Évaluation formative 05

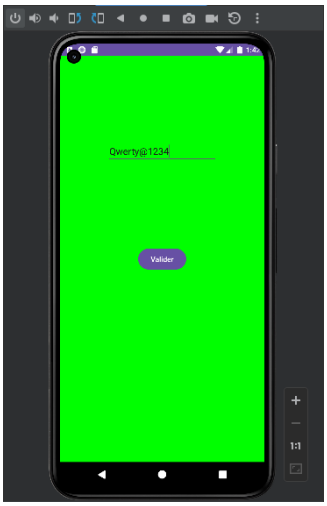
Objectif

Se familiariser avec la capture et gestion d'événements.

Problème

Nous avons vu en classe comment capturer les touchers de bouton dans une application *Android Studio*.

Dans le cadre de cette évaluation formative, vous devez concevoir une application *Android* validant la robustesse d'un mot de passe. Voici les spécifications de cette application :



1. Un `EditText` permet à l'utilisateur d'entrer en texte clair un mot de passe à tester.

2. Le `Button` permet à l'utilisateur de tester la robustesse du mot de passe fourni.

3. Lorsque l'utilisateur touche le bouton « VÉRIFIER », l'application calcule la robustesse du mot de passe à l'aide de la fonction **`évaluerRobustesseMotDePasse()`** ci-dessous, puis change la couleur de fond de l'activité selon le résultat obtenu :

    a. Si la robustesse du mot de passe est évaluée à moins de 0.50, la couleur de fond de l'activité doit passer au rouge.

    b. Si la robustesse du mot de passe est évaluée à plus de 0.75, la couleur de fond de l'activité doit passer au vert.

    c. Sinon, la couleur de fond de l'activité doit passer au jaune.

4. Votre projet doit exploiter la liaison de vues pour répondre aux actions de l'utilisateur.

Matériel fourni

Voici la fonction permettant d'évaluer la robustesse d'un mot de passe :

```
// Évaluation de la robustesse d'un mot de passe (0.0 = faible, 1.0 = fort)
private fun évaluerRobustesseMotDePasse(motDePasse: String): Float {
    var robustesse: Float = 0f

    // Y a-t-il au moins une minuscule?
    if (motDePasse.matches(Regex(".*[a-z]+.*")))
        robustesse += 0.25f

    // Y a-t-il au moins une majuscule?
    if (motDePasse.matches(Regex(".*[A-Z]+.*")))
        robustesse += 0.25f

    // Y a-t-il au moins un chiffre?
    if (motDePasse.matches(Regex(".*[\\d]+.*")))
        robustesse += 0.25f

    // Y a-t-il au moins un symbole?
    if (motDePasse.matches(Regex(".*[@#$%]+.*")))
        robustesse += 0.25f

    // Y a-t-il au moins six caractères?
    if (motDePasse.length < 6)
        robustesse -= 0.2f

    return robustesse
}
```

À soumettre

Une fois l'exercice solutionné, démontrez à l'instructeur que ça fonctionne bien.

Suggestions

Voici quelques suggestions qui vous aideront à solutionner cet exercice :

1. La propriété `EditText.`**`text`** retourne une instance de la classe `Editable`. Pour convertir cette instance en `String`, il faut exploiter la méthode **`toString()`**. Par exemple :

```
val motDePasse: String = binding.editText.text.toString()
```

2. *Android Studio* n'attribue pas automatiquement un identificateur au `ConstraintLayout` de base d'une activité. Vous devez donc explicitement lui en attribuer un afin de pouvoir y faire référence via **`binding`**.

3. Pour changer la couleur d'arrière-plan d'un `View`, utilisez le mutateur **`setBackgroundColor()`**. Investiguez sur *Google* comment spécifier un couleur en *Kotlin* sur *Android*.