CÉGEP STE-FOY – HIVER 2019
PROGRAMMATION MOBILE II – 420-V60-SF

Énoncé du Travail Pratique 3

4 février 2019

Préparé par
Benjamin Lemelin

1 Résumé

Créez une application multiplateforme permettant d'apprendre l'alphabet d'une langue (au choix, mais il est proposé d'utiliser les « Hiraganas »). Deux modes doivent être inclus : apprendre et entrainement infini.

2 Conditions de réalisation

Valeur de la note finale	Type	Durée	Nombre de remises
30 %	En équipe	2 semaines	1

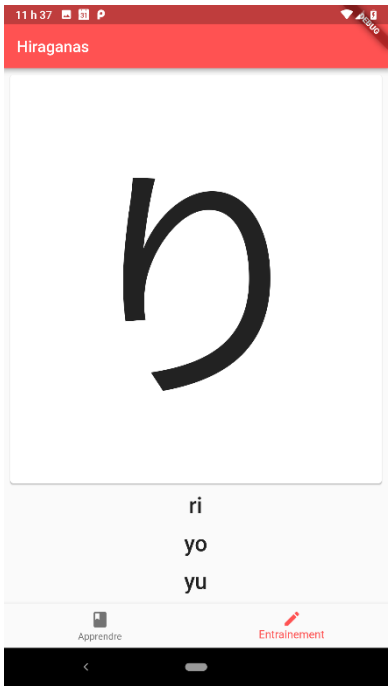
3 Spécifications

3.1 Projet Android Studio

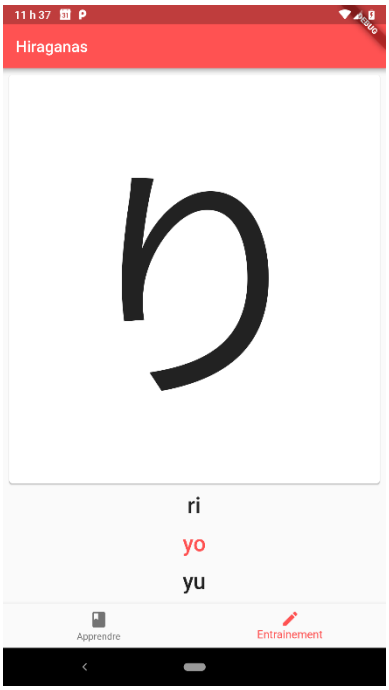
Importez le projet fourni avec cet énoncé. Placez-le dans un dépôt Git (GitHub, GitLab ou BitBucket). Donnez en accès à votre professeur. Tout le code doit être en Dart.

3.2 Interface utilisateur

Apprendre



Entrainement



Entrainement (Erreur)

Il est recommandé de commencer votre travail par la création de l'interface en utilisant de fausses données. Créez des Widgets supplémentaires au besoin. Le mode « Apprendre » doit contenir une grille de cartes pour chaque lettre de l'alphabet choisi. Le mode « Entrainement » doit montrer une lettre (sans la réponse) et une liste de choix. Lors de la sélection d'un choix incorrect, ce dernier doit être désactivé et devenir rouge. Enfin, utilisez « FitedBox » pour agrandir du texte selon la taille de l'écran.

L'interface doit, comme toujours, supporter l'anglais et le français. Pour ce faire, utilisez la classe « Strings » fournie avec le projet. Vous pouvez aussi modifier les couleurs de l'application en modifiant le thème.

3.3 Données de l'application

L'application doit contenir elle-même ses données. Dans le cas des « Hiraganas », elles sont déjà fournies avec le projet dans le fichier « hiragana.dart » sous forme de dictionnaire.

3.4 Fonctionnalités

Description
Au lancement, l'application doit montrer le mode « Apprendre ».
Il doit être possible de passer en tout temps au mode « Apprendre ».
Il doit être possible de passer en tout temps au mode « Entrainement ».
Le mode courant doit être clairement indiqué en rouge dans la barre de navigation en bas de l'écran.
Le mode « Apprendre » doit montrer les lettres de l'alphabet choisi avec leur traduction sous forme de grille de 2 colonnes.
Le mode « Entrainement » doit montrer une lettre au hasard sans sa traduction.
En mode « Entrainement », 3 choix de réponses doivent être montrés, dont 1 est un choix valide.
En mode « Entrainement », les 3 choix de réponse doivent être différents.
En mode « Entrainement », faire un mauvais choix désactive le choix courant et l'affiche en rouge.
En mode « Entrainement », faire le bon choix passe à une nouvelle lettre au hasard ainsi que 3 nouveaux choix de réponse.
L'interface de l'application doit supporter l'anglais et le français.

3.5 Spécifications techniques

Description
L'usage du framework « Flutter » est obligatoire.
Il doit être possible de passer d'un mode à un autre sans perte de données. Autrement dit, l'état de chaque mode doit être conservé en tout temps.

3.6 Autres spécifications

Description
L'interface de l'application doit supporter l'orientation portrait et paysage.

4 Modalités de remise

Remettre sur LÉA un fichier texte incluant :

1. L'adresse vers le dépôt Git
2. Les matricules de chaque membre de l'équipe

Une seule équivaut à une pénalité de 15 %. Au-delà de ce délai, le travail la note « 0 » est attribuée.

5 Évaluation

Éléments
Fonctionnalités (incluant, mais sans s'y limiter) : • En mode « Apprendre », il est possible de consulter l'alphabet avec sa traduction. • Le mode « Entrainement » montre une lettre aléatoire. • Le mode « Entrainement » montre 3 choix de réponse différents. • Le mode « Entrainement » passe à une autre lettre si l'usager fait le bon choix. • Dans le cas d'une mauvaise réponse en mode « Entrainement », le choix est désactivé et mis en rouge. • L'un des 3 choix du mode « Entrainement » est valide.
Interface utilisateur (incluant, mais sans s'y limiter) : • Création d'interface (méthodes « build ») propres. • Visuel de l'interface utilisateur propre et stable. Interface disponible en anglais et en français.
Multiplateforme (incluant, mais sans s'y limiter) : • Utilisation correcte de « Flutter » pour le multiplateforme. • Utilisation des « StatelessWidget » et des « StatefulWidget » dans les cas appropriés. • Séparation correcte de l'application en plusieurs petits Widgets.
Qualité générale de l'application mobile (incluant, mais sans s'y limiter) : • Changement d'orientation et d'application supporté, sans perte de données.
Qualité générale du code (incluant, mais sans s'y limiter) : • Logique bien pensée, juste, et « non patché ». • Découpage adéquat du code en classes, méthodes et packages. • Architecture adéquate au type d'application développée. • Séparation raisonnable des responsabilités entre les classes. • Respects des standards du langage de programmation. • Nommage clair et sans ambiguïté des éléments. • Utilisation de constantes, lorsque nécessaires. • Commentaires compensant uniquement le manque d'expressivité du code. • Respect des bonnes pratiques de programmation générales. • Code propre, sans résidus et facilement lisible. • Aucune erreur ni avertissement à la compilation.
Qualité générale du travail (incluant, mais sans s'y limiter) : • Configuration Gradle fonctionnelle. • Respect des consignes de remise. • Aucun fichier inutile remis avec le projet.

La qualité de la langue française fait partie de l'évaluation. Chaque faute de français retire 0,5 % à la note finale jusqu'à concurrence de 20 %.