GRAND CERCLE MOBILE - GCM

Document d'implantation

Luiza CICONE - Jérémy KREIN - Jérémy LUQUET - Paul MAYER
ISI - IF

Mai 2012

Table des matières

1 Modèle objet **2**

2 Présentation technologie **2**

3 Validation **3**

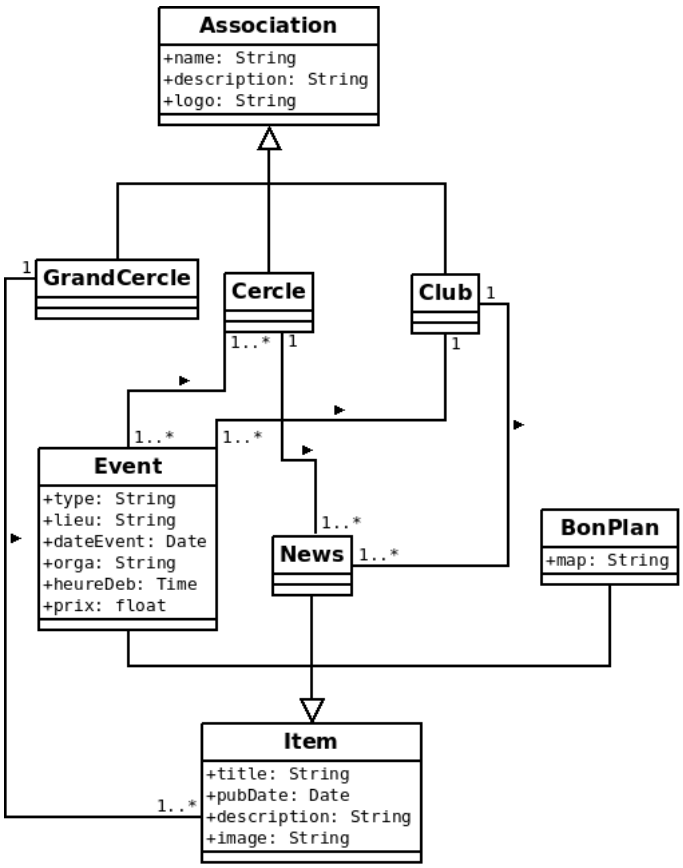
3.1 Tests unitaires 3

3.2 Tests d'intégration et de validation informatique 4

3.3 Tests utilisateurs, tests de conformité 4

1 Modèle objet

Le diagramme suivant représente la modélisation objet du domaine d'application qui guide nos choix de conception.



2 Présentation technologie

La technologie android repose sur la dualité entre la langage Java et le langage xml. A chaque affichage correspond un fichier xml qui met en place les différentes fenêtres qui constituent l'écran, appelées des « layouts », ainsi qu'un fichier Java qui représente ce qu'on appelle une activité, c'est à dire la fenêtre visible à l'écran. La vue offerte à l'utilisateur est mise en place par la méthode setContentView([fichier].xml). Néanmoins, ce fichier xml en paramètre de la méthode mentionnée correspond à une configuration statique de l'affichage. Ainsi, dès que nous avons besoin de modifier dynamiquement une vue, nous le faisons dans le code Java, grâce à l'identifiant de la vue considérée renseigné dans le code xml : c'est ce qui fait la dualité Java/xml. Néanmoins, le placement des différents objets dans les layouts est moins facile en Java que dans un fichier xml, c'est pourquoi nous sommes passé par les fichiers xml dès que cela était possible.

Dans la technologie android, c'est un fichier appelé AndroidManifest.xml qui gère les différentes activités de l'application. La première activité lancée est celle contenue dans un label « intent-filter ». Dans notre cas, c'est le fichier GCMLaunching.java qui constitue l'activité de démarrage. Ce dernier fait appel à la méthode `parseFiles(Context context)` , qui utilise une classe factory pour obtenir une instance d'un SAXParser, nécessaire pour la récupération des données. En effet, l'application réalisée ne peut exister sans que le site du Grand Cercle ne soit tenu à jour. Nous récupérons ainsi toutes les données présentes sur ce site à l'aide de cinq

parser, appelés par la méthode `parseFiles(Context context)` sur cinq fichiers xml différents en même temps que l'affichage d'accueil à l'aide de threads.

Plusieurs choses sont ainsi réalisées lors de cette phase :
- récupération des données sur le site
- création des listes de clubs, de cercles et de types d'événements, nécessaire pour la gestion des préférences
- initialisation de la base de données nécessaire pour la gestion des préférences

Un des points importants de la récupération de donnée est la présence ou non de connexion internet :
- soit une connexion internet est disponible. Dans ce cas on sauvegarde les fichiers xml en provenance du site dans la mémoire et on parse ces données.

- soit aucune connexion n'est possible et dans ce cas, on parse les données qui sont dans la mémoire, sauvegardées lors de connexions antérieures.

Lors de la première utilisation de l'application, un message est affiché à l'utilisateur s'il n'est pas connecté à internet pour lui indiquer qu'aucune donnée n'a pu être chargée (mais l'application se lance correctement). Lors de l'initialisation de la base de données, aucun filtre n'est appliqué et les données sont récupérées dans leur totalité, l'utilisateur ayant ensuite le choix d'appliquer ou non ces filtres.

Une fois la récupération des données et les initialisations effectuées, l'activité GCM.java est lancée. La différence avec les autres activités est qu'elle étend la classe TabActivity et non Activity. En effet, cette activité permet de construire les onglets de notre application à l'aide d'un objet appelé `TabHost`. Toutes les vues des différents onglets sont crées dans GCM.java, ce qui permet à l'utilisateur de naviguer rapidement entre les différents onglets. Au sein de ces onglets, un clic sur les différents boutons lancent d'autres activités à l'aide de `listeners`.

Lors de l'initialisation de la base de données, aucun filtre n'est appliqué et les données sont récupérées dans leur totalité, l'utilisateur ayant ensuite le choix d'appliquer ou non ces filtres.

La base de données est utilisée dans la gestion des préférences, pour choisir les différents filtres ainsi que l'apparence ergonomique. Quatre types de préférences sont proposés à l'utilisateur et pour chaque type, les éléments cochés sont sauvegardés dans la base de données. On effectue alors des tests sur ces données dans les autres activités, notamment pour les couleurs de fond et les filtres sur les données récupérées sur le site du Grand Cercle.

3 Validation

Les deux applications ont été validées suivant une démarche commune afin d'assurer un niveau de qualité équivalent d'une application à l'autre.

Certains tests ont été réalisés au fur et à mesure du développement des applications, d'autres ont été effectués sur des prototypes fonctionnels et enfin la version finale des deux application a été évaluée par un panel d'utilisateurs potentiels.

3.1 Tests unitaires

Chaque module a été testé grâce à des tests unitaires. Majoritairement, ces tests consistent à envoyer en entrée du module un cas de test spécifique, de tracer le module afin de vérifier qu'au cours de l'exécution rien d'anormal ne se produit, et de récupérer en sortie un résultat que l'on compare au résultat attendu. Les tests spécifiques sont pour la plupart des tests dits "boite blanche".

Ces tests sont avant tout des tests techniques qui permettent de s'assurer qu'il n'y a pas d'erreur d'analyse et/ou de programmation.

3.2 Tests d'intégration et de validation informatique

Ces tests permettent de tester la cohérence et l'articulation des modules entre eux, de vérifier que les modules communiquent bien entre eux.

Ces tests ont été réalisés de manière similaire aux tests unitaires, à savoir un traçage des opérations effectuées et une comparaison du résultat obtenu avec le résultat attendu.

3.3 Tests utilisateurs, tests de conformité

Afin de vérifier la conformité de nos différents prototypes avec les besoins formulés par les utilisateurs, regroupés dans le cahier des charges, nous avons fait appel à un panel d'utilisateurs potentiels disposant de leur propre smartphone sous iOS ou Android, et ce dans le but de confronter ces applications à des utilisateurs habitués aux pratiques sur ces deux systèmes d'exploitation.

Nous avons mis en place des une procédure qui nous a permis de toujours demander à ces utilisateurs potentiels d'effectuer les mêmes manipulations dans l'application et d'observer leur réactions. Pour prendre en compte leurs remarques et ne pas en oublier, nous avons procédé à des enregistrements vidéo de ces tests.