Les types de données en langage Dart

1. Présentation

- Le langage Dart supporte plusieurs types de données qui peuvent être assignés à des variables.
  - Les chaînes de caractères(String)
  - Les nombres (int et double)
  - Les booléens (Vrai ou Faux / True ou False)
  - Les listes (Lists ou Arrays)
  - Les ensembles (Sets)
  - les dictionnaires (Maps)
- Dans ce tutoriel nous allons nous focaliser sur les trois premiers types de données.

2. Les chaines de caractères en langage Dart

1. Définition

- Une chaîne **Dart** est une séquence d'unités de code UTF 16. qui représente une séquence de caractères.
- Les valeurs de chaîne dans **Dart** peuvent être représentées à l'aide de guillemets simples, doubles ou triples.
  - Les chaînes à une seule ligne sont représentées à l'aide de guillemets simples ou doubles.
  - Les guillemets triples sont utilisés pour représenter des chaînes multilignes.

2. Syntaxe

```
String nom_variable = 'valeur' // représentation à l'aide des quotes simples
String nom_variable = "valeur" // représentation à l'aide des quotes doubles
String nom_variable = """ // représentation à l'aide des guillemets triples
String  variable_name = 'value'
OR
String  variable_name = ''value''
OR
String  variable_name = '''line1
line2'''
```

apcpedagogie.com

```
OR
String  variable_name= ''''''line1
line2''''''
```

3. **Exemple**

```
void main () {
   String s1 = 'chaîne sur une seule ligne';
   String s2 = "chaîne sur une seule ligne";
   String s3 = '''une chaîne
                  de caractères
                  sur plusieurs lignes ''';
   String s4 = """une autre chaîne
                  de caractères
                  sur plusieurs lignes""";

    print(s1);
    print(s2);
    print(s3);
    print(s4);
}
```

4. **Création de chaînes de caractères :**

   - En Dart, vous pouvez créer des chaînes de caractères en utilisant des guillemets simples ('), des guillemets doubles ("), ou des guillemets triples (""").

```
String chaine1 = 'Ceci est une chaîne avec des guillemets simples.';
String chaine2 = "Ceci est une chaîne avec des guillemets doubles.";
String chaine3 = """Ceci est une chaîne avec des guillemets triples.""";
```

5. **Longueur d'une chaîne de caractères :**

   - Pour obtenir la longueur d'une chaîne de caractères en Dart, vous pouvez utiliser la propriété 'length'.

```
String chaine = "Hello, World!";
int longueur = chaine.length;
```

apcpedagogie.com

6. **Accès aux caractères individuels :**
    - Vous pouvez accéder aux caractères individuels d'une chaîne de caractères en utilisant l'opérateur [] avec l'index du caractère.

```
String chaine = 'Hello';
String premierCaractere = chaine[0]; // 'H'
String deuxiemeCaractere = chaine[1]; // 'e'
```

7. **Recherche dans une chaîne de caractères :**
    - Dart fournit plusieurs méthodes pour rechercher des sous-chaînes dans une chaîne de caractères, telles que 'contains()', 'indexOf()', 'lastIndexOf()', etc.

```
String chaine = "Hello, World!";
bool contientWorld = chaine.contains("World"); // true
int indexVirgule = chaine.indexOf(","); // 5
```

8. **Modification de cas :**
    - Vous pouvez modifier le cas des caractères dans une chaîne de caractères en utilisant les méthodes 'toUpperCase()' et 'toLowerCase()'.

```
String chaine = 'Hello';
String premierCaractere = chaine[0]; // 'H'
String deuxiemeCaractere = chaine[1]; // 'e'
```

9. **Division d'une chaîne de caractères :**
    - Vous pouvez diviser une chaîne de caractères en plusieurs parties en utilisant la méthode 'split()'.

```
String chaine = "Bonjour, Monde!";
List mots = chaine.split(" "); // ['Bonjour,', 'Monde!']
```

10. **Concaténation de chaînes de caractères**

apcpedagogie.com

- Le processus de création d'une nouvelle chaîne en ajoutant une valeur à une chaîne statique est appelé concaténation ou interpolation.
- Concaténation de chaînes de caractères, c'est le processus d'ajout d'une chaîne à une autre chaîne.
- L'opérateur plus **(+)** est un mécanisme couramment utilisé pour concaténer/interpoler des chaînes.

```
void main() {
   String str1 = "hello";
   String str2 = "world";
   String res = str1+ " " +str2;

   print("La chaîne concaténée est : $res");
}
```

11. Vous pouvez utiliser "**`${}`**" pour afficher la valeur d'une variable chaine de caractères Dart au sein d'une chaine **Dart**.

12. **Dart – Propriété de chaîne `codeUnits`**

    - La propriété de chaîne `codeUnits` dans le langage de programmation Dart renvoyait la liste du code UTF-16 des caractères d'une chaîne. Il s'agit de propriétés de chaîne très utiles pour trouver directement les unités de code UTF-16.

13. **Exemple**

```
void main() {
  // initialize string st
  String st = "apcpedagogie";
  // print the UTF-16 code
  print(st.codeUnits);
}
```

14. **Sortie**

```
[97, 112, 99, 112, 101, 100, 97, 103, 111, 103, 105, 101]
```

3. **Dart – Propriété de chaîne `isEmpty`**

apcpedagogie.com

- La propriété de chaîne `isEmpty` renvoie **vrai** si la chaîne est vide ; sinon renvoie **faux**.

4. **Exemple**

```
void main() {
  String str = "Hello";
   print(str.isEmpty);
}
```

5. **Sortie**

```
Faux
```

- **Dart – Propriété de chaîne length**

  - La propriété de chaîne `length` renvoie la longueur de la chaîne, y compris les espaces, les tabulations et les sauts de ligne.

- **Exemple**

```
String str = "Hello All";
   print("La longueur de la chaîne est : ${str.length}");
```

- **Sortie**

```
La longueur de la chaîne est : 9
```

- **Méthodes pour manipuler les chaînes**

  - **1. contains()**
    - Cela vous permet de vérifier si la chaîne spécifiée existe :
      `str1.contains('rem'); // true`
  - **2. startsWith()**

apcpedagogie.com

- Cela vous permet de vérifier si la chaîne commence par les caractères spécifiés :

```
str2.startsWith('Lorem'); // true str3.startsWith('Noorem'); // false
```

- **3. endsWith ()**
  - Vérifie si la chaîne se termine par les caractères spécifiés :

```
str3.endsWith('ipsum'); // true str3.endsWith('oopsum'); // false
```

- **4. toLowerCase(), toUpperCase()**
  - Convertit la chaîne aux formats minuscules et majuscules :

```
str1.toLowerCase(); // lorem str1.toUpperCase(); // LOREM
```

- **5. split()**
  - Fractionne la chaîne au motif correspondant, renvoyant une liste de sous-chaînes :

```
str3.split('\n'); // ['Multi', 'Line', 'Lorem Lorem ipsum'];
```

- **6. splitMapJoin()**
  - Divise la chaîne, convertit chaque élément de la liste et combine ensuite en une nouvelle chaîne :

```
str3.splitMapJoin(RegExp(r'^', multiLine: true), // match line start onMatch: (m) =>
'**${m.group(0)} ', // add asterisk to line start onNonMatch: (n) => n); // leaves non
matches as is /* Output: ** Multi ** Line ** Lorem Lorem ipsum */
```

- **7. indexOf(), dernierIndexOf()**
  - Renvoie la position des première et dernière correspondances du modèle donné :

```
str3.indexOf('rem'); // 13 str3.lastIndexOf('rem'); // 19
```

  - Les deux méthodes acceptent également un paramètre facultatif spécifiant l'index à partir duquel commencer la recherche :

```
str3.lastIndexOf('rem', 18); // 13
```

- **8. trim()**
  - Supprime les espaces blancs de début et de fin

```
" $str2 ".trim(); // 'Lorem ipsum'
```

- **9. padLeft(), padRight()**
  - Remplit la chaîne à gauche et à droite avec le remplissage donné si la chaîne est inférieure à la longueur spécifiée :

```
str1.padLeft(8, 'x'); // xxLorem str1.padRight(8, 'x'); // Loremxx
```

- **10. replaceAll()**
  - Remplace toutes les sous-chaînes qui correspondent au modèle spécifié par la chaîne de remplacement :

```
str2.replaceAll('e', 'é'); // Lorém
```

- **11. compareTo()**
  - Renvoie une nouvelle chaîne en supprimant tous les espaces de début et de fin. Cependant, cette méthode ne supprime pas les espaces entre deux chaînes.

apcpedagogie.com

- Renvoie un entier représentant la relation entre deux chaînes.
  - **0** - lorsque les chaînes sont égales.
  - **1** - lorsque la première chaîne est supérieure à la seconde
  - **-1** - lorsque la première chaîne est plus petite que la seconde

```
String str1 = "A"; String str2 = "A"; String str3 = "B"; print("str1.compareTo(str2): ${str1.compareTo(str2)}"); // str1.compareTo(str2): 0 print("str1.compareTo(str3): ${str1.compareTo(str3)}"); // str1.compareTo(str3): -1 print("str3.compareTo(str2): ${str3.compareTo(str2)}"); // str3.compareTo(str2): 1
```

- **12.substring()**
  - Renvoie la sous-chaîne de cette chaîne qui s'étend de **startIndex**, inclus, à **endIndex**, exclusif.
  - **Syntaxe**: substring(int startIndex, [int endIndex])
  - **Paramètres**
    - **startIndex** - l'index à partir duquel commencer l'extraction (inclusif).
    - **endIndex** - l'index pour arrêter l'extraction (exclusif).
  - **Remarque** − Les index sont basés sur zéro, c'est-à-dire que le premier caractère aura l'index 0 et ainsi de suite.

- **Exemple**

```
void main() {
   String str1 = "Hello World";
   print("New String: ${str1.substring(6)}");
   // from index 6 to the last index
   print("New String: ${str1.substring(2,6)}");
   // from index 2 to the 6th index
}
```

Les nombres

- Les nombres , dans Dart, peuvent être de 2 types :

- **`int`**

- `int`: Integer est un type de valeur numérique non fractionnaire sans. Ici non fractionnaire signifie une valeur sans point. La dose entière ne prend pas en charge les valeurs POINT. Il ne peut contenir que des valeurs numériques pures. Voir l'exemple ci-dessous.

apcpedagogie.com

```
int tempNum = 123456;
print(tempNum);
// Output should be : 123456
int tempNum = 123456;
print(tempNum);
// Output should be : 123456
```

- **double**
    - `double` : Double sont essentiellement des valeurs FLOAT plus grandes. Il peut contenir des valeurs décimales fractionnaires. Dans Dart, le double prend en charge les valeurs de prescription double 64 bits. double représente également des littéraux à virgule flottante.

```
double i = 52.11 ;
 print(i);
 // Output should be 52.11
 double i = 52.11 ;
 print(i);
 // Output should be 52.11
```

- ## Booleen
    - Le type de données booléen est utilisé pour contenir les valeurs vraies et fausses. Le type de données booléen utilise le mot-clé ' bool ' au moment de la déclaration.

```
bool val1 = true;
bool val2 = false;
print(val1);
print (val2);
// Output should be true & false
bool val1 = true;
bool val2 = false;
print(val1);
print (val2);
// Output should be true & false
```

apcpedagogie.com

- ## Applications

  - **App01**

    - **1-** Créer un programme dart qui demande à l'utilisateur de saisir une couleur qui sera stockée dans une variable appelée : **color**
    - **2-** Sur la ligne suivante, créer une variable String appelée '**favColor**' et affectez-la à 'Ma couleur préférée est le rouge', lorsque l'utilisateur à saisi rouge
    - **3-** Afficher le contenu du variable '**favColor**'

      Solution

```
import 'dart:io';

void main() {
  print("Bonjour, veuillez saisir votre couleur préférée :");
  String? color = stdin.readLineSync();
  String? favColor = "Ma couleur préférée est le $color";
  print(favColor);
}
```

  - **App02**

    - **1-** Pour cet exercice, on vous demande de créer un programme Dart
    - **2-** Le programme demande à l'utilisateur de saisir un montant dans une variable entière appelée : '**montantFacture**'
    - **3-** Sur la ligne suivante, créez une variable String '**informationFacture**' qui stocke le message du type : '**Votre montant de facture est : 250**', lorsque '**montantFacture**' est 250
    - **4-** Afficher le contenu du variable '**informationFacture**'

      Solution

```
import 'dart:io';

void main() {
  print("Bonjour, veuillez saisir le montant de la facture :");
  double? montantFacture = double.parse(stdin.readLineSync()!);
  String? informationFacture = "Votre montant de facture est : $montantFacture";
```

apcpedagogie.com

```
  print(informationFacture);
}
```

- **App03**
  - **1-** Créez un programme Dart sous VSCode
  - **2-** On vous donne une variable de type **int** appelée **anneeNaissance**, une variable de type **String** appelée **name** et une variable de type int appelée **anneeAujourdhui**
  - **3-** Sur la ligne suivante, créez une variable de type String appelée '**profileInformation**' qui est affectée à 'Mohamed a 26 ans' si l'utilisateur a saisie Mohamed comme name et 1996 comme année de naissance et 2022 est l'année en cour (2022 - 1996 = 26)
    - *Pour récupérer l'année en cours, Utilisez la fonction prédéfinie* `new DateTime.now().year.`

- **App04**
  - On vous donne une variable entière appelée **minutes**
  - Sur la ligne suivante, créer une variable appelée '**countdownHours**' et affectez-la à la valeur que vous obtenez après avoir converti les minutes en heures
  - Sur la ligne suivante, créer une variable appelée '**countdownMinutes**' et affectez-la aux minutes restantes
  - Créer une variable String '**launchInfo**' qui stocke : '**Lancement de filme en 2 heures 15 minutes**' lorsque '**countdownHours**' vaut 2 et '**countdownMinutes**' vaut 15

Solution

```
import 'dart:io';

void main() {
  print("Bonjour, le nombre des minutes.");
  int? minutes = int.parse(stdin.readLineSync()!);
  int? countdownHours = (minutes ~/ 60);
  int? countdownMinutes = (minutes % 60);

  print("Lancement de film en $countdownHours h $countdownMinutes minutes");
}
```

- **App05**

apcpedagogie.com

- Écrire un programme en Dart qui affiche les lettres minuscules de langue française sur la même ligne.
  Sur la deuxième ligne le programme doit afficher les lettres minuscules de langue française sur la même ligne.
  Solution

```
import 'dart:io';

void main() {
  stdout.write('Les lettres minuscules de la langue française\n');
  print(
      '--------------------------------------------------------------------');
  for (int i = 97; i < 97 + 26; i++) {
    stdout.write("${String.fromCharCode(i)} ");
  }
  stdout.write('\n\nLes lettres majuscules de la langue française\n');
  print(
      '--------------------------------------------------------------------');
  for (int i = 65; i < 65 + 26; i++) {
    stdout.write("${String.fromCharCode(i)} ");
  }
}
```

Cours Flutter

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