|  |  |  |  |
| --- | --- | --- | --- |
| **Nom : Bayrakci** | **Prénom : Murat** | **Classe : 3M3I2** | |
| **Semaine N° : -** | **Du : 13.03.2017** | **Au : -** | |
| Genre de travail | | | Durée |
| *Lundi 13.03.2017 : (4 périodes)* | | |  |
| Lecture du cahier des charges | | | 1 |
| Mise en place du Trello, d’une liste de fonctionnalités et définition priorités des tâches | | | 1 |
| Création du Planning (Répartition des tâches) | | | 1 |
| Création du Planning (Mise en place des périodes nécessaires aux tâches et dépendances) | | | 1 |
|  | | |  |
| *Mardi: 14.03.2017 (9 périodes)* | | |  |
| Création du Schéma de la première vue (sur papier) | | | 1 |
| Re-Création du Schéma de la première vue car l’espace était mal utilisé (sur papier) | | | 1 |
| Création du Schéma de la seconde vue (sur papier) | | | 1 |
| Création du protocole de test | | | 1 |
| Fin du planning. Ajout des schémas et du planning dans Trello sous « Documentation » | | | 1 |
|  | | |  |
| Création du schéma réseau | | | 1 |
| Lecture et Prise en main du site Hypercool | | | 1 |
| Tests de différentes requêtes sur le site Hypercool | | | 1 |
| Analyse des données JSON reçues et rédaction de quelques questions à poser au client (choix du client par exemple) | | | 1 |
|  | | |  |
| Lundi : 20.03.2017 (4 périodes) | | |  |
| Changement des libellés des cartes du tableau « liste des fonctionnalités » + création diagramme de flux de l’activité principal + poser les questions au prof | | | 1 |
| Création du layout (simple) de la première activité | | | 1 |
| Création du code pour prendre l’id de la classe, puis utiliser cet id pour faire une seconde requête qui va nous renvoyer l’horaire Brut de la classe | | | 1 |
| Création du code pour placer toutes les informations d’une branche dans un tableau (pas fini) | | | 1 |
|  | | |  |
| Mardi : 21.03.2017 | | |  |
| Relecture du code crée précédemment et simplification de quelques fonctions. Suite de la mise en place de données reçues dans des tableaux. | | | 1 |
| Mise en place des données dans les tableaux finie. Début du tri pour avoir l’horaire chronologique. | | | 1 |
| Tri des tableaux mis en attente. Recherche de documentation pour la création de layouts dynamiques pour afficher les données. | | | 1 |
| Création de plusieurs fichiers xml dans /drawable pour le layout dynamique. | | | 1 |
| Création du layout principal + tests. Aucun résultat positif. Pour ne pas perdre de temps, je décide de passer à la suite. | | | 1 |
| Création du menu pour changer d’activité et création des boutons permettant de changer de jour/semaine. | | | 1 |
| Résolution d’un bug qui faisait crash l’application quand aucunes données n’étaient reçues. Fin des boutons permettant de changer jour/semaine, désormais fonctionnel. | | | 1 |
| Mise en place de la date automatique dès le lancement de l’application | | | 1 |
| Différents tests pour la création du layout dynamique (ajout d’un layout depuis le fichier java, ajout de plusieurs textview avec une boucle…) + décision de changer les arrays en arraylist. | | | 1 |
|  | | |  |
| Lundi 27.03.2017 | | |  |
| Démonstration du programme actuel au professeur. Lors de la vérification de mon code par M. Saccheti, celui-ci m’a donné de nombreux conseils à mettre en place dans le code. | | | 1 |
| Ajouts des sources dans git. Changement dans le code, dorénavant, les données json reçues sont placées dans des arrayslists (array avant). Cela va être plus simple à utiliser par la suite. Mise en place des conseils donnés par M. Sachetti dont : Utilisation du fichier strings.xml, passage des certaines variables de int en bool, création d’une nouvelle classe pour l’asyncstask. | | | 1 |
| Création du use case (sur papier) et suite du sorting des tableaux cette fois ci avec des arraylists. | | | 1 |
| Fin du sorting et ajout de divers try/catch au Jsonobject. Sinon ça faisait crash si rien n’était reçu. | | | 1 |
| Mardi 28.03.2017 | | |  |
| A la suite des recherches effectuées les jours précédents, j’ai pu commencer la création du layout dynamique. Tout d’abord création des fonctions pour créer un textview, un layout horizontal, un layout vertical en fonction du nombre d’entrées reçues. | | | 1 |
| Utilisation des fonctions créees précédemment pour créer le layout. Dorénavant fonctionnel. Ajout d’une scrollview en xml car les données reçues prenaient plus de place que la taille de l’écran. | | | 1 |
| Recherche sur l’utilisation de l’autocompleteTextview et sur l’écriture ligne par ligne dans un fichier texte. Début de la mise en place de la persistance. | | | 1 |
| Création de 3 fonctions : Ecriture texte, Lecture texte et suppression des doublons. Mise en commun des fonctions pour ainsi avoir la persistance fonctionnelle. Affinement de la version démo pour jeudi. | | | 1 |
|  | | |  |
| Jeudi 30.03.2017 | | |  |
| Création de la seconde activité, du layout et mise en place des requêtes avec des dates choisies arbitrairement. | | | 1 |
| Traitement des données JSON reçues pour les mettre dans des arraylists. Le code est pratiquement le même que la première activité, mais cette fois-ci, je devais faire une boucle avec le nombre de jour. | | | 1 |
| Début de la fonction permettant de mettre dans un tableau les dates du lundi à dimanche en fonction de la date qui était dans l’edittext etdate. | | | 1 |
| Fin de la fonction et présentation & et discussion avec l’expert. | | | 1 |
| Implémentation de la fonction DateEnSemaine dans l’onCreate, et modification des requêtes qui utilisent dorénavant les dates de la semaine (fourni par le tableau des dates de la semaine). | | | 1 |
|  | | |  |
| Mardi 18.04.2017 | | |  |
| Changement des id de quelques views dans la seconde activité, car les mêmes noms d’id dans les deux activités créaient des interférences. Création des cases pour les périodes dans le xml. | | | 1 |
| Récupération des views pour les mettre dans des linearlayouts et textview 2 dimensions (avec findviewbyid). Ajout du switch avec comme case les heures de débuts (par exemple 8h10, 8h55…). Selon le cas, on envoie à la fonction afficheSemaine le numéro du layout en question et la date du début. | | | 1 |
| Création de la fonction afficheSemaine qui va s’occuper de traiter les données (nom de la branche, salle…) et les placer dans la période correspondante dans le layout. | | | 1 |
| Amélioration de la fonction afficheSemaine qui va maintenant détecter le nombre de période et ainsi colorier plusieurs cases. | | | 1 |
| Une fois l’affichage relativement fonctionnelle, création du tableau de couleur et de l’algorithme pour attribuer la même couleur aux mêmes branches. | | | 1 |
| Amélioration esthétique : maintenant, si la fonction détecte plusieurs périodes, alors le texte va être centré. Si la branche dure plus d’une période, alors on va également afficher le nom du professeur concerné. | | | 1 |
| Lorsque la branche dure une seule période, pour ne pas perdre trop de place, le codeMatière va être affiché à la place du libellé. Si la branche dure plusieurs périodes, alors on va couper les libelles de plus de 20 caractères et les afficher sur plusieurs lignes. Car sans cela, les libellés trop longs faisaient décaler tout le tableau. | | | 1 |
| Petit problème technique survenu : Hypercool et Horaires.cpln ne répondaient plus du tout et je ne recevais donc plus aucunes données. Je décide de passer à la suite et puis revenir lorsque tout devient à nouveau fonctionnel. Création du menu pour pouvoir changer de vue dans l’activité semaine. | | | 1 |
| Recherche d’informations pour pouvoir utiliser les données JSON lorsque l’on ne connait pas l’id. | | | 1 |
|  | | |  |
| Jeudi 20.04.2017 | | |  |
| Mise en place de l’autocompletetextview qui va afficher toutes les classes reçues. Pour cela j’utilise un iterator, comme ça je peux récupérer l’id des classes (que je ne connais pas). Envoi du mail à M. Ferrari pour savoir s’il était possible de faire une requête nous renvoyant uniquement les ressources de type classe. | | | 1 |
| Changement de la fonction parseId dans le deux activités. Avant pour avoir l’id de la classe, étant donné que je ne connaissais pas le nom du jsonarray, je l’obtenais en faisant un substring. Ce n’était pas très propre comme manière de faire. Maintenant j’ai changé et je travaille avec l’iterator. (Même principe que le point précédent.) | | | 1 |
| Recherche d’information pour pouvoir afficher un calendrier lorsque l’on appuie sur un bouton. Début de la mise en place du datePicker. | | | 1 |
| Fin de la mise en place du datePicker dans les deux activités. Dorénavant fonctionnelle. | | | 1 |
|  | | |  |
| Vendredi 21.04.2017 | | |  |
|  | | | 1 |
|  | | | 1 |
|  | | | 1 |
|  | | | 1 |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
| Visa: | | |  |