Les fonctions dans Dart

1. Présentation

- Chaque semaine, il y a des tâches que vous répétez encore et encore : prendre votre petit-déjeuner, vous brosser les dents, écrire votre nom, lire des livres sur Dart, etc. Chacune de ces tâches peut être divisée en tâches plus petites. Se brosser les dents, par exemple, consiste à mettre du dentifrice sur la brosse, à brosser chaque dent et à se rincer la bouche avec de l'eau.
- La même idée existe en programmation informatique. Une fonction est une petite tâche, ou parfois un ensemble de plusieurs petites tâches connexes que vous pouvez utiliser conjointement avec d'autres fonctions pour accomplir une tâche plus importante.
- En programmation, une fonction ou une méthode est un ensemble de codes qui exécute une tâche spécifique.
- Les fonctions **Dart**, sont des objets de type `Function`. Cela signifie que les fonctions peuvent être affectées à des variables ou transmises comme arguments à d'autres fonctions.
- Dans ce tutoriel, vous apprendrez à écrire des fonctions dans Dart, y compris des fonctions nommées et des fonctions anonymes.

2. Définition d'une fonction Dart

- La fonction est un ensemble d'instructions groupées logiquement qui exécutent une tâche spécifique. Dans le programme Dart, une fonction est créée pour réaliser quelque chose.
- Chaque programme Dart a au moins une fonction c'est-à-dire `main()` où l'exécution du programme commence. C'est une fonction obligatoire dans Dart.
- Une fonction est utilisée pour :
  - Eviter la répétition des codes.
  - Augmenter la lisibilité du programme.
  - Diviser un problème complexe en problèmes plus simples.
  - Réduire les risques d'erreur.

> - *La fonction est définie avant **main()***

apcpedagogie.com

3. ## Fonctions de base en Dart

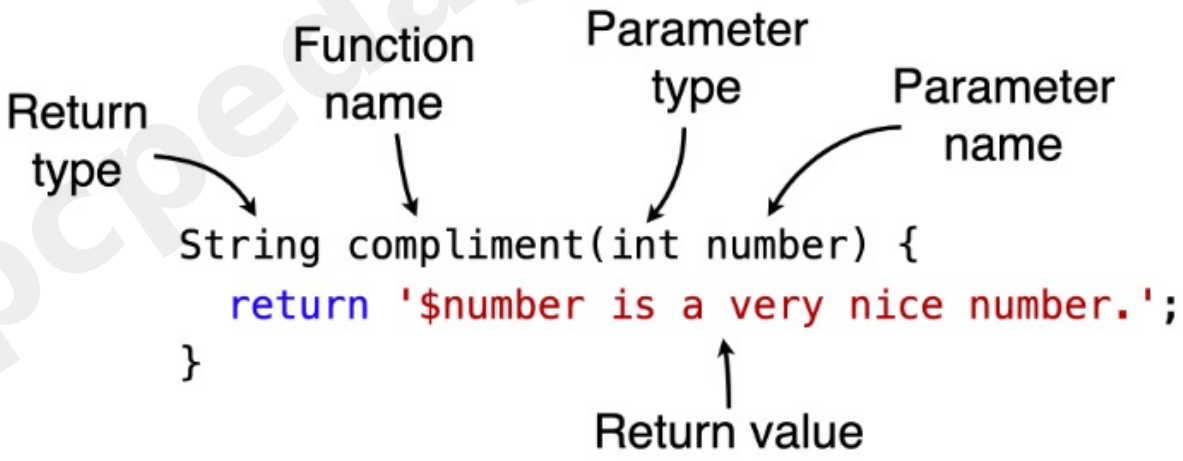
   - Vous pouvez penser à des fonctions comme des machines ; ils prennent quelque chose que vous leur fournissez (l'intrant) et produisez quelque chose de différent (la sortie).



   - Il existe de nombreux exemples de cela dans la vie quotidienne. Par exemple:
     - Avec une pomme presse-agrumes, vous mettez des pommes et vous sortez du jus de pomme, l'entrée est des pommes, la sortie est du jus.
     - Pour un lave-vaisselle l'entrée est de la vaisselle sale et la sortie est propre vaisselle.
     - Blenders, cafetières, micro-ondes et fours sont tout comme les fonctions du monde réel qui acceptent une entrée et produire une sortie.

4. ## Syntaxe d'une fonction `Dart`

   - Dans Dart, une fonction se compose d'un **type de retour**, d'un **nom**, d'**un liste de paramètres** entre **parenthèses** et un **corps** entre **accolades**.



   - En Dart, une fonction se compose généralement des éléments suivants :

     1. **Type de retour (Return type).**

- Une fonction peut renvoyer une valeur de n'importe quel type, y compris void si elle ne renvoie aucune valeur. Le type de retour est spécifié avant le nom de la fonction.
- Cette fonction particulière renverra une chaîne, mais vos fonctions peuvent renvoyer n'importe quel type que vous aimez.
- Si la fonction ne renverra rien, c'est-à-dire s'il effectue un travail mais ne produit pas de valeur de sortie, vous pouvez utiliser void comme type de retour.

```
void main() {
  const input = 12;
  final output = compliment(input);
  print(output);
}

String compliment(int number) {
  return '$number is a very nice number.';
}
```

2. **Nom de la fonction (Function name).**

- C'est un identificateur qui donne un nom à la fonction. Le nom doit être unique dans le contexte où il est défini.
- vous pouvez presque nommer les fonctions tout ce que vous voulez, mais vous devez suivre les convention de dénomination **`lowerCamelCase`**. Vous apprendrez quelques

3. **Paramètres d'une fonction Dart (Parameters).**

- Les paramètres sont les valeurs que la fonction accepte en entrée. Ils sont placés entre parenthèses après le nom de la fonction. La liste de paramètres contient les noms des paramètres et leurs types.
- Cet exemple n'a qu'un seul paramètre, mais si vous aviez plus d'un, vous les sépareriez par des virgules.
- Pour chaque paramètre, vous écrivez d'abord le type, suivi d'un nom. Tout comme avec les noms de variables, vous devez utiliser **`lowerCamelCase`** pour vos noms de paramètres.

4. **Valeur de retour d'une fonction Dart (Return value).**

apcpedagogie.com

- C'est la sortie de la fonction, et elle doit correspondre au type de retour. Dans l'exemple ci-dessus, la fonction renvoie une valeur **String** en utilisant le retour mot-clé. Si le type de retour est **void**, alors vous ne retourne rien.

- **Activité 01**

  - Écrivez un programme en Dart qui demande à l'utilisateur de saisir deux notes. Le programme doit calculer la moyenne des deux notes et afficher le résultat. Assurez-vous de gérer les cas où l'utilisateur ne saisit pas des valeurs valides (des nombres).
  - **Contraintes :**
    - Utilisez une fonction appelée "**calculerMoyenne**" qui prend les deux notes en tant que paramètres et renvoie la moyenne sous forme de chaîne de caractères.
    - Utilisez une boucle pour redemander à l'utilisateur de saisir les notes si les valeurs saisies ne sont pas valides.
  - **Exemple d'exécution du programme :**

```
Entrez la première note : abc
Veuillez saisir des nombres valides.

Entrez la première note : 15.5
Entrez la deuxième note : 18.0
La moyenne est : 16.75
```

  Solution

```
import 'dart:io';

String calculerMoyenne(double note1,double note2){
  double moyenne=(note1+note2)/2;
  return moyenne.toString();

}
void main(){
  double nombre1=0;
  double nombre2=0;
  bool saisieValide=false;
  while(!saisieValide){
    stdout.write("Entrez la première note");
```

apcpedagogie.com

```
    String? input1=stdin.readLineSync();
    stdout.write("Entrez la deuxième note");
    String? input2=stdin.readLineSync();
    if(input1!=null && input2!=null){
      if(double.tryParse(input1)!=null && double.tryParse(input2)!=null){
      nombre1=double.parse(input1);
      nombre2=double.parse(input2);
      String moyenne=calculerMoyenne(nombre1, nombre2).toString();
      print('La moyenne est $moyenne');
      saisieValide=true;
    }else{
      print("Veuillez saisir des nombres valides");
    }
    }else{
      print("Veuillez saisir des nombres valides");
    }
  }
}
```

- **Activité 02**

  - Demander à l'utilisateur de saisir trois nombres.
  - Écrire une fonction **trouverMax** qui prend ces trois nombres en paramètres et retourne le plus grand.
  - Afficher le résultat dans la console.

    Solution

```
import 'dart:io';

// Fonction pour trouver le plus grand parmi trois nombres
double trouverMax(double a, double b, double c) {
  return (a > b) ? ((a > c) ? a : c) : ((b > c) ? b : c);
}

void main() {
  // Demander à l'utilisateur de saisir trois nombres
  stdout.write("Entrez le premier nombre : ");
  double? nombre1 = double.tryParse(stdin.readLineSync()!);

  stdout.write("Entrez le deuxième nombre : ");
  double? nombre2 = double.tryParse(stdin.readLineSync()!);
```

apcpedagogie.com

```
  stdout.write("Entrez le troisième nombre : ");
  double? nombre3 = double.tryParse(stdin.readLineSync()!);

  // Vérification des entrées
  if (nombre1 == null || nombre2 == null || nombre3 == null) {
    print("Veuillez saisir des nombres valides !");
  } else {
    // Appel de la fonction et affichage du plus grand nombre
    double max = trouverMax(nombre1, nombre2, nombre3);
    print("Le plus grand nombre est : $max");
  }
}
```

- **Activité 03**

  - Écrivez un programme en Dart qui permet à l'utilisateur de calculer le montant d'un pourboire à la suite d'un repas. Le programme doit demander à l'utilisateur le montant total de l'addition et le pourcentage de pourboire qu'il souhaite laisser. Le programme doit ensuite calculer le montant du pourboire et l'afficher, ainsi que le total de l'addition incluant le pourboire.
  - **Contraintes :**
    - Utilisez une fonction appelée "**calculerPourboire**" qui prend le montant total de l'addition et le pourcentage de pourboire en tant que paramètres, et renvoie le montant du pourboire.
    - Assurez-vous de gérer les cas où l'utilisateur ne saisit pas des valeurs valides (des nombres)en utilisant un bloc try-catch .
  - **Exemple d'exécution du programme :**

    ```
    Montant total de l'addition : 75.50
    Pourcentage de pourboire (%) : 15

    Montant du pourboire : 11.33
    Total de l'addition incluant le pourboire : 86.83
    ```

    Solution

apcpedagogie.com

```
import 'dart:io';

double calculerPourboire(double montant, double pourcentage) {
  return montant * pourcentage / 100;
}

void main() {
  double montantTotal;
  double pourcentagePourboire;

  while (true) {
    stdout.write('Montant total de l'addition : ');
    String? montantInput = stdin.readLineSync();

    stdout.write('Pourcentage de pourboire (%) : ');
    String? pourcentageInput = stdin.readLineSync();

    try {
      montantTotal = double.parse(montantInput!);
      pourcentagePourboire = double.parse(pourcentageInput!);
      break;
    } catch (e) {
      print('Veuillez saisir des nombres valides.');
    }
  }

  double pourboire = calculerPourboire(montantTotal, pourcentagePourboire);
  double total = montantTotal + pourboire;

  print('Montant du pourboire : ${pourboire.toStringAsFixed(2)}');
  print('Total de l'addition incluant le pourboire :
${total.toStringAsFixed(2)}');
}
```

5. **Types des fonctions**

   - Dart offre une grande flexibilité pour écrire des fonctions adaptées aux besoins de ton programme !

     Fonction simple avec retour (return) Fonction void (sans retour) Paramètres optionnels [] Paramètres nommés {} Fonction anonyme (Lambda) Fonction récursive Fonction asynchrone (async et await)

apcpedagogie.com

1. **Fonctions paramétrées**

   - Ceux-ci sont des fonctions qui, à leur tour, peuvent accepter des valeurs au moment de leur appel dans le programme.
   - Ces valeurs sont appelées paramètres et elles sont passées à la fonction lors de l'appel de celle-ci.
   - Si il n'y a pas d'indication contraire que le programmeur ajoute selon ses attentes, le nombre des valeurs à entrer est obligatoirement égal au nombre des paramètres de la fonction.
   - **La syntaxe est la suivante :**

```
//Le nombre des paramètres n'est pas limité
type_de_retour nom_fonction(type_paramètre1 nom_paramètre1, type_paramètre2
nom_paramètre2, ...){
     //instructions
     return valeur //si il y en a
}
```

2. **Fonctions fléchées**

   - Il y a des fonctions dit fonctions fléchées, elles permettent une syntaxe plus concise pour les fonctions simples, utilisant `=>`.
   - A noter qu'elles ne sont pas capables de tout comme les fonctions normales, elles sont utilisées pour des fonctions n'ayant pas plusieurs instructions à réaliser.
   - Dans notre exemple précédent, prenons le cas de la fonction d'affichage, elle peut s'écrire aussi de la manière suivante :

```
void afficherMessage(String message) => print(message);
```

6. Les types de paramètres dans les fonctions Dart

- Les paramètres sont classés en deux types dans Dart : **obligatoires** et **optionnels**. Puis, les paramètres optionnels sont divisés en **nommés**, **positionnels** ou **par défaut**.

apcpedagogie.com



1. **Les paramètres obligatoires**

    - Ici, il faut passer des arguments pour tous les paramètres sinon on ne pourra pas exécuter la fonction, elle renverra une erreur.

2. **Paramètres optionnels ou facultatifs**

    - Les paramètres optionnels peuvent être soit positionné ou nommé, mais pas les deux en même temps.
    - Les deux peuvent avoir une valeur par défaut. Elle doit être une constante de compilation telle qu'une valeur littérale. Si aucune valeur par défaut n'est fournie, la valeur est null.
    - On, peut passer ou ignorer l'argument pour le paramètre qui est entouré de crochets `[]`.
    - Si l'argument est ignoré pour le paramètre, il prendra la valeur **NULL**.

    1. **Exemples: Utilisation du paramètre positionnel**

```
void main() {
  pays('France', 'Allemagne');
}
void pays(String name1, String name2, [String? name3]) {
  print("J'habite en $name1");
  print("J'habite en $name2");
  print("J'habite en $name3");
}
//Sortie
J'habite en France
J'habite en Allemagne
J'habite en null
```

    Ou

```
void main() {
  afficherDetails("votre_nom", "votre_adresse", "21677777777");
  print('\n');
  afficherDetails("votre_nom", "votre_adresse");
}

//Paramètres positionnels facultatifs
void afficherDetails(String nom, String adresse, [String? numTelephone]) {
  print("votre nom est: $nom");
  print("votre adresse est: $adresse");
  print(
      "votre numéro de téléphone est: ${numTelephone
?? "Pas de numéro de téléphone"}");
}
Sortie
votre nom est: votre_nom
votre adresse est: votre_adresse
votre numéro de téléphone est: 21677777777


votre nom est: votre_nom
votre adresse est: votre_adresse
votre numéro de téléphone est: Pas de numéro de téléphone
```

2. **Paramètres positionnels**

   - Si nous encapsulons un ensemble de **paramètres de fonction** dans `[]`, nous les marquons en tant que **paramètres positionnels facultatifs**.
   - On utilise des crochets `[]` pour spécifier des paramètres optionnels de position dans Dart. En fait, il faut savoir quelle position correspond à quel paramètre.
   - Dans les paramètres positionnels, vous devez fournir les arguments dans **le même ordre** que celui que vous avez défini sur les paramètres lorsque vous avez écrit la fonction. Si vous appelez la fonction avec le paramètre dans le mauvais ordre, vous obtiendrez un mauvais résultat.
   - **Exemple 1 : Utilisation du paramètre positionnel**
     - Dans l'exemple ci-dessous, la fonction **printInfo** prend deux paramètres. Vous devez passer le nom et le sexe de la personne dans le même ordre. Si vous passez des valeurs dans le mauvais ordre, vous

apcpedagogie.com

obtiendrez un mauvais résultat .

```
void printInfo(String name, String genre) {
  print("Bonjour $name votre sexe est $genre.");
}

void main() {
  // transmettre des valeurs dans le mauvais ordre
  printInfo("Homme", "Mohamed");

  // transmettre les valeurs dans le bon ordre
  printInfo("Mohamed", "Homme");
}
//Sortie
Bonjour Homme votre sexe est Mohamed.
Bonjour Mohamed votre sexe est Homme.
```

3. **Exemple 2 : fournir une valeur par défaut pour le paramètre de position**
   - Dans l'exemple ci-dessous, la fonction **printInfo** prend deux paramètres positionnels et un paramètre optionnel. Le paramètre de titre est facultatif ici. Si l'utilisateur ne transmet pas le titre, il définira automatiquement la valeur du titre sur Monsieur.

```
void printInfo(String nom, String genre, [String titre = "Monsieur"]) {
  print("Bonjour $titre $nom votre genre est $genre.");
}
void main() {
  printInfo("Mohamed", "Homme");
  printInfo("Mohamed", "Homme", "Mr.");
  printInfo("Eve", "Female", "Ms.");
}
//Sortie
Bonjour Monsieur Mohamed votre genre est Homme.
Bonjour Mr. Mohamed votre genre est Homme.
Bonjour Ms. Eve votre genre est Female.
```

4. **Exemple 3 : fournir une valeur par défaut pour le paramètre de position**
   - Dans l'exemple ci-dessous, la fonction **add** prend **deux paramètres positionnels** et un paramètre **optionnel**. Le paramètre **nbr3** est facultatif ici avec la valeur par défaut 0 .

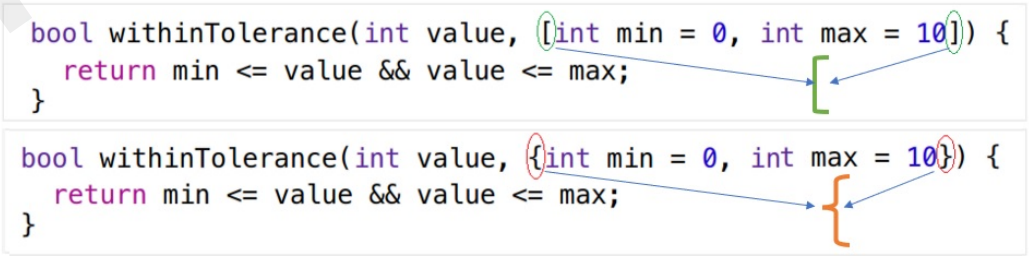
apcpedagogie.com

```
 void add(int nbr1, int nbr2, [int nbr3=13]){
   int somme;
  somme = nbr1 + nbr2 + nbr3;

   print("La somme est $somme");
}
void main(){
  add(10, 20);
  add(10, 20, 30);
}
//Sortie
La somme est 43
La somme est 60
```

5. **Paramètres nommés**

- Les **`accolades {}`** sont utilisées pour les paramètres optionnels nommés dans Dart.
- Nous définissons les paramètres nommés en les listant entre **`accolades {}`**, par exemple, `{int a, int b, …}`.
- Dart vous permet d'utiliser des paramètres nommés pour rendre le signification des paramètres plus claire dans les appels de fonction.
- Pour créer un paramètre nommé, vous l'entourez de **boucles accolade**s au lieu de **crochets**.
- Voici la même fonction comme ci-dessus, mais en utilisant à la place des **paramètres nommés**:



Notez ce qui suit :

- `min` et `max` sont entourés d'accolades, ce qui signifie que vous devez utiliser les noms de paramètre lorsque vous fournissez leur valeurs d'argument à la fonction.

apcpedagogie.com

- Comme les **crochets**, les **accolades** rendent les paramètres à l'intérieur en option. Puisque la valeur n'est pas à l'intérieur des accolades, cependant, c'est toujours nécessaire.

> *Les **`accolades {}`** sont utilisées pour les paramètres optionnels nommés dans Dart. Comme les paramètres nommés sont référencés par leur nom, ils peuvent donc être utilisés lors de l'appel dans un ordre différent de celui de la déclaration.*

- **Exemple 1 : Utilisation du paramètre nommé**
  - Dans l'exemple ci-dessous, la fonction **printInfo** prend **deux paramètres nommés**. Vous pouvez transmettre la valeur dans n'importe quel ordre.

```
void printInfo({String? nom, String? genre}) {
  print("Bonjour $nom votre genre est $genre.");
}
void main() {
  // vous pouvez passer des valeurs dans n'importe
  //quel ordre dans les paramètres nommés.
  printInfo(genre: "Homme", nom: "Mohamed");
  printInfo(nom: "Sarra", genre: "Femme");
  printInfo(nom: "Rima", genre: "Femme");
  printInfo(nom: "Rima", genre: "Femme");
  printInfo(nom: "Habib", genre: "Homme");
  printInfo(genre: "Homme", nom: "Sarra");
}
//Sortie
Bonjour Mohamed votre genre est Homme.
Bonjour Sarra votre genre est Femme.
Bonjour Rima votre genre est Femme.
Bonjour Rima votre genre est Femme.
Bonjour Habib votre genre est Homme.
Bonjour Sarra votre genre est Homme.
```

6. **Exemple 2 : Utilisation de `Required` dans le paramètre nommé**
   - Dans l'exemple ci-dessous, la fonction **printInfo** prend deux paramètres nommés. Vous pouvez voir un mot-clé obligatoire , ce qui signifie que vous devez transmettre le nom et le genre de la personne. Si vous ne le passez pas, cela ne fonctionnera pas.

apcpedagogie.com

```
void printInfo({required String nom, required String genre}) {
  print("Bonjour $nom votre genre est $genre.");
}

void main() {
  // vous pouvez passer des valeurs dans n'importe
  //quel ordre dans les paramètres nommés.
  printInfo(genre: "Homme", nom: "Mohamed");
  printInfo(genre: "Femme", nom: "Sarra");
}
//Sortie
Bonjour Mohamed votre genre est Homme.
Bonjour Sarra votre genre est Femme.
```

7. Applications

- **App01**
  - Écrire un programme Dart qui demande à l'utilisateur de saisir deux nombres positifs.
  - Écrire une fonction somme qui renvoie la somme des deux nombres saisies.
    Solution

```
import 'dart:io';

void main() {
  // Demande à l'utilisateur de saisir les deux nombres
  stdout.write('Entrez le premier nombre positif : ');
  double premierNombre = double.parse(stdin.readLineSync());

  stdout.write('Entrez le deuxième nombre positif : ');
  double deuxiemeNombre = double.parse(stdin.readLineSync());

  // Vérification que les nombres sont positifs
  if (premierNombre < 0 || deuxiemeNombre < 0) {
    print('Les nombres doivent être positifs.');
  } else {
    // Appel de la fonction somme pour calculer la somme
```

apcpedagogie.com

```
    double resultat = somme(premierNombre, deuxiemeNombre);
    print('La somme de $premierNombre et $deuxiemeNombre est :
$resultat');
  }
}

double somme(double nombre1, double nombre2) {
  return nombre1 + nombre2;
}
```

- **App02**
  - Écrire un programme Dart qui demande à l'utilisateur de saisir une chaîne de caractère.
  - Écrire une fonction qui vérifie si une chaîne passée est palindrome ou non ?
  - **Remarque** : Un palindrome est un mot, une phrase ou une séquence qui se lit de la même manière vers l'arrière que vers l'avant, par exemple, madame ou nursesrun.

    Solution

```
import 'dart:io';
void main() {
  print("Veuillez saisir une chaîne de caractère");
  String? chaine = stdin.readLineSync()!;
  bool resultat = palindrome(chaine);
  if (resultat == true) {
    print("La chaîne $chaine est un plaindrome");
  } else {
    print("La chaîne $chaine n'est pas un plaindrome");
  }
}
bool palindrome(String txt) {
  for (int index = 0; index < txt.length ~/ 2; index++) {
    if (txt[index] != txt[txt.length - index - 1]) return false;
  }
  return true;
}
```

apcpedagogie.com

- **App03**
  - Écrire un programme Dart qui demande à l'utilisateur de saisir un mot. Selon que le mot est de type String affiche à l'utilisateur "le mot saisie est une chaine de caractère, si le mot est de type `int` affiche le nombre est pair ou impair.
    Solution

Cours Flutter

Source : https://apcpedagogie.com/les-fonctions-dans-dart · téléchargé le 08/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com