objet instancié d'un programme Java, en prenant en compte l'ensemble de ses attributs et en respectant l'indentation propre au format XML.

```
<com.ecn.featureapp.zones.SetOfZone>
  <balconies>
    <com.ecn.featureapp.zones.Balcony>
      <estimatedHeight>199</estimatedHeight>
      <zone>
        <type>Toit</type>
        <material>Cuivre</material>
        <frontage>
          <android.graphics.Point>
            <x>1713</x>
            <y>575</y>
          </android.graphics.Point>
          <android.graphics.Point>
            <x>1716</x>
            <y>924</y>
          </android.graphics.Point>
          <android.graphics.Point>
            <x>1945</x>
            <y>958</y>
          </android.graphics.Point>
          <android.graphics.Point>
            <x>1945</x>
            <y>637</y>
          </android.graphics.Point>
          <android.graphics.Point>
            <x>1810</x>
            <y>604</y>
          </android.graphics.Point>
          <android.graphics.Point>
            <x>1726</x>
            <y>577</y>
          </android.graphics.Point>
          <android.graphics.Point reference="../android.graphics.Point"/>
        </frontage>
        <finished>true</finished>
        <selected>false</selected>
        <color>-16777216</color>
      </zone>
    </com.ecn.featureapp.zones.Balcony>
  </balconies>
  <frontages>
    <com.ecn.featureapp.zones.Zone>
      <type>Façade</type>
      <material>Béton</material>
      <frontage>
        <android.graphics.Point>
          <x>982</x>
          <y>846</y>
```

IMAGE 4.1 - FICHIER XML RESULTANT DU TRAITEMENT D'UNE IMAGE

Bien que la librairie XStream soit totalement compatible avec Java, elle ne l'est pas de manière complète nativement avec Android. Il a donc été nécessaire de rechercher une version de XStream modifiée pour fonctionner parfaitement avec Android.

4.4.2. La sérialisation

Afin de pouvoir sérialiser un objet, il est nécessaire qu'il possède une méthode appelant la librairie XStream. Ainsi, pour sérialiser l'ensemble des informations relatives à une image, il est nécessaire d'exporter toutes les façades qu'elle contient. C'est la raison pour laquelle la méthode de sérialisation, appelée *save*, est située dans la classe SetOfZone, qui regroupe toutes les façades d'une image.

[14] Cf. bibliographie (18)
[15] Cf. bibliographie (19)

La sérialisation débute par la création d'un fichier XML vide dont le nom et le lieu de stockage sont les mêmes que ceux de la photo correspondant. Puis, à partir des fonctionnalités offertes par XStream, ce fichier XML est alors rempli avec les instances de l'objet.

Lors du fonctionnement du programme, c'est dans l'activité principale (MainActivity) que l'on fait appel à cette méthode *save* lorsque l'utilisateur appuie sur le bouton « Enregistrer ».

4.4.3. La désérialisation

De même que pour la méthode *save*, nous avons développé une méthode *open*, là aussi située dans la classe SetOfZone, qui permet d'instancier un objet de type SetOfZone à partir des informations lues dans un fichier XML dont le nom est le même que celui de la photo en cours d'utilisation par le programme.

L'application fait appel à cette méthode *open* lors du choix d'une image au lancement de l'application : un test est fait pour savoir si l'image choisie est issue de la mémoire de la tablette ou prise par l'appareil photo ; en effet, dans le second cas, aucun fichier XML n'existe puisque la photo vient juste d'être créée. Ainsi, lorsqu'une image est chargée depuis la mémoire, son fichier XML associé est également lu et instancie donc un objet de type SetOfZone permettant à l'utilisateur de reprendre son travail de caractérisation de façades là où il avait sauvegardé la fois précédente.

Lors de la sauvegarde, le fichier XML est écrasé et remplacé par un nouveau, contenant toutes les informations nouvellement ajoutées.

5. Améliorations possibles

L'ensemble des objectifs fixés par le cahier des charges a été atteint. Néanmoins, avec seulement 11 semaines consacrées à ce projet, l'application créée n'est bien évidemment pas dans sa version finale : toutes les fonctionnalités implémentées fonctionnent parfaitement mais il sera intéressant d'en développer de nouvelles afin de **rendre le programme plus complet**, toujours dans le but de satisfaire au mieux les utilisateurs.

Nous pensons ainsi qu'il pourra être intéressant d'implémenter **des fonctions de zoom et dézoom sur l'image** lors du traitement. Cela aura pour conséquence de permettre encore plus de précision dans la délimitation et le tracé des zones que ce que l'application offre actuellement.

Toujours dans un souci d'optimisation, il pourra être utile de chercher à **améliorer la précision des données GPS obtenues** grâce à la tablette. En effet, dans le cadre d'une application de réalité augmentée, la connaissance de la position exacte de l'endroit où l'image est prise détermine à elle seule les éléments composant cette image. Il sera notamment possible de s'appuyer sur le travail réalisé dans le cadre du projet Sig-Ar, dirigé par Myriam Servières, utilisant une carte tirée de Google Maps pour entrer à la main la position exacte de la tablette.

Il faudra associer à cette optimisation la prise en compte de la direction de la tablette et pas seulement de sa position GPS, par l'utilisation d'une fonctionnalité de type « boussole ».

Concernant l'aspect de gestion des matériaux, **le problème de la végétation** se posera tôt ou tard : un arbre, basé sur le sol, cache une façade. Nous pensons donc judicieux d'implémenter un type « végétation » parmi les différents types de façades (Façade, Sol, Toit).

Afin de faciliter le travail de collaboration nécessaire au travail au sein d'un groupe de chercheurs, nous pensons qu'il sera avantageux de tirer parti des **possibilités de partage de fichiers** offertes par Android, à savoir un bouton « Partage » permettant d'envoyer l'image et le fichier XML associé vers d'autres applications prenant en charge ce type de fichier comme la messagerie, Dropbox, Google Drive, etc.

Enfin, encore dans l'optique de faciliter l'utilisation de l'application pour des personnes travaillant dans tout type de domaine, et notamment pas l'informatique, un **second fichier d'export**, en plus du fichier XML, pourra être envisagé. Nous avons pensé à une nouvelle image de mêmes dimensions que l'image traitée mais uniquement constituée des surfaces créées par l'utilisateur, chacune avec la couleur définie et ne comportant comme texte que le type de matériau. Le principal problème qui se posera sera lorsque l'image possèdera des zones trop proches entre elles ou trop petites pour contenir le nom du matériau : il faudra pouvoir gérer les éventuelles collisions.

Conclusion

Le projet de création d'une application Android d'extraction de caractéristiques de façades et sols sur site à partir d'images est en bonne voie. Sur les trois mois au cours desquels notre équipe a travaillé sur le projet, nous avons réussi à mettre en place **une application aux bases solides** afin qu'il puisse être repris par d'autres équipes, aussi bien composées de chercheurs que d'étudiants, dans les mois futurs.

Toutes les fonctions de l'application imposées par le cahier des charges sont **implémentées et fonctionnelles**. Il est donc possible de charger une image depuis la mémoire de l'appareil ou de prendre une photo ; de réaliser l'extraction des caractéristiques sur les façades et sols qui la composent : type de matériau, couleur, nombre d'étages de bâtiment, nombre de balcons et leur hauteur estimée, mais aussi le calcul automatique du pourcentage de vitrage dans l'image ; d'afficher ces informations de manière très visuelle sur l'image ; d'enregistrer ces données dans un fichier extérieur ; de rouvrir une image déjà traitée avec fusion des nouvelles informations ajoutées ; et enfin, l'obtention de la position GPS de la tablette lorsqu'une image est prise en photo.

Bien entendu, la durée de temps limitée consacrée au projet ne nous a pas permis de finaliser l'application : nous n'avons pu implémenter l'export sous forme plus visuelle qu'un fichier XML, comme nous le craignions déjà lors de la rédaction du cahier des charges. De plus, de **nombreuses améliorations du programme sont envisageables** afin de poursuivre ce projet dans le futur.

Concernant le programme développé, nous avons apporté le plus grand soin à ce qu'il soit **distribuable sous licence CeCILL** et l'ensemble de la javadoc est disponible en anglais.

On voit donc que, bien qu'un travail important ait été effectué, il reste un nombre élevé de fonctionnalités qui seront nécessaires à implémenter afin d'obtenir l'application tant désirée par l'équipe de chercheurs en thermique. Néanmoins, les éléments décidés en début de projet sont bel et bien présents et opérationnels, ce qui nous mène à considérer ce travail comme une réussite.

Bibliographie

Réalisation de l'interface graphique

(1) **Android Design.** Guide de conception d'interfaces graphique pour les applications Android.
(2) **Android Design – Action Bar.** Guide du fonctionnement de l'action bar.
(3) **Android Design – Dialogs.** Conseils d'utilisation des boites de dialogue.
(4) **Android Color Picker.** Boite de dialogue de choix de couleur.
(5) **Android Seekbar preference.** Code permettant l'affichage d'une barre de choix d'un entier.

Développement de l'application

(6) **Site du zéro.** Créez des applications pour Android (Par Apollidore).
(7) **Android developers guide – Fragments.** Rôle et utilisation des fragments.
(8) **Application Ombre.** Code source de l'application Ombre.
(9) **Dimitar Darazhanski's blog.** How to get Picasa images using the Image Picker on Android devices running any OS version.
(10) **Android developers guide.** Location Strategies.
(11) **Android developers guide.** ExifInterface.
(12) **Wikipedia.** Conversion from DMS to Decimal Degree.
(13) **Android developers guide.** Displaying Bitmaps Efficiently.
(14) **Android developers guide.** Canvas and Drawables.
(15) **Android developers guide.** Handling Runtime Changes.
(16) **Darel Rex Finley.** Ultra-Easy Polygon Area Algorithm With C Code Sample.
(17) **Darel Rex Finley.** Determining Whether A Point Is Inside A Complex Polygon.
(18) **Site officiel XStream**.

Gestion des licences de l'application

(19) **Open Source Initiative**. The BSD 3-Clause License.
(20) **Licence CeCILL.** Mettre mon logiciel sous une licence de la famille CeCILL.
(21) **GNU Website.** Licences de logiciel libre compatibles avec la GPL

Annexes

Annexe 1 – Cahier des charges

1. Présentation du projet

1.1. Préambule

Le présent document traite des spécifications fonctionnelles et techniques concernant la réalisation du projet : « Application Android d'extraction de caractéristiques de façades et de sols sur site à partir d'images ». Il est élaboré suite à la réunion du 14 janvier entre l'équipe projet et l'équipe pédagogique.

Ce document a pour but de lister l'ensemble du travail qui sera réalisé. Il permet de traduire l'accord entre les étudiants du projet et l'équipe pédagogique. Une description claire des objectifs va permettre de dégager les besoins du projet et de spécifier ses lignes directrices et limites. Il en découlera le dispositif de conception et les fonctionnalités du programme.

Ce document doit être validé par les deux parties, à savoir les étudiants et l'équipe pédagogique. Il permet de cadrer les conditions et modalités d'exécution des missions/tâches, de communiquer en amont du projet et de formaliser et ordonner l'expression des besoins et des objectifs.

1.2. Contexte

Dans le cadre de l'option Informatique de troisième année, les étudiants sont amenés à réaliser un travail d'application basé sur un projet déterminé par l'équipe pédagogique. Il est question ici du développement d'une application Android permettant l'extraction de données sur les façades et les sols à partir d'images.

Ce projet a lieu suite à l'apparition d'un besoin auprès de chercheurs dans le domaine de la thermique. En effet, les simulations microclimatiques nécessitent de leur part le renseignement d'un nombre important d'informations et de caractéristiques sur les modèles utilisés et, bien qu'ils ne possèdent pas nécessairement ces renseignements, ces derniers peuvent être acquis sur site et intégrés comme annotations ou informations sur des images géo-localisées.

1.3. Fiche signalétique de contact

Étudiants

Patrick Rannou	Patrick.Rannou@eleves.ec-nantes.fr	06.06.55.57.69
Jonathan Cozzo	jonathancozzo@gmail.com	06.89.52.01.65

Équipe pédagogique

Vincent Tourre	Encadrant projet	vincent.tourre@ec-nantes.fr
Myriam Servières	Encadrant projet	myriam.servieres@ec-nantes.fr

2. Environnement

2.1. License et technologie

L'ensemble du programme sera écrit en langage Java sous environnement Android 4.1.1. Le code sera écrit et commenté entièrement en langue anglaise à des fins de partage.

Le programme final devra être sous licence CeCILL.

2.2. Environnement de développement

Le code du projet sera stocké sur un dépôt SVN mis à disposition par l'équipe enseignante. Chaque membre de l'équipe projet possèdera un accès à ce dépôt.

2.3. Méthode de suivi / Déroulement du projet

Le suivi du projet sera réalisé au moyen de trois outils principaux :

- Un rapport d'avancement hebdomadaire, remis tous les vendredis avant 20h à Vincent Tourre et Myriam Servières ;
- Un suivi (tableau Excel) des temps passés sur chaque tâche ;
- Des réunions, en fonction de l'avancement du projet.

Tous ces éléments seront présents sur le SVN fourni par l'équipe pédagogique.

2.4. Livrables

Les livrables sont constitués des éléments suivants :

- Les rapports d'avancement hebdomadaires ;
- Le rapport final ;
- Les sources du programme ;
- Les manuels d'utilisateur et de développeur.

3. Spécifications du programme

3.1. Organisation générale du programme

Le but du programme est la caractérisation au maximum de façades et sols présents sur des images acquises sur site par l'intermédiaire d'un outil mobile (tablette et/ou téléphone Android). Dans la version que l'équipe projet réalisera, cette caractérisation comprendra les types de matériaux, leurs couleurs, le pourcentage de vitrage, le nombre d'étages du bâtiment ainsi que la position suivant l'axe vertical, la longueur et le nombre des balcons présents.

Cette application doit tourner sur tablette Android, fonctionnant avec la dernière version du système d'exploitation (Android 4.2).

L'application à développer présente donc trois grands aspects :

- La récupération de la position exacte de la tablette ;
- L'extraction de caractéristiques dans l'image ;
- L'interface graphique et la navigation tactile du programme.

Ci-dessous sont présentées en détails ces fonctionnalités :

3.1.1. Position GPS

La connaissance de la position exacte (coordonnées GPS) de l'endroit où l'image a été prise est nécessaire puisqu'elle détermine à elle seule les éléments composants l'image. Il sera donc important d'en déterminer la valeur la plus proche possible de la réalité (avec une imprécision moins importante que les 10 à 15 mètres inhérents à la technologie GPS).

On ne pourra donc pas s'appuyer uniquement sur les informations fournies par la puce GPS intégrée à la tablette. Pour résoudre ce problème, il sera possible de reprendre l'existant et notamment le travail réalisé dans le cadre du projet Sig-Ar, dirigé par Myriam Servières et qui utilisait une carte tirée de Google Maps pour en entrer à la main la position.

Cette fonctionnalité constituant une optimisation de la solution, elle sera donc réalisée en fonction de l'état d'avancement du projet et si le temps le permet.

Dans le cas où le temps manquerait, la position de l'image sera celle donnée par la position de la puce GPS embarquée dans la tablette.

3.1.2. Extraction des caractéristiques

A terme, l'application devra permettre l'extraction automatique des caractéristiques d'un flux vidéo (hors du projet actuel).

Comme point de départ, il est attendu que la version développée par l'équipe étudiante assure l'extraction manuelle d'informations d'une image source. Cette opération devant être précise, l'utilisation d'un stylet pourra apporter de meilleurs résultats mais ne sera néanmoins pas indispensable.

3.1.3. Interface graphique et navigation tactile

Les différentes caractéristiques extraites d'une image devront être présentées à l'utilisateur de manière claire, par exemple sous forme de listes des matériaux composants l'image.

Plus précisément, une fois que l'utilisateur a pris à une photo, le programme permet à l'utilisateur, grâce au code « Façade » de délimiter les différents éléments de l'image (façade, sol, porte, fenêtre, etc.) suivant une méthode de ciseaux intelligents ou par la définition des quatre points constituants les coins de l'élément. Il est ensuite offert à l'utilisateur de grouper, dégrouper et fusionner plusieurs éléments délimités.

Lors d'un clic sur une zone particulière de l'image, le programme affichera une liste des matériaux possibles pour les différents éléments constituant cette zone. Cette liste sera modulée suivant la nature de la zone (façade ou sol) ; cette dernière sera déterminée soit par le programme lorsqu'il en sera capable soit par l'utilisateur (un message lui demandant si la zone correspond à un sol ou une façade pourra apparaître). On privilégiera néanmoins l'interaction avec l'utilisateur sans rechercher l'automatisation complète du programme.

L'utilisateur devra alors sélectionner les matériaux qu'il désire ; ses choix seront alors enregistrés par le programme et les autres éléments de la liste disparaîtront (Cf. Figure 1).

En parallèle, un nouveau fichier récapitulant les informations de l'image (types de matériaux, couleurs, etc.) est créé. La mise en forme de ces éléments sera déterminée lors de la phase de développement et dépendra notamment du fonctionnement des méthodes d'extraction d'image. On peut par exemple imaginer la création d'un fichier XML pouvant ensuite être traité par une application desktop extérieure au projet permettant d'utiliser les informations recueillies sur l'image.

D'autre part, un second fichier, orienté utilisateur, rapportant ces mêmes informations mais de manière plus simple pour le lecteur, sera, si le temps le permet, créé à des fins d'exploitation directe et aisée par tous. Le format de ce second fichier reste à déterminer, mais une première solution possible est la création d'une image JPG comprenant l'image originale et les listes de matériaux et caractéristiques en annotation. Le développement de ce second fichier est optionnel du fait des contraintes de temps accordées pour le développement.

De plus, afin de permettre la reprise d'une image déjà traitée, l'équipe projet développera une solution de gestion de réouverture de document. La réouverture d'une image ayant déjà était traitée entraînera la réouverture du fichier XML associé (afin d'en récupérer la position GPS) et les ajouts faits par l'utilisateur sur l'image seront fusionnés avec les informations déjà enregistrées dans le XML.

De manière générale, l'interface de l'application devra être épurée et aisée de prise en main afin de permettre à tout utilisateur un emploi facile et rapide.

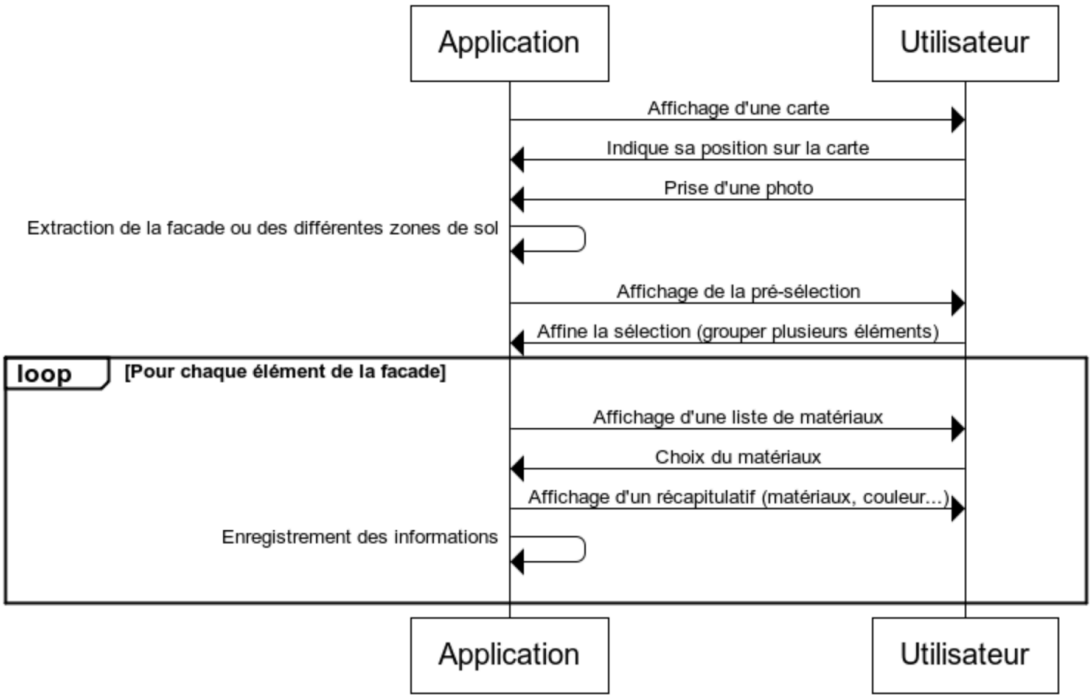


FIGURE 1 : DIAGRAMME DE SEQUENCE

3.2. Documentation

L'ensemble du code sera commenté afin d'alimenter la Javadoc. De plus, un document détaillant les fonctionnalités et l'architecture du programme sera rédigé en parallèle au développement. Des diagrammes UML seront réalisés afin de détailler le travail de conception du programme.

4. Réalisation

La réalisation d'une application Android suppose l'accès au matériel nécessaire par l'équipe projet. Ainsi, une tablette Android sera fournie par l'équipe pédagogique à l'équipe étudiante pour toute la durée du projet.

De plus, l'équipe pédagogique s'engage à fournir à l'équipe projet le code de l'application « Façade », pleinement opérationnelle et dont le fonctionnement sur la tablette prêtée sera vérifié en début de projet. Il n'est pas du ressort de l'équipe projet de devoir assurer la maintenance de ce programme pour le faire fonctionner sur les appareils Android, mais de celui de l'équipe pédagogique.

5. Récapitulatif

5.1. Fonctionnalités de l'application

L'application FeatureApp présentera donc les fonctionnalités suivantes :

- Position GPS de l'image ;
- Extraction des caractéristiques sur les façades et les sols, à savoir les types de matériaux, leurs couleurs, le pourcentage de vitrage, le nombre d'étages du bâtiment et la position suivant l'axe vertical, la longueur et le nombre des balcons présents ;
- Affichage graphique de ces caractéristiques, par exemple sous forme des listes affichables sur l'image ;
- Enregistrement des données des matériaux des sols et façades dans un fichier ;
- Enregistrement des données dans un second fichier, orienté utilisateur et aisé de lecture (optionnel) ;
- Gestion de réouverture de documents avec fusion des informations.

5.2. Livrables intermédiaires

L'équipe projet présentera :

- Une version alpha de l'application, aux alentours du 20 février (à définir suivant les disponibilités des membres des équipes projet et pédagogique), avec l'intégration des codes extérieurs et les fonctionnalités développées à cette date ;
- Une version beta de l'application, aux environs du 10 mars (suivant les disponibilités des membres des équipes projet et pédagogique), présentant l'ensemble des fonctionnalités de l'application développées à cette date.

5.3. Livrables finaux

Les livrables de la fin du projet sont :

- Le rapport final ;
- Les sources du programme ;
- Les manuels d'utilisateur et de développeur.

Ces éléments seront remis à la date du vendredi 22 mars.

6. Planning

Le projet est décomposé en 5 grandes étapes :

1. L'étude préalable comprenant :
    a. La découverte du développement sur plateforme Android ;
    b. La familiarisation avec le code existant (extraction de façade);
    c. Les recherches documentaires sur la manipulation d'image et l'extraction de façade ;
2. Spécifications détaillées
3. Le développement :
    a. Intégration du code de la partie « localisation » du projet Sig-Ar et du projet Façade ;
    b. Programmation de l'application ;
4. Les tests unitaires ;
5. La rédaction :
    a. Du manuel utilisateur ;
    b. Du manuel développeur ;
    c. Du rapport du projet.

Un planning indicatif est disponible ci-dessous :

Nom	Durée	Début	Fin
Etude préalable	15 jours	**17/01/13...**	**06/02/13...**
Découverte du développement sur Android	8 jours	17/01/13 ...	28/01/13 ...
Familiarisation avec le code existant	15 jours	17/01/13 ...	06/02/13 ...
Recherche documentaires	15 jours	17/01/13 ...	06/02/13 ...
Spécifications détaillées	10 jours	17/01/13 ...	30/01/13 ...
Développement	30 jours	**07/02/13...**	**20/03/13...**
Intégration du code Sig-Ar et Facade	15 jours	07/02/13 ...	27/02/13 ...
Programmation de l'application	25 jours	14/02/13 ...	20/03/13 ...
Tests unitaires	7 jours	14/03/13 ...	22/03/13 ...
Rédaction	21 jours	**22/02/13...**	**22/03/13...**
Manuel utilisateur	7 jours	14/03/13 ...	22/03/13 ...
Manuel développeur	7 jours	14/03/13 ...	22/03/13 ...
Rapport	21 jours	22/02/13 ...	22/03/13 ...

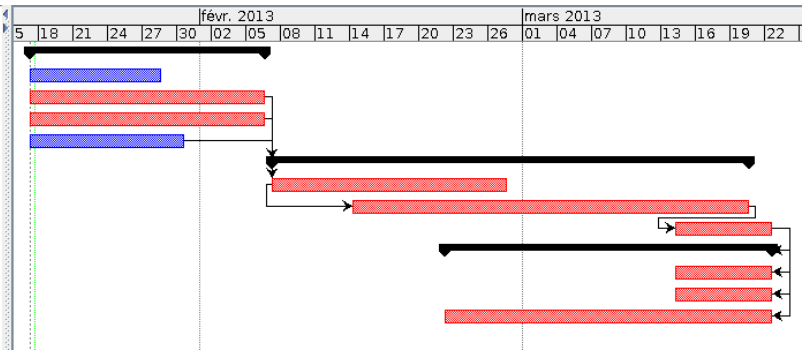


FIGURE 2 : PLANNING INDICATIF

Annexe 2 – Récapitulatif du temps de développement

Tâche		Durée
Pré-projet		**22h**
	Rédaction du cahier des charges	**6h**
	Découverte du développement Android	**5h**
	Familiarisation avec les applications Façade et Ombre	**3h**
	Conception et premières recherches	**8h**
Développement		**75h**
	Structure de l'application	**13h**
	Gestion de la prise des photos et de la position GPS	**8h**
	Gestion des façades	**16h**
	Gestion des caractéristiques	**15h**
	Affichage des caractéristiques	**5h**
	Import et export XML	**8h**
	Améliorations et correction du code, ajout de commentaires	**10h**
Rédaction de la documentation		**16h**
	Rédaction du guide utilisateur	**1h**
	Rédaction du guide développeur	**1h**
	Rédaction du rapport	**14h**
Management de projet		**13h**
	Réunions	**10h**
	Rapports d'avancement et mise en commun hebdomadaire	**3h**
Total		**126 h**

Il est important de noter que les durées de certains développements ont pu être allongées dû à notre inexpérience dans le développement d'applications Android.

Annexe 3 – Guide du développeur

1. Licence de l'application

Cette application Android est **distribuée sous la licence libre CeCILL** (Plus d'informations : http://www.cecill.info/). Il s'agit d'une licence libre compatible avec la licence GNU GPL et adaptée au droit français.

Vous pouvez donc librement utiliser, modifier et redistribuer cette application.

Le code de cette application **utilise certaines librairies externes** ainsi que des extraits de code :

- XStream (http://xstream.codehaus.org/)[16]
  - Distribuée sous la licence BSD (Berkeley Software Distribution license)
  - Utilisée pour l'enregistrement et l'ouverture des fichiers XML
- Android Color Picker (http://code.google.com/p/android-color-picker/)
  - Distribuée sous la licence Apache 2.0.
  - Utilisée pour l'affichage de la boite de dialogue de choix des couleurs
- Android seekbar preference (http://robobunny.com/wp/2011/08/13/android-seekbar-preference/)
  - Utilisé pour afficher le choix de la précision dans les paramètres
- Extraits du code de l'application Ombre (https://github.com/CERMA-ECN/Ombre)
  - Distribuée sous la licence CeCILL
  - En partie utilisée pour l'affichage des zones et la récupération de photos depuis la tablette (avec des modifications substantielles)
- Extraits de code du guide du développeur Android (Android Developer : http://developer.android.com/develop/index.html)
  - Utilisé pour le redimensionnement des photos avant leur affichage.

2. Compilation

Cette application a été développée et compilée avec la version **la version 17 de l'API Android** (Android 4.2). Cependant elle est compatible avec les versions d'Android supérieures à la version Android 3.2 (API 13).

Il n'y pas d'actions spécifiques pour effectuer la compilation, la librairie externe présente dans le dossier lib étant (normalement) automatiquement ajoutée au projet.

3. Modification du code

Les différentes classes de l'application ont été réparties dans **plusieurs paquets** pour une meilleure lisibilité et compréhension :

- *Activities :* Contient toutes les activités de l'application ;
- *Dialogs :* Contient l'implémentation de toutes les boites de dialogue ;
- *Fragments :* Regroupe les fragments de l'application ;
- *Utils :* Regroupe certaines fonctions utilitaires (pour afficher l'image, trouver la position GPS,…) ;

[16] Une version d'XStream spécialement adaptée à une utilisation sous Android a été utilisée (http://www.java2s.com/Code/Jar/x/Downloadxstreamandroidjar.htm)

- *Utils.Colorpicker :* Contient les fonctions permettant l'affichage de la boite de dialogue de choix des couleurs ;
- *Zones :* Contient les fonctions permettant la mémorisation des informations entrées par l'utilisateur (zones, ensembles de zones, balcons…).

Pour une meilleure compréhension du code, nous vous invitons à consulter **le rapport ainsi que les nombreux commentaires et la Javadoc** présente dans le code de l'application.

4. Modification des matériaux proposés par défaut

La **liste des matériaux par défaut peut être configurée** en modifiant le fichier XML présent dans le dossier : /res/values/material.xml

Il suffit d'ajouter le matériau dans la liste correspondant à son type. Les lignes contenant « @string/other » sont celles qui permettent à l'utilisateur d'entrer un matériau non présent dans la liste. Elle peut être supprimée si l'on souhaite retirer cette fonctionnalité.

5. Internationalisation de l'application

Il est possible **d'internationaliser facilement cette application**. Pour cela, il faut créer un dossier spécifique à la langue dans le dossier « res » en suivant les notations à deux lettres du code ISO 639-1 (Plus d'informations : http://www.loc.gov/standards/iso639-2/php/code_list.php). Par exemple, un dossier « values-fr » pour le français et un dossier « values-en » pour l'anglais. Il faut ensuite copier les fichiers « strings.xml » et « material.xml » présents dans le dossier « /res/values » et les traduire.

La langue de l'application sera alors **la langue configurée sur l'appareil**. Si celle-ci n'est pas disponible dans l'application, la langue par défaut sera celle présente dans le dossier « values ».

Annexe 4 – Guide d'utilisation

1. Lancement de l'application

Au lancement de l'application, un écran de choix apparaît. L'utilisateur doit décider s'il veut « Charger une image » ou « Prendre une photo ». Les deux boutons correspondant sont la seule chose à l'écran.

1.1. Charger une image

L'appui sur le bouton « Charger une image » ouvre la galerie d'images de la tablette et permet à l'utilisateur d'effectuer son choix. Une fois l'image sélectionnée, il est amené à la page principale de l'application.

1.2. Prendre une photo

L'appui sur le bouton « Prendre une photo » lance l'application « Camera » interne à la tablette. Lorsque l'utilisateur a pris une photo, il a le choix entre « Enregistrer » et « Annuler ». Le premier choix l'amène à la page principale de l'application ; le second va lui permettre de reprendre une photo.

1.3. Retour en arrière

Dans un cas comme dans l'autre, il peut cliquer sur la flèche de retour, située en bas à gauche de l'écran, pour revenir au choix initial de la prise de photo ou chargement d'image.

2. Traitement d'image

2.1. Aperçu de l'interface

2.1.1. Aspect général

L'interface globale est très proche des recommandations fournies par le guide de design d'Android. Ce choix a été fait, entre autre, dans le but de faciliter l'utilisation de l'application en conservant une certaine homogénéité avec les applications déjà connues par l'utilisateur de la tablette.

2.1.2. Action bar

La pierre angulaire de l'utilisation de FeatureApp réside dans le bandeau horizontal situé en haut de l'écran, couramment appelé « action bar ».

IMAGE 0.1 - ACTION BAR DE L'APPLICATION

Ce bandeau est composé de trois onglets, régissant les trois grands modes de fonctionnement de l'application : la sélection de façade, le choix des caractéristiques et le récapitulatif. On sait à tout moment dans quel mode on se trouve grâce au petit bandeau bleu présent sous le nom de l'onglet actif.

Sur le côté droit de ce bandeau, d'autres boutons sont présents ; ils ne proposent pas les mêmes options suivant quel mode actif. Cependant, dans les trois cas, il y a toujours l'option « Enregistrer » qui permet de sauvegarder son travail, à tout moment, ainsi qu'un menu déroulant (dont le contenu, lui, dépend du mode actif).

Il est possible de passer d'un mode actif à l'autre à tout moment par simple appui sur le bouton correspondant. Néanmoins, l'ordre logique est celui suivi lors de la lecture (de gauche à droite).

2.2. Sélection des zones

2.2.1. Ajouter une zone

Ce mode permet de dessiner des zones, correspondant à des façades, que l'on pourra par la suite caractériser.

Afin de pouvoir dessiner une zone, il est nécessaire d'appuyer sur le bouton « Ajouter une zone » situé en haut à droite de l'écran. Dès lors, on va pouvoir cliquer sur l'image, sur les sommets de la façade que l'on souhaite définir.

Tant que la façade n'est pas fermée, les traits sont rouges ; une fois que l'on est revenu sur le premier sommet qui a été défini, le contour devient vert.

IMAGE 0.2 - AJOUT DE FAÇADES

2.2.2. Actions du bandeau déroulant

En haut à droite de l'écran se trouve un bandeau déroulant. Lorsque le mode actif est la sélection de zone, ce bandeau donne différentes possibilités :

Supprimer une zone

Après avoir appuyé sur « Supprimer une zone », l'utilisateur peut alors appuyer sur la zone qu'il souhaite faire disparaître. Un message de confirmation s'affiche et l'utilisateur peut continuer ou annuler suivant son choix.

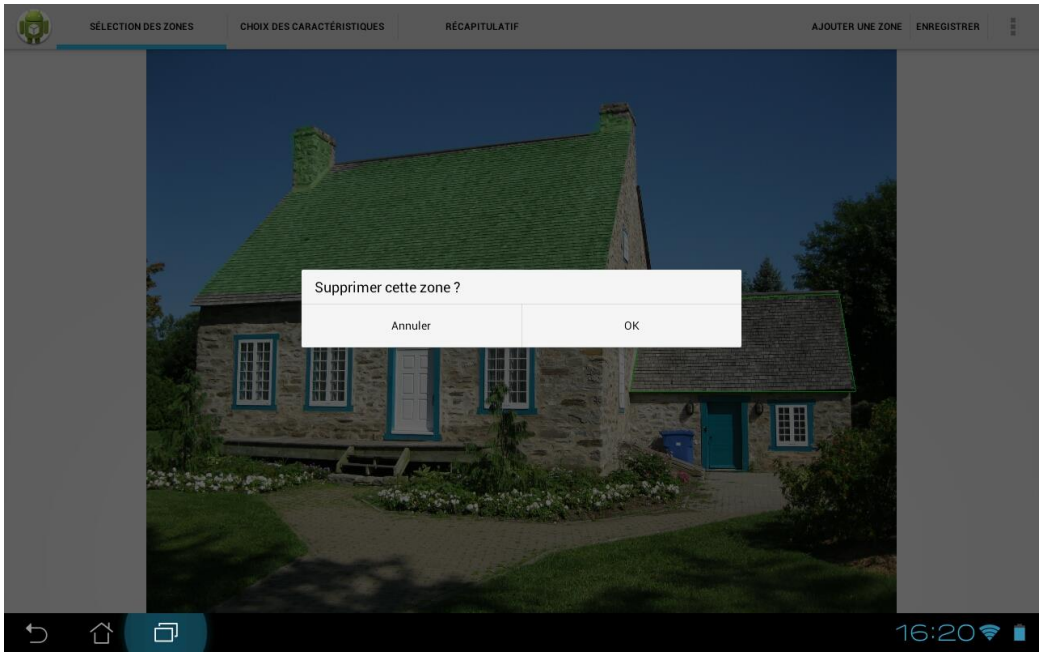


IMAGE 0.3 - SUPPRESSION DE ZONE

Regrouper des zones
Lorsque plusieurs zones ont été tracées, l'utilisateur a la possibilité de les regrouper. En effet, il est possible qu'il se trouve dans une situation où un même mur est séparé en deux un vitrage et qu'il veuille donc que le programme considère ces deux pans de mur comme un seul élément.

Après avoir appuyé sur le bouton « Regrouper des zones », l'utilisateur peut alors appuyer sur toutes les zones qu'il souhaite regrouper. Une fois une zone sélectionnée, elle est marquée de la couleur verte (cf. image précédente 2.2 – Ajout de façade). Lorsqu'il a sélectionné toutes les zones qu'il souhaitait, l'utilisateur appui sur le bouton « Regrouper les zones sélectionnées » en haut à droite de l'écran. Un message l'informant de la réussite de l'opération apparaît alors en bas de l'écran. Si l'utilisateur désire annuler son opération, le bouton « Annuler » est disponible en haut à droite également.

Séparer des zones
Lorsque plusieurs zones sont groupées, il est possible que l'utilisateur souhaite les séparer pour pouvoir les traiter séparément. Il peut alors appuyer sur le bouton « Séparer des zones » puis sélectionner l'une des zones regroupées. Toutes les zones qui appartenaient à ce groupe deviennent alors indépendantes et un message de réussite apparaît à l'écran.

Paramètres
Cette option est disponible dans tous les modes de fonctionnement de l'application. Il permet de choisir la précision du tracé d'une zone. En effet, il est probable que l'utilisateur ne recherche pas la même précision de tracé suivant qu'il utilise son doigt ou le stylet de la tablette pour réaliser sa sélection de zone.

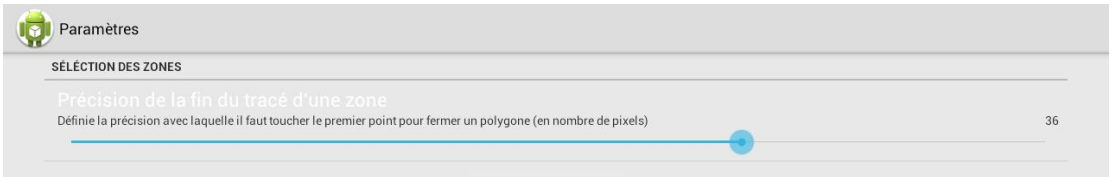


IMAGE 0.4 - PARAMETRES

2.3. Choix des caractéristiques

Une fois une zone tracée, l'utilisateur peut lui donner des caractéristiques.

2.3.1. Caractériser une zone

En cliquant sur le bouton « Caractériser une zone », situé en haut à droite de l'écran, l'utilisateur peut ensuite sélectionner une zone qu'il a préalablement définie.

A la suite de cet appui, une boite de dialogue apparaît, demandant à l'utilisateur de choisir le type de zone : façade, sol ou toit. Après cela, un type de matériau est demandé. L'utilisateur peut choisir un matériau présent dans la liste (dépendant du type de zone sélectionné précédemment) ou bien le taper à la main en sélection « Autre ».

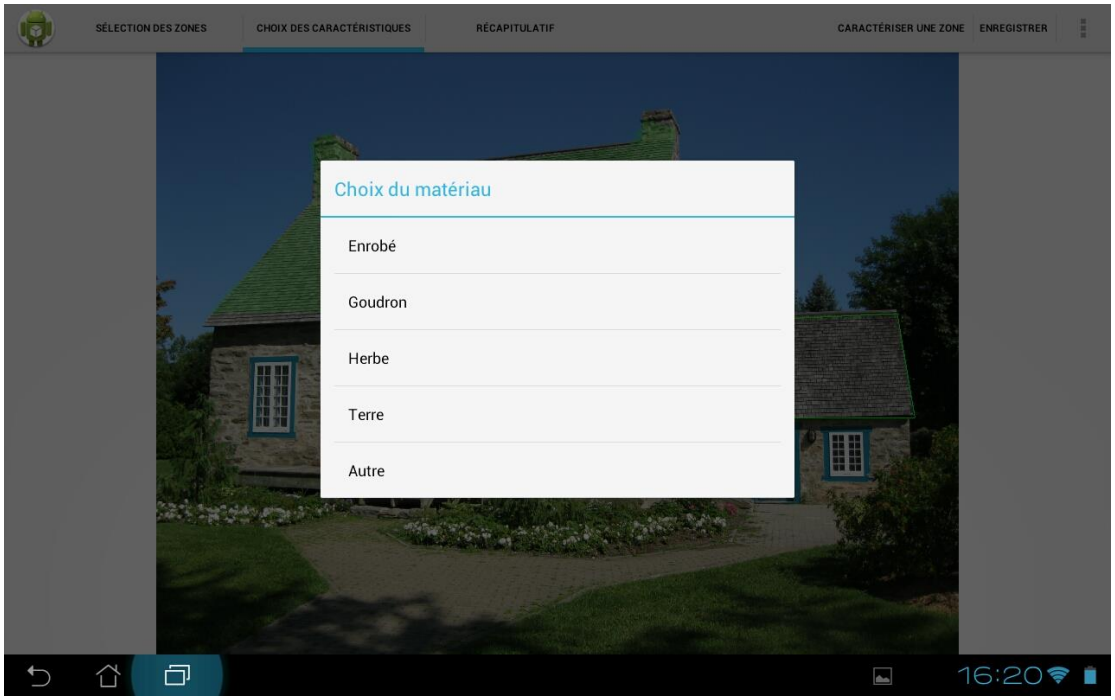


IMAGE 0.5 - CHOIX DU TYPE DE MATERIAU

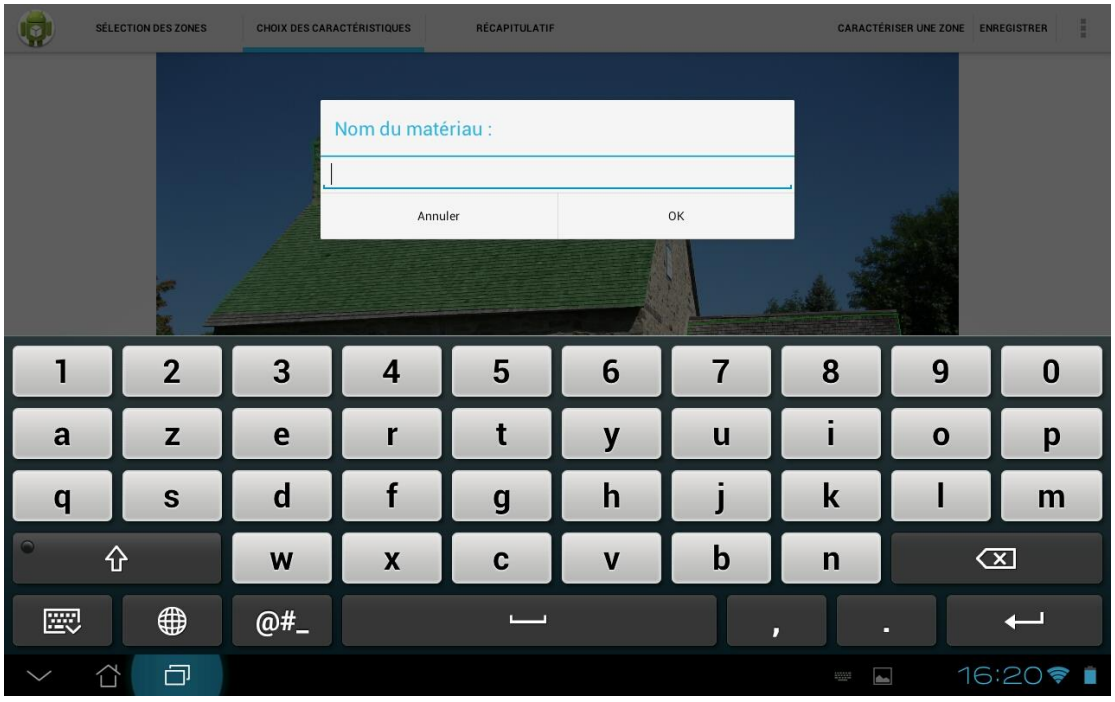


IMAGE 0.6 - ENTREE A LA MAIN DU TYPE DE MATERIAU

2.3.2. Actions du bandeau déroulant

Le bandeau déroulant situé en haut à droite offre ici encore différentes possibilités de caractérisation de façade :

Définir la couleur
Après avoir appuyé sur « Définir la couleur », l'utilisateur peut alors appuyer sur une zone et une boite de dialogue apparaît, dans laquelle il pourra choisir la couleur de la façade.

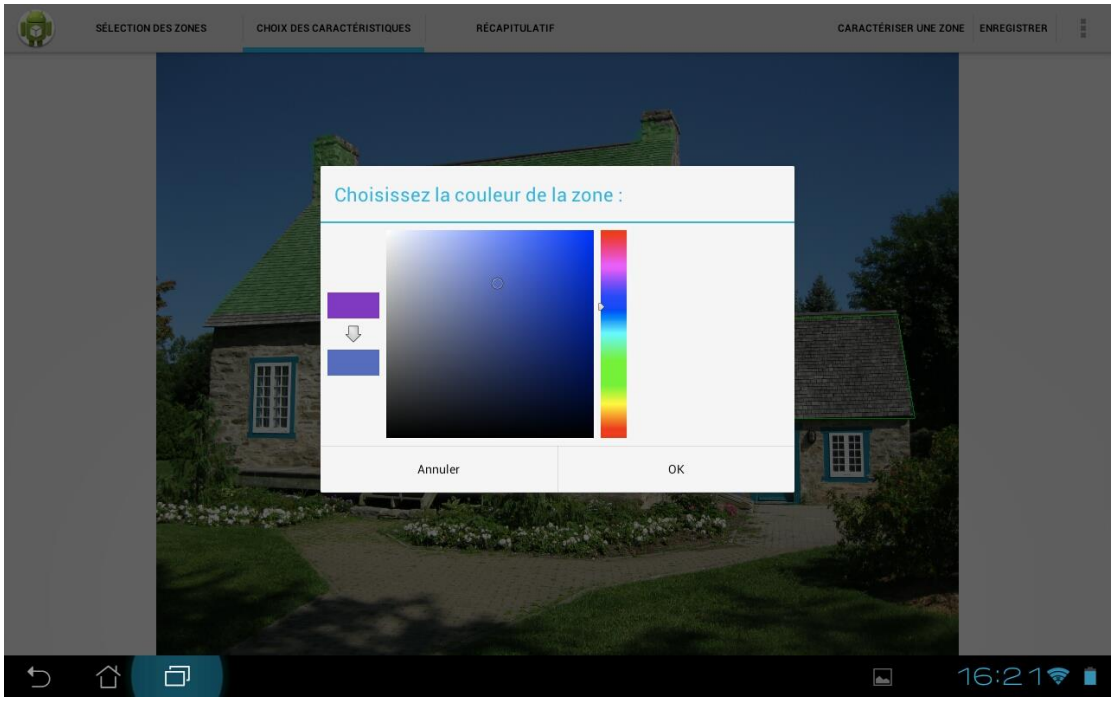


IMAGE 0.7 - DEFINIR LA COULEUR

Indiquer un balcon
De la même manière, on peut choisir d'indiquer qu'une façade particulière est un balcon par simple clic sur cette façade.

Une fois le clic effectué, une nouvelle boite de dialogue apparaît demandant à l'utilisateur d'estimer la hauteur du balcon par rapport au sol (en cm).

Ajouter le nombre d'étages

Par le clic sur « Ajouter le nombre d'étages », une autre boite de dialogue apparaît proposant à l'utilisateur de définir le nombre d'étages du bâtiment situé sur la photo.

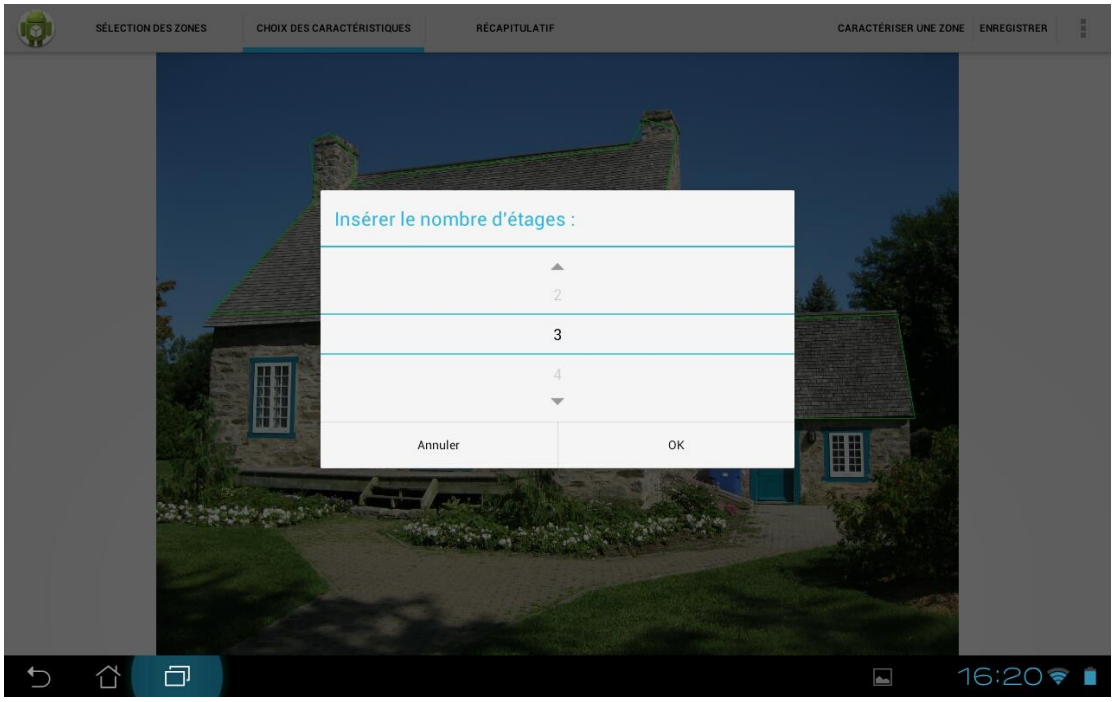


IMAGE 0.8 - AJOUTER UN NOMBRE D'ETAGES

2.4.Récapitulatif

2.4.1. Afficher les informations relatives à une zone

Lorsque le mode « Récapitulatif » est actif, l'appui sur n'importe quelle zone définie sur l'image va faire apparaître une boite de dialogue rapportant toutes les informations qui ont été annotées à cette façade (type, matériau, couleur, etc.).

Afin de revenir à l'affichage classique de l'image, il faut cliquer sur la flèche « retour arrière » située en bas à gauche de l'écran.

2.4.2. Afficher les informations globales

Par l'appui sur le bouton « Afficher les informations globales », une autre boite de dialogue apparaît. Elle rapporte les informations générales à l'image : le nombre d'étages, le pourcentage de vitrage et le nombre de balcons.

L'appui sur la flèche « retour arrière » en bas à gauche de l'écran ramène à l'affichage de l'image.

3. Choisir une nouvelle image

Il est possible de changer d'image de travail sans avoir à quitter et relancer l'application. Pour cela, il suffit d'appuyer sur la flèche de « retour arrière » lorsque l'on se trouve dans l'un des trois modes de fonctionnement de l'application pour revenir à l'écran d'accueil.

Avant que le retour à ce premier écran ne soit effectif, un message d'avertissement apparaît. Il précise que toute donnée non sauvegardée sera perdue. L'utilisateur peut alors décider de poursuivre ou non.

Annexe 5 – Diagrammes de classe

Le diagramme de classe général a été subdivisé en 3 diagrammes pour améliorer la lisibilité :

- Présentation des classes concernant les activités et les fragments
- Description des classes implémentant les boites de dialogue
- Classes permettant la gestion de des zones

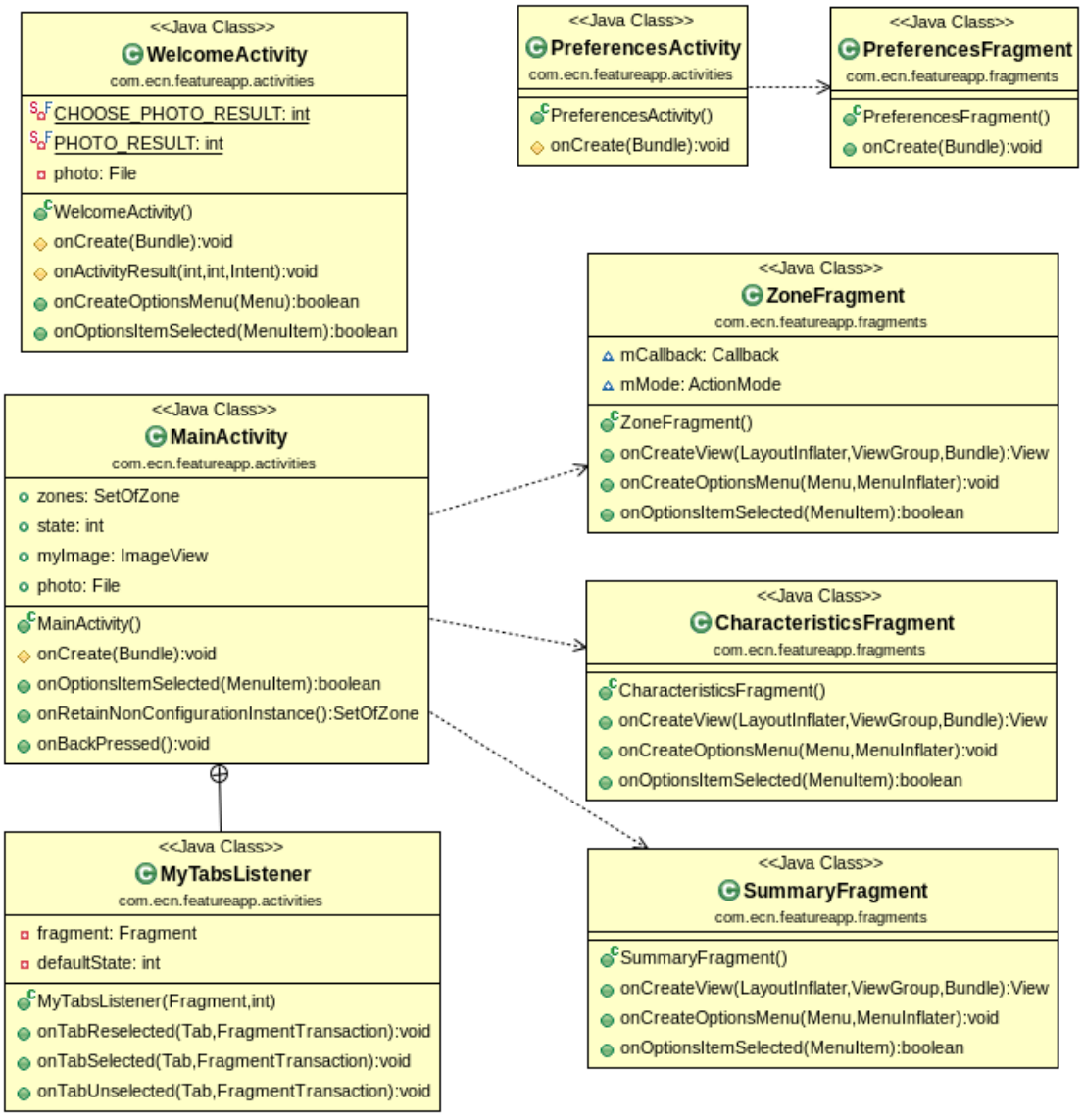


DIAGRAMME 3 - DIAGRAMME DE CLASSE (ACTIVITES ET FRAGMENTS)

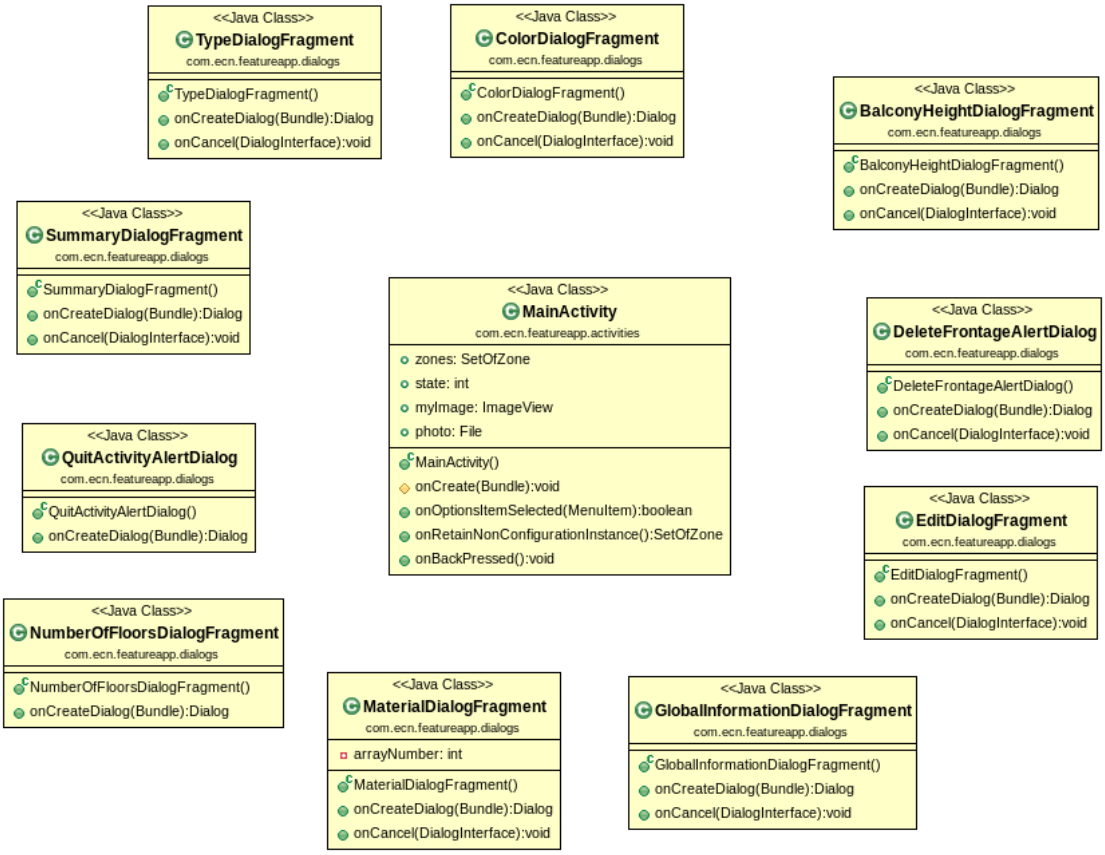


DIAGRAMME 4 - DIAGRAMME DE CLASSE (BOITES DE DIALOGUE)

DIAGRAMME 5 - DIAGRAMME DE CLASSE (GESTION DES ZONES)