TP - Calculator

Flutter

Objectifs

- Initiation à Flutter.
- Emploi du CLI de Flutter.
- Découverte du langage Dart.
- Découverte du Paradigme Déclaratif.

Aidez-vous de la documentation officielle de Flutter pour réaliser ce TP (cf. https://docs.flutter.dev/cookbook).

Etape 01 : Initialisation

- Créez un nouveau projet Flutter nommé ***calculator*** à l'aide de votre terminal de commande (ou via votre IDE).
  Par défaut, le projet généré par le CLI de Flutter va contenir les sources du projet *"Counter"*.

  $ `flutter create calculator`

- Vérifiez que votre application fonctionne en l'exécutant dans votre navigateur web ou dans l'émulateur de votre choix (mobile ou desktop).
  $ `flutter run`

Etape 02 :
Adaptation fonctionnelle

- Adaptez le fichier ***./lib/main.dart*** afin d'afficher le terme *"Calculator"* dans le widget ***AppBar*** à la place du titre affiché par défaut.
- Personnalisez la couleur d'arrière-plan de l'***AppBar*** en adaptant le ***ThemeData*** de l'application.
- Faites en sorte que la valeur de la variable ***_counter*** soit à présent incrémentée de **2 en 2**.
- Au centre de l'écran, affichez **dynamiquement** la mention suivante en fonction de la valeur courante de la variable d'état ***_counter***.

  Pour ce faire, employez 2 widgets de type ***Text*** :

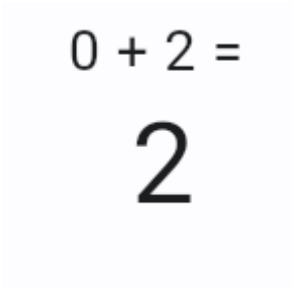
  - Un premier widget ***Text*** affichant dynamiquement l'expression mathématique, soit :

    `"<counter> + <increment> ="`

    ex : 0 + 2 =

  - Un second widget ***Text*** affichant la valeur obtenue en appliquant l'expression mathématique, ***_counter*** + ***_increment***, soit :

    `<counter> + <increment>`



Etape 03 : Refactoring

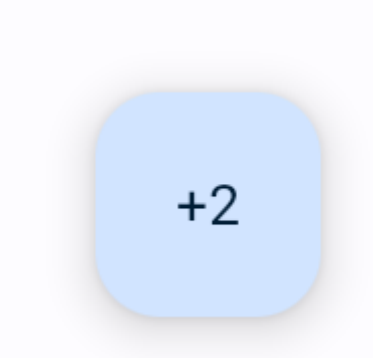
- Pour faciliter la lecture, retirez les commentaires superflus présents dans le code généré par défaut lors de la création du projet.
- Observez le code du fichier ***./lib/main.dart*** et identifiez les différents éléments de l'application (classes, fonction…).
- Pour plus de cohérence, renommez la classe ***MyHomePage*** en ***Calculator*** et la classe ***_MyHomePageState*** en ***_CalculatorState***.
- Dans le dossier ***./lib***, créez un fichier avec l'extension ***.dart*** pour chaque élément identifié et déplacez la portion de code correspondante dans chaque fichier créé. Respectez les conventions de nommage définies par Flutter et Dart (code en *camelCase*, fichier en *snake_case*).

  Vérifiez que l'onglet *"problèmes" (*ou *errors)* de votre *IDE* est vide. Si des messages sont affichés, effectuez les corrections indiquées.

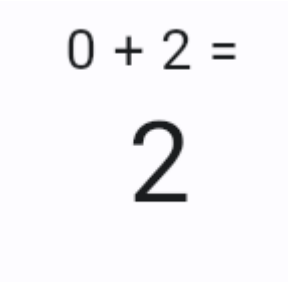
  Important : pour un ***Widget Stateful***, conservez la classe définissant le *Widget Stateful* et la classe définissant son état dans un seul et même fichier.
- Dans votre fichier ***./lib/main.dart*** et dans les nouveaux fichiers créés, importez les éléments nécessaires afin de reconstituer l'application.

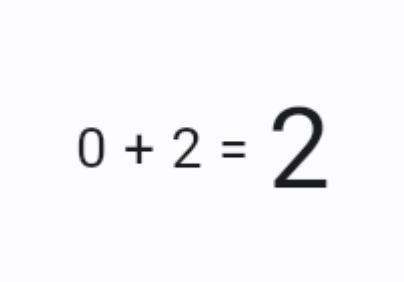
Etape 04 : Personnalisation graphique

- Modifiez la couleur primaire employée dans le thème de l'application Flutter (par défaut violet clair). Sélectionnez la couleur de votre choix.
- A la place de l'icône *"+"* en bas à droite, affichez la mention *"+2"* sur le bouton.



- Affichez les 2 widgets *Text* **l'un à côté de l'autre**, plutôt que l'un au-dessus de l'autre.

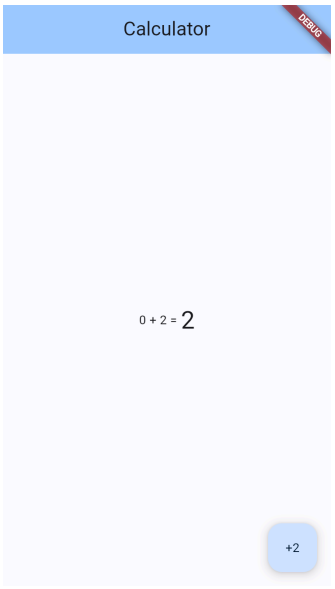




Avant Après

(cf. aperçu page suivante)

Aperçu



Etape 05 : Affichage conditionnel

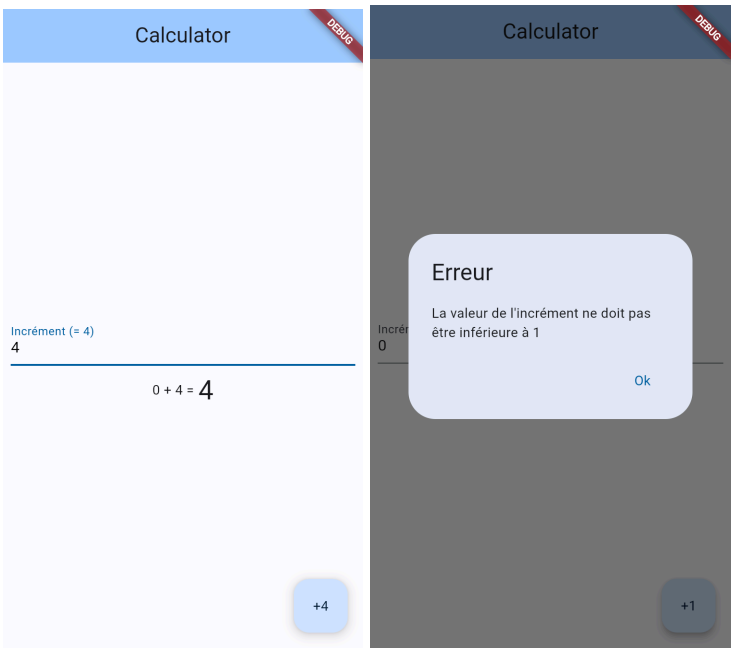
- Affichez dynamiquement le texte **`Vous avez cliqué <n> fois`**, à condition que l'utilisateur ait cliqué au moins 1 fois sur le bouton *"+2"*.
- Mettez à jour le texte à chaque clic.

Aperçu

Etape 6 : Formulaire

- Permettez à l'utilisateur de personnaliser la valeur de l'incrément en remplissant un champ de texte dans un formulaire. A ce stade, la valeur de l'incrément est égale à 2.
- Faites en sorte que l'utilisateur ne puisse saisir que des valeurs numériques.
- Si la valeur saisie est égale à 0
    - affichez une alerte,
    - faites en sorte que la valeur de l'incrément passe à 1.

Aperçu



Etape 7 : Opérations mathématiques

- Permettez à l'utilisateur de sélectionner le type d'opération mathématique à appliquer au moment de cliquer où il clique sur le *FloatingActionButton*.
- Par défaut, l'opération appliquée est une addition de la valeur d'incrément à la variable _counter.
- L'objectif est de permettre à l'utilisateur d'effectuer au choix : une multiplication, une soustraction et une division de ces 2 valeurs.
- Ajouter un widget de type *DropdownButton* permettant de sélectionner le type d'opération mathématique et adaptez dynamiquement les éléments de l'interface :
    - icône du FloatingActionButton,
    - texte du détail de l'opération mathématique.

Aperçu