Projet final du cours de Flutter

Description de l'application

Principe de base

Vous allez coder une application de quiz.

Le but est de proposer au moins deux quiz. Chaque quiz possède un titre, une description et entre 3 et 6 questions. Une question n'a qu'une seule bonne réponse.

Deux types de questions sont proposées.

Le premier type consiste à laisser l'utilisateur écrire la réponse dans un Textfield.

- S'il n'a rien écrit, on lui indique qu'il doit écrire quelque chose. (via un snackbar, un toast, une popup, ou un fond rouge, etc).
- Si c'est faux, un message indique que la réponse est incorrecte (idem, via une snackbar, etc)
- S'il a écrit la bonne réponse avec des majuscules ou des espaces, la réponse doit être validée.
    - Exemple : si la réponse est "Gandalf", les réponses suivantes sont correctes : "gandalf", "gAndalf", " gandalf " (avec les espaces), etc…
- Si c'est correct, l'utilisateur peut passer à la page suivante.

Le second type de question affiche au moins trois options possibles parmi lesquelles il faut choisir. Si une mauvaise réponse est sélectionnée, la ligne devient rouge. Si la bonne réponse est choisie, la ligne devient verte.

Chaque question est ouverte dans une nouvelle page.
Le numéro de la question et le titre du quiz sont affichés en haut de la page.
Le bouton "suivant" sera désactivé tant que la bonne réponse n'a pas été trouvée.

En cliquant sur le bouton "suivant" de la dernière question, on revient à l'écran principal contenant la liste de quiz.

Idées d'améliorations

Des points seront accordés au fait d'aller plus loin que le principe de base.

Vous pouvez améliorer le concept comme vous voulez. Voici quelques idées :

- Enchaîner les questions dans un ordre random
- Limiter à 3 essais par questions
- Ajouter la notion d'indice pour aider à trouver la bonne réponse
- Trouver d'autres types de questions (en plus du textfield, et du choix multiple)
- Afficher en fin de quiz le score et/ou un résumé des réponses sélectionnées.
- Afficher le score du dernier quiz joué quand on revient à l'écran principal.
- Etc

Requis techniques

Notions attendues

Vous devez reprendre les concepts vu en cours, dont :

- Créez des classes
    - Trois classes sont obligatoires pour le quiz, la question et l'option de réponse. Inspirez vous de la classe Pokémon du cours.
    - Variez les types de paramètres : paramètres nommés, non nommés, obligatoires, facultatifs, etc… (Voir le 2ème cours si besoin)
- Utilisez un routeur pour naviguer entre les pages.
    - La première page contient la liste des quiz.
    - Chaque question du quiz est ouverte dans une nouvelle page.
    - Conseil : pour l'appli de base attendue, vous aurez besoin de 2 ou 3 pages environ.
- Réutilisez les widgets
    - Inspirez vous du widget TheAmazingRow du cours, pour segmenter votre code et éviter le code redondant
- Peaufinez le rendu visuel
    - Utilisez ce qui a été vu en cours : border, borderRadius, padding, couleurs, opacité, placements variés dans les column/row, etc)
    - Pour la charte graphique, vous êtes libre de faire ce que vous voulez.

Idées d'améliorations

Voici quelques idées :

- Utiliser la notion d'héritage
    - Inspirez vous du cours 2
    - Exemple : TextQuestion extends Question, MultipleChoiceQuestion extends Question.
- Afficher des images
- Utiliser des widgets qui n'ont pas été vu en cours : Checkbox, Dropdown, etc
- Essayer de reproduire un design de quiz trouvé sur des sites comme Dribbble, Behance, etc (si c'est le cas, fournissez le lien ou l'image, et citez le nom du graphiste)
- Utiliser un package
- Utiliser des concepts variés de Dart : des enums, des fonctions statiques, des ternaires, etc.
- Ajouter des commentaires explicatifs
- Etc

Conditions du rendu

Date limite

Le projet doit être rendu au plus tard le 16 juin à 8h00.
Passé ce délai, vous perdez 2 points par jour de retard
Si votre projet n'est pas envoyé le 21 juin à 8h, votre note sera zéro.

Git

Le projet doit être mis sur Git.
Le lien du repository doit être envoyé par mail à l'adresse : meriem.tazen@gmail.com
Attention à ne pas mettre votre repository en privé. Si votre projet n'est pas accessible à la date limite, il sera considéré en retard et pénalisé en points de la même manière.

Respect

Aucun contenu raciste, homophobe, transphobe, sexiste, antisémite, etc, ne sera toléré. En cas de non-respect, votre note sera zéro, peu importe la qualité de votre code.

Correction

Si vous souhaitez une correction personnalisée, vous pouvez le demander par mail. Elle vous sera fournie sur une branche git après le 22 juin.