Les chaines de caractères en langage Dart

1. ## Présentation

   - en Dart, une chaîne (String) est une séquence d'unités de code UTF-16, et vous pouvez utiliser des guillemets simples (') ou doubles (") pour créer des chaînes. Vous avez également la possibilité d'échapper le délimiteur de chaîne si nécessaire.

2. **Utilisation de guillemets simples :**

   ```
   var s1 = 'Je suis une chaîne délimitée par des guillemets simples.';
   var s3 = 'comment échapper le caractère d'apostrophe (').';
   ```

   - Dans le premier exemple, **s1** contient une chaîne délimitée par des guillemets simples. Dans le troisième exemple, **s3** illustre comment échapper le caractère d'apostrophe (') en utilisant ' à l'intérieur d'une chaîne délimitée par des guillemets simples.

3. **Utilisation de guillemets doubles :**

   ```
   var s2 = "Je suis une chaîne délimitée par des guillemets doubles.";
   var s4 = "comment inclure un caractère d'apostrophe sans avoir besoin de
   l'échapper";
   ```

   - De manière similaire, les exemples avec des guillemets doubles fonctionnent de la même manière. **s2** contient une chaîne délimitée par des guillemets doubles. Le quatrième exemple, **s4**, montre comment inclure un caractère d'apostrophe sans avoir besoin de l'échapper, car la chaîne est délimitée par des guillemets doubles.
   - Les guillemets simples et doubles sont équivalents en Dart pour définir des chaînes de caractères, vous pouvez utiliser celui qui vous convient le mieux en fonction de vos besoins.

- ## Une chaîne de caractères brute dans dart

  - En Dart, vous pouvez créer des chaînes de caractères brutes en utilisant le préfixe r avant la déclaration de la chaîne. Une chaîne brute est une chaîne de caractères dans laquelle les caractères d'échappement (comme `\n`, `\t`, `\\`, etc.) ne sont pas interprétés et sont traités littéralement. Voici comment vous pouvez déclarer une chaîne de caractères brute en Dart :

apcpedagogie.com

```
String chaineBrute = r'Ceci est une chaîne brute avec des caractères d'échappement non traités.';
```

- Dans l'exemple ci-dessus, la chaîne chaineBrute contient la séquence de caractères telle quelle, sans interpréter le caractère d'échappement \. Cela signifie que chaineBrute contiendra le texte "Ceci est une chaîne brute avec des caractères d'échappement non traités." et n'interprétera pas le caractère \ comme un caractère d'échappement.
- Les chaînes brutes sont utiles lorsque vous devez travailler avec des chaînes qui contiennent fréquemment des caractères d'échappement ou des expressions régulières, car elles évitent d'avoir à échapper manuellement chaque caractère d'échappement.
- Vous pouvez également utiliser des chaînes brutes avec des guillemets simples ou doubles, selon vos besoins. Par exemple :

```
String chaineBrute1 = r'Une chaîne brute avec guillemets simples : '';
String chaineBrute2 = r"Une chaîne brute avec guillemets doubles : \"";
```

- Les chaînes brutes en Dart facilitent le traitement de certaines données sans avoir à s'inquiéter de l'interprétation des caractères d'échappement.

Concaténer deux chaînes de caractères dans Dart

- Pour concaténer deux chaînes de caractères en Dart, vous avez plusieurs options, notamment l'utilisation de l'opérateur `+`, la méthode `concat()`, et l'interpolation de chaînes. Voici comment effectuer chaque opération :

1. **Utilisation de l'opérateur + :**
   - Vous pouvez utiliser l'opérateur + pour concaténer deux chaînes de caractères en Dart.

```
String chaine1 = "Hello";
String chaine2 = " World";
String resultat = chaine1 + chaine2;
print(resultat); // Résultat : "Hello World"
```

2. **Utilisation de la méthode join() :**

- Vous pouvez utiliser la méthode `join()` pour concaténer des chaînes. Cette méthode prend un itérable de chaînes (par exemple, une liste) comme argument et renvoie la chaîne résultante.

```
List mots = ["Bonjour", " le", " monde"];
String resultat = mots.join();
print(resultat); // Résultat : "Bonjour le monde"
```

3. **Utilisation de l'interpolation de chaînes :**

   - L'interpolation de chaînes est une manière élégante de concaténer des chaînes avec des variables ou d'autres expressions.

```
String prenom = "John";
String nom = "Doe";
String nomComplet = "Nom : $nom, Prénom : $prenom";
print(nomComplet); // Résultat : "Nom : Doe, Prénom : John"
```

4. Ces méthodes de concaténation de chaînes vous permettent de créer des chaînes plus complexes en combinant plusieurs chaînes ou en insérant des valeurs dynamiques dans vos chaînes. Vous pouvez choisir la méthode qui convient le mieux à votre situation.

Mettre une variable dans une chaîne de caractères en dart

- En Dart, vous pouvez insérer des variables dans une chaîne de carble dans une chaîne, vous devez utiliser le sigactères en utilisant l'interpolation de chaînes.
- Pour insérer une variable dans une chaîne, vous devez utiliser le signe de **dollar** `$` suivi du nom de la variable entre **accolades** `{}`. Voici comment cela fonctionne :

```
String nom = "Alice";
int age = 30;

String message = "Bonjour, je m'appelle $nom et j'ai $age ans.";
print(message); // Résultat : "Bonjour, je m'appelle Alice et j'ai 30 ans."
```

- Dans l'exemple ci-dessus, nous avons utilisé l'interpolation de chaînes pour insérer les valeurs des variables nom et age dans la chaîne "Bonjour, je m'appelle $nom et j'ai $age ans.".
- Vous pouvez également effectuer des opérations ou des expressions plus complexes à l'intérieur des accolades. Par exemple :

apcpedagogie.com

```
int x = 10;
int y = 20;

String resultat = "La somme de $x et $y est ${x + y}";
print(resultat); // Résultat : "La somme de 10 et 20 est 30"
```

- L'interpolation de chaînes est une façon pratique d'incorporer des variables et des expressions dans des chaînes de caractères en Dart. Elle rend votre code plus lisible et plus efficace en évitant de concaténer manuellement des chaînes et des valeurs.

Index d'un caractère d'une chaîne de caractères en Dart

1. **Utilisation des crochets `[]`:**
   - Les **crochets []** vous permettent d'accéder à un caractère spécifique dans une chaîne de caractères en utilisant son index.
   - L'indice commence à zéro pour le premier caractère, et vous pouvez accéder à un caractère en utilisant `chaine[index]`.
   - Les **crochets []** sont principalement utilisés pour l'accès direct aux caractères d'une chaîne de caractères en fonction de leur position.

```
String texte = "Dart";
char premierCaractere = texte[0]; // Accès au premier caractère "D".
char deuxiemeCaractere = texte[1]; // Accès au deuxième caractère "a".
```

2. **La méthode `indexOf()`:**
   - En Dart, vous pouvez obtenir l'index d'un caractère particulier dans une chaîne de caractères à l'aide de la méthode **`indexOf()`** de la classe **String**.
   - Cette méthode recherche la première occurrence du caractère spécifié et renvoie son indice (position) dans la chaîne. Si le caractère n'est pas trouvé, la méthode renvoie -1.
   - Voici comment utiliser **indexOf()** pour obtenir l'index d'un caractère dans une chaîne :

```
String texte = "Dart est génial !";
char recherche = 't';
int index = texte.indexOf(recherche);
```

apcpedagogie.com

```
if (index != -1) {
  print("L'indice du caractère '$recherche' est $index.");
} else {
  print("Le caractère '$recherche' n'a pas été trouvé dans la chaîne.");
}
```

- Dans cet exemple, nous cherchons l'indice du caractère 't' dans la chaîne texte. Si le caractère est trouvé, nous affichons son indice. Sinon, nous affichons un message indiquant que le caractère n'a pas été trouvé.
- Il est important de noter que `indexOf()` ne renverra que la première occurrence du caractère s'il apparaît plusieurs fois dans la chaîne.
- Si vous souhaitez trouver toutes les occurrences d'un caractère dans une chaîne, vous devrez itérer sur la chaîne et utiliser `indexOf()` de manière répétée.

3. **Différences entre `indexOf()` et les crochets `[]`**

   Les **`crochets []`** sont utilisés pour accéder à des caractères spécifiques par leur index dans une chaîne, tandis que la méthode **`indexOf()`** est utilisée pour rechercher la première occurrence d'un caractère spécifique dans une chaîne et renvoyer son indice. Le choix entre les deux dépend de ce que vous voulez faire : accéder à un caractère spécifique ou rechercher un caractère dans la chaîne.

Quelques propriétés et méthodes de chaines de caractères en Dart

- En Dart, les chaînes de caractères sont représentées par la classe String. Voici quelques propriétés et méthodes couramment utilisées pour travailler avec des chaînes de caractères en Dart :

1. **`length`:**
   - Retourne la longueur de la chaîne de caractères.

```
String myString = "Bonjour";
int length = myString.length; // length vaudra 7
```

2. **`isEmpty`:**
   - Retourne true si la chaîne de caractères est vide, sinon false.

```
String emptyString = "";
bool isEmpty = emptyString.isEmpty; // isEmpty vaudra true
```

3. **isNotEmpty:**
   - Retourne true si la chaîne de caractères n'est pas vide, sinon false.

```
String nonEmptyString = "Hello";
bool isNotEmpty = nonEmptyString.isNotEmpty; // isNotEmpty vaudra true
```

4. **substring(start, end):**
   - Retourne une sous-chaîne de caractères commençant à l'indice start et se terminant à l'indice end. L'indice end est exclusif.

```
String originalString = "Hello, World!";
String subString = originalString.substring(0, 5); // subString vaudra "Hello"
```

5. **toUpperCase() et toLowerCase():**
   - Retournent une copie de la chaîne de caractères en majuscules ou en minuscules, respectivement.

```
String mixedCase = "ExAmPlE";
String upperCase = mixedCase.toUpperCase(); // upperCase vaudra "EXAMPLE"
String lowerCase = mixedCase.toLowerCase(); // lowerCase vaudra "example"
```

6. **trim(), trimLeft(), et trimRight():**
   - Suppriment les espaces inutiles au début, à la fin ou des deux côtés de la chaîne de caractères.

```
String stringWithWhitespace = "  Trim me!  ";
String trimmed = stringWithWhitespace.trim(); // trimmed vaudra "Trim me!"
```

apcpedagogie.com

7. **`split(delimiter)`:**
    - Divise la chaîne de caractères en une liste de sous-chaînes en utilisant le délimiteur spécifié.

```
String csv = "apple,banana,cherry";
List fruits = csv.split(','); // fruits vaudra ["apple", "banana", "cherry"]
```

8. **`contains(substring)`:**
    - Vérifie si la chaîne de caractères contient la sous-chaîne spécifiée.

```
String text = "Hello, World!";
```

Cours Flutter

Source : https://apcpedagogie.com/les-chaines-de-caracteres-en-langage-dart · téléchargé le 08/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com