Comment créer des Map dans Dart

1. Objectif

- Comprendre le concept de Map en Dart.
- Apprendre à déclarer, utiliser et manipuler des Maps en Dart.
- Explorer les différentes propriétés et fonctions disponibles pour travailler avec les Maps.

2. Présentation

- Les Maps en Dart sont des collections d'objets qui associent des clés uniques à des valeurs. Chaque clé est associée à une valeur spécifique, ce qui permet un accès rapide et efficace aux données. Les Maps sont utilisées pour stocker des données structurées de manière associative.

3. Déclaration des Maps dans Dart

- En Dart, vous pouvez déclarer et Map de plusieurs manières. Voici comment faire :

1. **Utilisation de la notation littérale en Dart**

   - Vous pouvez créer un ensemble en utilisant une notation littérale, en entourant les éléments par des accolades `{}`.
   - Pour déclarer une carte à l'aide de littéraux de carte, vous devez placer les paires clé-valeur entre une paire d'accolades "`{}`" .

```
// Déclaration d'une Map à l'aide de littéraux de Map
Map<String, int> scores = {
  'Alice': 100,
  'Bob': 90,
  'Charlie': 80,
};
```

   > - *La syntaxe des littéraux de Map permet de définir directement les paires clé-valeur.*

2. **Utilisation d'un constructeur de Map (`Constructeur Dart Map<K, V>()`)**

apcpedagogie.com

```
// Déclaration d'une Map à l'aide d'un constructeur de Map
Map<String, String> fruits = Map();
fruits['apple'] = 'Pomme';
fruits['banana'] = 'Banane';
fruits['orange'] = 'Orange';
```

4. Ajout d'un élément dans un ensemble Dart

- Pour ajouter un élément dans l'ensemble, nous utilisons la fonction `.add()` ou la fonction `.addAll()`.
- En Dart, pour ajouter un élément ou plusieurs éléments dans un ensemble, vous pouvez utiliser les méthodes `add()` et `addAll()`. Voici comment fonctionnent ces deux méthodes avec des exemples :

1. **add()**

   - La méthode `add()` permet d'ajouter un seul élément à un ensemble. Si l'élément est déjà présent dans l'ensemble, il ne sera pas ajouté à nouveau, car les ensembles n'acceptent que des éléments uniques.

```
Set<String> ensembleDeNoms = Set<String>();

ensembleDeNoms.add('Alice');
ensembleDeNoms.add('Bob');
ensembleDeNoms.add('Charlie');
ensembleDeNoms.add('David');

// Tentative d'ajout d'un élément en double
ensembleDeNoms.add('Alice'); // Ignoré, car 'Alice' est déjà dans l'ensemble

print(ensembleDeNoms); // Affiche : {'Alice', 'Bob', 'Charlie', 'David'}
```

2. **addAll()**

   - La méthode `addAll()` permet d'ajouter plusieurs éléments en une seule opération. Vous pouvez lui passer une liste ou un autre ensemble contenant les éléments que vous souhaitez ajouter à l'ensemble existant.

apcpedagogie.com

```
Set<String> ensembleDeNoms = Set<String>();

ensembleDeNoms.addAll(['Alice', 'Bob', 'Charlie']);
ensembleDeNoms.addAll(['David', 'Eve']);

// Tentative d'ajout d'éléments en double
ensembleDeNoms.addAll(['Alice', 'Eve']); // Ignorés, car 'Alice' et
'Eve' sont déjà dans l'ensemble

print(ensembleDeNoms); // Affiche : {'Alice', 'Bob', 'Charlie', 'David',
'Eve'}
```

- La méthode `addAll()` vous permet d'ajouter plusieurs éléments à la fois sans avoir à appeler add() individuellement pour chaque élément.
- En résumé, la méthode `add()` ajoute un seul élément à un ensemble, tandis que la méthode addAll() ajoute plusieurs éléments en une seule opération. Les éléments en double sont ignorés dans les deux cas, car les ensembles ne peuvent contenir que des éléments uniques.

5. ## Accéder à l'élément d'un l'ensemble Dart

   - En Dart, pour accéder à un élément spécifique d'un ensemble (Set), vous pouvez utiliser les méthodes `elementAt()`, `for-in`, et `forEach`. Voici comment chacune de ces méthodes fonctionne pour accéder aux éléments d'un ensemble :

6. **Boucle `for-in`**

   - Vous pouvez utiliser une boucle `for-in` pour parcourir tous les éléments d'un ensemble et les inspecter à la recherche de l'élément souhaité. Vous pouvez mettre en place une condition pour identifier l'élément recherché.

```
Set<String> couleurs = {'rouge', 'vert', 'bleu'};

for (String couleur in couleurs) {
  if (couleur == 'bleu') {
    print("La couleur bleue a été trouvée dans l'ensemble.");
  }
}
```

apcpedagogie.com

7. **Méthode `forEach`**

   - La méthode `forEach` permet de parcourir tous les éléments d'un ensemble en spécifiant une fonction callback qui sera exécutée pour chaque élément. Vous pouvez vérifier chaque élément dans la fonction callback pour trouver l'élément recherché.

```
Set<String> couleurs = {'rouge', 'vert', 'bleu'};

couleurs.forEach((couleur) {
  if (couleur == 'rouge') {
    print("La couleur rouge a été trouvée dans l'ensemble.");
  }
});
```

8. **Méthode `elementAt()`**

   - Dart fournit la méthode `elementAt()`, qui est utilisée pour accéder à l'élément en transmettant sa position d'index spécifiée.
   - La méthode `elementAt()` permet d'accéder à un élément d'un ensemble en fonction de son indice. Cependant, il est important de noter que les ensembles ne sont pas indexés par défaut, ce qui signifie qu'ils n'ont pas d'ordre particulier.
   - Pour utiliser elementAt(), vous devez d'abord convertir l'ensemble en une liste, puis accéder à l'élément par indice.

```
Set couleurs = {'rouge', 'vert', 'bleu'};
List listeDeCouleurs = couleurs.toList();

String couleur = listeDeCouleurs.elementAt(1); // Accès au deuxième élément (indice 1)

print(couleur); // Affiche : 'vert'
```

Cours Flutter

Source : https://apcpedagogie.com/comment-creer-des-map-dans-dart · téléchargé le 08/11/2026 · © APC Pédagogie · apcpedagogie.com