|  |  |
| --- | --- |
| A picture containing text, tableware, dishware, plate  Description automatically generated | **Travail pratique #2** |
| **Expérience utilisateur**  **d’applications mobiles** | |
| **Code : 582-5PP-BB**  **Programme : Technique d’intégration multimédia** | |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | Session : | **Automne 2021** | | | Pondération : | **20 %** | | | Contexte : | ☑ | **Individuel** | |  | ☐ | **En équipe** | | |  |  | | --- | --- | | Enseignant : | **Carlos Tremblay** | | Téléphone : |  | | Courriel : | **carlos.tremblay@bdeb.qc.ca** | | Bureau : |  | | | |

|  |  |
| --- | --- |
| **Description :** | Ce travail pratique vise à rendre l’étudiant à l’aise avec l’environnement de développement Visual Studio Code, le langage Dart et l’ajout d’éléments interactifs dans une application Flutter. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Échéancier :** | Groupe 1 | | **Mercredi 3 novembre 2021** | | | | |
|  |  | |  | | | | |
| **À remettre :** | Avant | ☐ | le début du cours | ☐ | la fin du cours | ☑ | minuit |

|  |  |  |  |
| --- | --- | --- | --- |
| **Dépôt :** | Physique | ☐ | Local : X-XXX |
|  | Numérique | ☑ | Dans Teams |

|  |  |
| --- | --- |
| **Fichiers :** | 📂 1234567\_grXX\_TP2.zip  ⤷ 📂 assets  ⤷ 🎬 shopping.jpeg  ⤷ 🎬 etc.jpeg  ⤷ 📂 lib  ⤷ 📄 main.dart  ⤷ 📄 shopping\_screen.dart  ⤷ 📄 etc.dart  ⤷ 📄 pubspec.yaml |

**Travail pratique #2**

# Exploration de l’environnement de développement VS Code

## 1 - Génération de code avec **Flutter Tree *(1 point)***

1. Installez l’extension **Flutter Tree** dans **VS Code**
2. Générez une nouvelle application
3. Remplacez tout le code dans le fichier **main.dart** avec le code d’un écran de départ vide à partir du [bloc-notes](https://bdebqcca.sharepoint.com/sites/Section_20213-EO-5825PPBB-000001/_layouts/15/Doc.aspx?sourcedoc=%7bf6f57eda-da32-47b9-95aa-8b52d8814786%7d&action=edit&wd=target%28_Biblioth%C3%A8que%20de%20contenu%2FFlutter.one%7C64e15a98-8fae-4196-9acb-c9f2ec472f56%2F%29&wdorigin=717) dans Teams
4. Remplacez tout le contenu de la propriété **body** avec le code suivant :

**body: Center>Column[Container,Expanded>Container,Text]**

* 1. Effacez le dernier crochet carré, puis retapez-le : ]
  2. Vous devriez voir apparaître ceci :

Website

Description automatically generated

* 1. Tapez un retour à la ligne ou un tab
  2. Observez que l’extension **Flutter Tree** générera tout le code automatiquement pour vous.

1. Éliminez les erreurs (soulignés rouges) et avertissements (soulignés bleus) qui subsisteraient après la création du code
   1. Indice : installez l’extension **Error Lens**, si ce n’est déjà fait, cela vous aidera à repérer les erreurs plus facilement dans le code
   2. Indice : le raccourci **CTRL + .** (point) peut vous suggérer des corrections et les appliquer pour vous
2. Copiez une capture d’écran du code Dart généré ici :

|  |
| --- |
|  |

## 2 - Remaniement du code (« refactoring ») ***(1 point)***

1. Exécutez l’application (F5)
   1. Prenez de l’emplacement du texte dans l’écran car les prochaines instructions le déplaceront.
2. Placez votre curseur et cliquez une fois sur le widget **Expanded**
   1. Dans le menu contextuel (clic droit), choisissez « *Refactor*… » (ou « *Remanier*… », si VS Code est en français), puis « *Remove this widget* ».
   2. Sauvegardez votre code pour observer le changement dans l’écran de l’émulateur.
3. Ajoutez l’attribut **color** avec une couleur différente de votre choix dans chacun des **Container**s. Remarquez que la couleur n’apparait pas pour le premier container.
4. Placez à nouveau votre curseur sur le premier widget **Container** et faites **CRTL +SHIFT + R** (*Refactoring…*) et choisissez « *Wrap with SizedBox* ». Répétez pour le 2e **Container**.
5. Maintenant ajoutez un attribut **height** pour chacun des **SizedBox** en plaçant votre curseur juste après la 1ère parenthèse et en faisant **CTRL + ESPACE BAR**. Choisissez des valeurs de 100 et 200 pixels, respectivement. Sauvegardez votre code.
6. Augmentez les valeurs de l’attribut **height** graduellement par intervalles de 100 pixels jusqu’à ce que vous obteniez cet avertissement dans l’émulateur :

A picture containing text, crosswalk, clipart

Description automatically generated

1. Maintenant, remplacez le nom du widget **Column** par **ListView**. L’erreur devrait disparaitre et maintenant vous devriez pouvoir faire défiler l’écran verticalement.
2. Copiez 2 captures d’écran de la première erreur générée, puis de l’écran corrigé, ici :

|  |
| --- |
|  |

## 3 - Explorer un projet : trouver toutes les références à une classe ***(1 point)***

1. Dans ce même projet, remplacez tout le code dans le fichier **main.dart** avec le code se trouvant [dans cette recette](https://flutter.dev/docs/cookbook/effects/expandable-fab) sur le site web de Flutter (le code à copier se trouve au bas de la page dans la section *Interactive example*)
   1. Trouvez la première référence à la classe FakeItem :
      1. **CTRL** + **F** , puis tapez *FakeItem*
   2. Pour trouver tous les autres endroits dans le code où cette classe est utilisée, placez votre curseur sur le texte *FakeItem* dans le code, puis, dans le menu contextuel (clic droit), choisissez « *Find all references* » ou encore faites *SHIFT + ALT + F12*
   3. Indiquez à quoi correspond chacun des 3 résultats de la recherche :

Text

Description automatically generated

1 : Définition de la classe

2 : Appel au constructeur pour instancier un objet de cette classe

3 : Définition du constructeur de la classe

## 4 – Énoncés vs. Expressions ***(2 points)***

1. Familiarisez-vous avec les différences entre [énoncés (« statements ») et expressions](https://programming.guide/statements-vs-expressions.html) (dans cette page web, ignorez la 2e partie intitulée *Details*)
2. Répondez en remplissant les colonnes de droite. La 3e colonne n’est à compléter que s’il s’agit d’une expression.

|  |  |  |
| --- | --- | --- |
| Code | **Énoncé** ou **expression** ? | Valeur (et type) de **l’expression**  (seulement si applicable) |
| 1 + 4 | Expression | Ex : 5 (int) |
| a = 5 | Énoncé |  |
| b == 6 | Expression | 6 (bool) |
| print(‘hello’) | Énoncé |  |

## 5 - Méthodes map() et where() ***(3 points)***

1. Familiarisez-vous [avec la **méthode** map()](https://www.codevscolor.com/dart-map-method-example) ([tutoriel](https://www.codevscolor.com/dart-map-method-example)) et la méthode **where**() ([tutoriel](https://flutterfix.com/understanding-flutter-foreach-map-where/))
2. À l’aide d’un navigateur web, rendez-vous sur le site web <http://dartpad.dev>
3. Ré-utilisez la classe Personne du premier TP et remplacez la propriété **age** par une propriété **dateDeNaissance** de type **DateTime** de façon à ce que vous puissiez instancier une liste de personnes comme suit :

final personnes = [

Personne("Justin", DateTime.parse("1971-12-25"), "Montreal"),

Personne("Erin", DateTime.parse("1973-01-22"), "Quebec"),

Personne("Jagmeet", DateTime.parse("1979-01-02"), "Ottawa"),

];

1. ***(2 points)*** À partir de cette liste de personnes, obtenez une nouvelle liste qui ne contient que les âges de chacune des personnes. Donc cette nouvelle liste contiendra ceci :

ages = (49, 48, 42)

Vous aurez besoin de :

* La méthode **map**() (à ne pas confondre avec les *objets* de type Map – avec un M majuscule)
* Plusieurs des méthodes de la classe DateTime (now, difference, inDays)
* La méthode **toInt**() pour ne garder que la partie entière d’un nombre

1. ***(1 point)***Ensuite créez une autre liste d’âges qui ne contient que les âges strictement inférieurs à 49. Donc cette liste contiendra ceci :

jeunes = (48, 42)

Vous aurez besoin de la méthode suivante : **where()**

****

## 6 – Création d’une nouvelle branche git **(1 point)**

1. Si ce n’est déjà fait, cloner le projet contenant le code de votre premier TP
2. Cliquez sur le nom de la branche actuelle (probablement **master**) en bas à gauche dans la barre de statut de VS Code.
3. Saisissez le nom de la nouvelle branche dans le champ : **feature/tp02**

Graphical user interface, text, application, email

Description automatically generated

1. En créant cette nouvelle branche, vous avez automatiquement fait un « checkout » de cette branche, c’est-à-dire que vous vous situez maintenant sur cette nouvelle branche, tel qu’indiqué par le nom de la branche affiché dans la barre de statut en bas à gauche

Graphical user interface, application

Description automatically generated

1. Faites un changement quelconque dans le fichier main.dart
2. Faire un **commit** avec le message *« test branche »* suivi d’un **push**
3. Pour fusionner ce changement dans la branche **master**, faites un **checkout** de la branche **master** en cliquant sur le nom de la branche en bas à gauche, puis choisissez **master** dans la liste qui s’ouvrira en haut.
4. Maintenant, ouvrez le menu de la section *SOURCE CONTROL (attention, le 2e en-tête SOURCE CONTROL, pas le 1er)* et choisissez *Branch/Merge Branch…,*  puis la branche **feature/tp02**

Graphical user interface, website

Description automatically generated

1. Remarquez que maintenant vous avez un nouveau commit qui attend d’être poussé sur le serveur. Simplement le pousser en cliquant sur la barre de statut

Graphical user interface

Description automatically generated

1. Prenez une capture d’écran de la branche et du commit sur le serveur GitHub. Exemple :

Graphical user interface, text, application, email

Description automatically generated

|  |
| --- |
|  |

## 7 – UX **(1 point)**

1. Pour 2 applications mobiles de votre choix, trouver la tâche principale que les concepteurs ont choisi mettre de l’avant. Coller les 2 captures d’écran ici et pour chacune d’entre elles, indiquez quelle est la tâche principale.

|  |
| --- |
| No description available.  #1 VaxiCode  Nous pouvons clairement voir que  l’action principal de cette application est  D’être un répertoire d’information et  L’action principale est bien sûre d’ouvrir  Le répertoire  #2 Cookie clicker  No description available.Dans ce jeu l’action principale est comme le nom le suggère de  Cliquer sur le biscuit. |

## 8 – Flutter **(10 points)**

Partez de l’application que vous avez complétée au TP précédent.

1. ***(3 points)*** Créez une nouvelle page qui affiche une liste (**ListView**) d’au moins 10 éléments
   1. *Pour créer une liste :* [*https://flutter.dev/docs/cookbook/lists/mixed-list*](https://flutter.dev/docs/cookbook/lists/mixed-list)
   2. *Pour afficher des images de l’internet :* [*https://flutter.dev/docs/cookbook/images/network-image*](https://flutter.dev/docs/cookbook/images/network-image)
2. ***(3 points)*** Navigation entre 2 écrans
   1. Naviguez vers ce nouvel écran en cliquant sur un des boutons (à votre choix)
   2. *Indice :* [*https://flutter.dev/docs/cookbook/navigation/navigation-basics*](https://flutter.dev/docs/cookbook/navigation/navigation-basics)
3. ***(4 points)*** Augmenter la valeur perçue d’un écran
   1. Clonez le projet Flutter suivant :

<https://github.com/nomadenumerique/tp02.git>

* 1. Améliorez la valeur perçue de l’application en intervenant au niveau d’un ou de plusieurs aspects suivants :
     1. L’espacement entre les éléments visuels
     2. Alignement et groupement des éléments
     3. La typographie
     4. Les couleurs. Choisissez une palette de couleurs sur le site **coolors.co**
     5. L’iconographie
     6. Les animations
  2. Indices :
     1. Utilisez la propriété **decoration** des **Container**s avec un **BoxDecoration**.
     2. Dans le BoxDecoration, vous pouvez utiliser la propriété **gradient**, **image**
     3. Pour partager les styles à travers tous les écrans de l’application, utilisez un Theme dans le widget **MaterialApp.**

*Voir la documentation :* [*https://flutter.dev/docs/cookbook/design/themes*](https://flutter.dev/docs/cookbook/design/themes)

* + 1. Dans l’objet **ThemeData**, vous pouvez ensuite définir un **TextTheme** pour partager les styles (fonts, taille, couleur) des widgets **Text**
    2. Vous avez accès à plus de 1000 fontes dans le package Google\_fonts

*Voir la documentation :* [*https://mrflutter.com/how-to-use-google-fonts-in-flutter*](https://mrflutter.com/how-to-use-google-fonts-in-flutter)