Rapport de TP : Développement d'une application Android

MathQuiz (Vérification de réponses)

Nom : Wssal Kabbour Groupe : GINF3

3 décembre 2025

Introduction

Ce TP a pour objectif de développer une application Android nommée **MathQuiz**, permettant de pratiquer des opérations arithmétiques simples. Dans la version finale, l'application ne se contente plus d'afficher le résultat : l'utilisateur doit saisir lui-même la réponse, et l'application indique si la réponse est correcte ou non. Ce travail permet de se familiariser avec Android Studio, la conception d'interfaces en XML, la gestion des événements en Java et la validation des entrées utilisateur.

Objectifs du TP

Les principaux objectifs pédagogiques sont :
— comprendre la structure d'un projet Android (activité, ressources, Gradle) ;
— concevoir une interface utilisateur avec `TextView`, `EditText` et `Button` ;
— relier les éléments de l'interface au code Java via `findViewById()` ;
— générer aléatoirement des nombres entiers dans un intervalle donné ;
— récupérer et vérifier une réponse saisie par l'utilisateur ;
— afficher un retour (feedback) en cas de bonne ou de mauvaise réponse.

Description de l'application

L'application **MathQuiz** affiche deux nombres entiers aléatoires entre 111 et 999. L'utilisateur choisit une opération ($+$, $-$ ou $\times$), saisit sa réponse dans un champ de texte, puis clique sur le bouton correspondant à l'opération. L'application calcule la bonne réponse, la compare à la valeur saisie et affiche un message :
— **"Bonne réponse !"** si la réponse est correcte ;
— **"Mauvaise réponse. La bonne réponse est : X"** si la réponse est incorrecte ;
— un message d'erreur si la réponse est vide ou invalide.

Un bouton **Générer** permet de créer une nouvelle paire de nombres et de réinitialiser la réponse et le message de résultat.

Interface utilisateur (XML)

L'interface principale est définie dans le fichier `activity_main.xml`. Elle contient les éléments suivants :

— un titre `TextView` pour le nom de l'application ;
— deux `TextView` : `tvNumber1` et `tvNumber2` pour afficher les nombres générés ;
— un `EditText` : `etAnswer` pour permettre à l'utilisateur de saisir sa réponse ;
— un `TextView` : `tvResult` pour afficher le feedback (bonne ou mauvaise réponse) ;
— trois boutons `Button` : `btnAdd`, `btnSub`, `btnMul` pour les opérations ;
— un bouton `btnGenerate` pour générer une nouvelle question.

Le champ `EditText` est configuré avec `inputType="numberSigned"` afin de limiter la saisie à des nombres entiers. L'utilisation de ressources comme `strings.xml` et `colors.xml` permet de séparer la présentation des textes et des couleurs du code Java.

Logique applicative (Java)

La classe principale `MainActivity` étend `AppCompatActivity`. Dans la méthode `onCreate()`, les vues sont reliées au code Java grâce à `findViewById()`, puis une première paire de nombres est générée.

Génération des nombres

La méthode `generateNumbers()` génère deux entiers aléatoires entre 111 et 999 à l'aide de la classe `Random`, puis met à jour les `TextView` :

```
number1 = random.nextInt(889) + 111;
number2 = random.nextInt(889) + 111;
```

Vérification de la réponse

La méthode `checkAnswer(char operation)` est appelée lorsqu'on clique sur un des boutons d'opération. Elle réalise les étapes suivantes :

1. récupérer le texte saisi dans `etAnswer` ;
2. vérifier que le champ n'est pas vide ;
3. convertir la chaîne en entier avec `Integer.parseInt()` ;
4. calculer la bonne réponse en fonction de l'opération choisie ;
5. comparer la réponse de l'utilisateur avec la réponse correcte ;
6. afficher un message adapté dans `tvResult`.

En pseudo-code :

```
si réponse vide -> "Veuillez entrer une réponse."
sinon si format invalide -> "Réponse invalide."
sinon
    calculer réponse correcte
    si réponse utilisateur == correcte
        afficher "Bonne réponse !"
    sinon
        afficher "Mauvaise réponse. La bonne réponse est : X"
```

Gestion des boutons

Chaque bouton d'opération appelle `checkAnswer()` avec le bon caractère :
— `btnAdd` : `checkAnswer('+');`
— `btnSub` : `checkAnswer('-');`
— `btnMul` : `checkAnswer('*');`
Le bouton **Générer** appelle `generateNumbers()`, efface le champ de réponse et remet le message de résultat à `"--"`.

Problèmes rencontrés

Plusieurs difficultés ont été rencontrées lors du développement :
— erreurs de synchronisation Gradle liées à des processus bloqués, résolues en supprimant le cache et en relançant la synchronisation ;
— erreurs de type `Cannot resolve symbol R` causées par des erreurs dans les fichiers XML ;
— gestion des entrées utilisateur vides ou non numériques nécessitant l'utilisation de `try/catch` pour `NumberFormatException` ;
— conflits lors du `git push` vers GitHub, résolus par un `git pull` puis un nouveau `push`.
Ces problèmes ont permis de mieux comprendre le fonctionnement de l'environnement Android Studio, de Gradle et de Git.

Résultat et amélioration possibles

L'application finale est fonctionnelle et interactive :
— génération aléatoire de paires de nombres ;
— saisie de la réponse par l'utilisateur ;
— vérification automatique et feedback immédiat ;
— gestion des erreurs de saisie (champ vide ou valeur invalide).
Parmi les améliorations possibles, on peut envisager :
— ajouter un score (nombre de bonnes réponses) ;
— limiter le temps de réponse (mode chronométré) ;
— proposer plusieurs niveaux de difficulté ;
— ajouter une interface plus graphique et attractive.

Conclusion

Ce TP m'a permis de mettre en pratique les bases du développement Android : conception d'interface, manipulation des ressources, gestion des événements et validation d'entrées utilisateur. La version enrichie de **MathQuiz**, qui vérifie les réponses saisies, est plus pédagogique et interactive que la version initiale, et constitue une bonne introduction au développement d'applications mobiles éducatives.