Fiche de Référence : Explorer le Langage Dart

Variables et types

Type	Description	Exemple de code
Var	Variable de type inféré	`var name = 'Dart';`
Dynamique	Les types de variables peuvent changer dynamiquement	`dynamic x = 42;`
Final	Constant à l'exécution	`final cityName = 'New York';`
Const	Constante à la compilation	`const PI = 3.14;`

Fonctions et méthodes

Type	Description	Exemple de code
Fonction	Définit une fonction qui additionne deux nombres	`int add(int a, int b) => a + b;`
Syntaxe des flèches	Utilise la syntaxe des flèches pour une déclaration de fonction concise	`void printItem(item) => print(item);`
Paramètres requis	Doivent être fournis explicitement dans l'appel de fonction	`int multiply(int a, int b) => a * b;`
Paramètres positionnels optionnels	Peuvent être omis ; entourés de crochets	`String fullName(String firstName, [String middleName, String lastName])`
Paramètres nommés	Spécifiés par nom ; peuvent être requis ou optionnels avec des valeurs par défaut, entourés d'accolades	`void greet({required String name, String greeting = 'Hello'}) => print('$greeting, $name!');`
Paramètres par défaut	Permet des valeurs par défaut si elles ne sont pas fournies dans l'appel de fonction	`String describe(String name, {int age = 30, String city = 'Unknown'})`
Fermetures	Fonctions anonymes qui peuvent capturer des variables de leur contexte	`List<int> numbers = [1, 2, 3]; numbers.forEach((number) => print(number * 2));`

Classes et OOP

Type	Description	Exemple de code
Classe	Définit une classe simple avec des propriétés	`class Person { String name; int age; }`
Héritage	Démontre l'héritage de base en Dart	`class Employee extends Person { int salary; }`
Encapsulation	Utilise les méthodes de classe et la visibilité pour imposer l'encapsulation	`class Person { String name; // Public property int _age; // Private property, underscore prefix in Dart }`
Propriétés publiques	Peuvent être accessibles depuis n'importe quel endroit où l'objet est visible	`class Person { String name; // Public property }`
Propriétés privées	Préfixées par un underscore et ne peuvent être accessibles que dans la classe	`class Person { int _age; // Private property }`
Accesseurs et Mutateurs	Contrôler l'accès aux propriétés de la classe	`class Person { int _age; int get age => _age; // Getter set age(int value) { // Setter _age = value; } }`
Méthodes statiques	Appartiennent à la classe plutôt qu'à une instance de la classe et peuvent être appelées sans un objet	`class Utility { static int add(int a, int b) { return a + b; }`

		`}`

Fonctions anonymes	Utilisées pour des fonctions à expression unique ; également connues sous le nom de lambdas ou de fermetures	`var list = ['apples', 'bananas', 'oranges'];`
		`list.forEach((item) {`
		`  print(item);`
		`});`

Structures de données courantes

Type	Description	Exemple de code
Liste	Collection ordonnée d'éléments	`List<int> numbers = [1, 2, 3];`
Carte	Collection de paires clé-valeur	`Map<String, int> ages = {'Alice': 18, 'Bob': 20};`
Ensemble	Collection non ordonnée d'éléments uniques	`Set<String> names = {'Alice', 'Bob'};`
Files d'attente	Collection FIFO pour les éléments	`Queue<int> queue = Queue(); queue.addAll([1, 2, 3]);`
Listes chaînées	Une séquence d'éléments où chaque élément pointe vers le suivant	`LinkedList<int> linkedList = LinkedList(); linkedList.add(1);`

Bibliothèques et utilitaires en ligne de commande

Type	Description	Exemple de code
Importation	Accéder aux bibliothèques intégrées de Dart	`import 'dart:math';`
Commandes CLI	Compiler le code Dart en exécutable natif	`dart compile exe test.dart`
Dart SDK	Outil essentiel pour exécuter et gérer des applications Dart	`dart run, dart create`
Outil Pub	Gestionnaire de paquets pour gérer les dépendances	`dart pub get, dart pub add http`
Dart DevTools	Ensemble pour le débogage et le profilage des performances	`Performance profiling, memory analysis, and widget inspection`
dart:core	Classes et fonctions fondamentales	`Handling strings, numbers, collections like List and Map.`
dart:math	Fournit des constantes et des fonctions mathématiques	`sin, cos, sqrt, and constants like pi.`
dart:async	Prend en charge la programmation asynchrone	`Future, Stream for handling asynchronous operations.`
dart:convert	Gestion des JSON, encodage/décodage UTF-8	`jsonEncode, jsonDecode for JSON data manipulation.`
Bibliothèques personnalisées	Créer et utiliser votre propre bibliothèque	`library my_utils; int add(int a, int b) => a + b;`

Skills Network