Dart - Programovací jazyk

Co je Dart?

Dart je objektově orientovaný, class-based programovací jazyk vyvinutý společností **Google**. Je optimalizovaný pro tvorbu uživatelských rozhraní a běží na různých platformách.

Základní syntaxe

Hello World

```
void main() {
  print('Ahoj světe!');
}
```

Proměnné a datové typy

```
// Automatické odvození typu
var name = 'Jan';
var year = 2024;

// Explicitní typy
String jmeno = 'Petr';
int vek = 25;
double vyska = 180.5;
bool jePravda = true;

// Dynamický typ
dynamic cokoliv = 'text';
cokoliv = 42;

// Konstanty
const PI = 3.14159;  // Compile-time konstanta
final nyni = DateTime.now();  // Runtime konstanta

// Null safety
String? nullableString;  // Může být null
String nonNullString = 'text';  // Nemůže být null
```

Řetězce (Strings)

```
// Základní řetězce
String text1 = 'Jednoduchý text';
```

```
String text2 = "Dvojité uvozovky";

// Víceřádkové řetězce
String viceradkovy = '''
Toto je
víceřádkový
text
''';

// String interpolace
String jmeno = 'Jan';
String pozdrav = 'Ahoj, $jmeno!';
String vypocet = 'Součet: ${2 + 2}';

// Raw string (bez escape sekvencí)
String cesta = r'C:\Users\Documents';
```

Kolekce

Seznamy (Lists)

```
// List s automatickým typem
var cisla = [1, 2, 3, 4, 5];

// List s explicitním typem
List<String> jmena = ['Jan', 'Petr', 'Marie'];

// Prázdný list
List<int> prazdny = [];

// Přidání prvku
cisla.add(6);
jmena.addAll(['Eva', 'Tomáš']);

// Přístup k prvkům
print(cisla[0]);  // První prvek
print(cisla.length);  // Délka
print(cisla.first);  // První
print(cisla.last);  // Poslední

// Spread operator
var vice = [...cisla, 7, 8, 9];
```

Množiny (Sets)

```
// Set (unikátní hodnoty)
var mnozina = {'a', 'b', 'c'};
Set<int> cisla = {1, 2, 3, 3};  // {1, 2, 3}
```

```
// Operace s množinami
mnozina.add('d');
mnozina.remove('a');
print(mnozina.contains('b'));  // true
```

Mapy (Maps)

```
// Map (klíč-hodnota)
var osoba = {
  'jmeno': 'Jan',
  'vek': 25,
  'mesto': 'Praha'
};

// Explicitní typ
Map<String, int> skore = {
  'Jan': 100,
  'Petr': 85
};

// Přístup k hodnotám
print(osoba['jmeno']);

// Přidání/změna
osoba['email'] = 'jan@example.com';

// Kontrola klíče
if (osoba.containsKey('vek')) {
  print('Věk existuje');
}
```

Funkce

Základní funkce

```
// Funkce s návratovým typem
int secti(int a, int b) {
  return a + b;
}

// Funkce void (bez návratové hodnoty)
void pozdrav(String jmeno) {
  print('Ahoj, $jmeno!');
}

// Arrow funkce (jednořádková)
```

```dart
int nasobok(int x, int y) => x * y;

// Volitelné parametry
void info(String jmeno, [int? vek]) {
  print('Jméno: $jmeno');
  if (vek != null) print('Věk: $vek');
}

// Pojmenované parametry
void vytvorUzivatele({required String jmeno, int vek = 18}) {
  print('$jmeno, věk: $vek');
}

// Použití
vytvorUzivatele(jmeno: 'Jan');
vytvorUzivatele(jmeno: 'Petr', vek: 25);
```

Anonymní funkce

```dart
var cisla = [1, 2, 3, 4, 5];

// Anonymní funkce
cisla.forEach((cislo) {
  print(cislo * 2);
});

// Arrow syntax
cisla.forEach((cislo) => print(cislo * 2));
```

Třídy a objekty

Základní třída

```dart
class Osoba {
  // Vlastnosti
  String jmeno;
  int vek;

  // Konstruktor
  Osoba(this.jmeno, this.vek);

  // Pojmenovaný konstruktor
  Osoba.bezVeku(this.jmeno) : vek = 0;

  // Metoda
  void predstavSe() {
    print('Jmenuji se $jmeno a je mi $vek let.');
```

```
  }

  // Getter
  bool get jeDospecly => vek >= 18;

  // Setter
  set nastavVek(int novyVek) {
    if (novyVek > 0) vek = novyVek;
  }
}

// Použití
void main() {
  var osoba = Osoba('Jan', 25);
  osoba.predstavSe();
  print(osoba.jeDospecly);
}
```

Dědičnost

```
class Zvire {
  String jmeno;

  Zvire(this.jmeno);

  void zvuk() {
    print('Zvíře vydává zvuk');
  }
}

class Pes extends Zvire {
  String rasa;

  Pes(String jmeno, this.rasa) : super(jmeno);

  @override
  void zvuk() {
    print('$jmeno štěká: Haf haf!');
  }

  void aportuj() {
    print('$jmeno aportuje míček');
  }
}

void main() {
  var pes = Pes('Rex', 'Labrador');
  pes.zvuk();
  pes.aportuj();
```

```
}
```

Abstraktní třídy

```
abstract class Tvar {
  // Abstraktní metoda
  double vypocitejObsah();

  // Konkrétní metoda
  void zobraz() {
    print('Obsah: ${vypocitejObsah()}');
  }
}

class Kruh extends Tvar {
  double polomer;

  Kruh(this.polomer);

  @override
  double vypocitejObsah() {
    return 3.14 * polomer * polomer;
  }
}
```

Rozhraní (Interfaces)

```
// V Dartu je každá třída implicitně rozhraní
class Lietadlo {
  void letej() {
    print('Létám');
  }
}

class Ptak implements Lietadlo {
  @override
  void letej() {
    print('Mávám křídly');
  }
}
```

Mixiny

```
mixin Plavani {
  void plav() {
    print('Plavám');
  }
```

```
}

mixin Letani {
  void letej() {
    print('Létám');
  }
}

class Kachna with Plavani, Letani {
  String jmeno;

  Kachna(this.jmeno);
}

void main() {
  var kachna = Kachna('Donald');
  kachna.plav();
  kachna.letej();
}
```

Řídící struktury

Podmínky

```
// If-else
int vek = 20;

if (vek >= 18) {
  print('Dospělý');
} else if (vek >= 13) {
  print('Teenager');
} else {
  print('Dítě');
}

// Ternární operátor
String status = (vek >= 18) ? 'Dospělý' : 'Nedospělý';

// Switch
String den = 'pondělí';

switch (den) {
  case 'pondělí':
  case 'úterý':
    print('Začátek týdne');
    break;
  case 'pátek':
    print('Konec pracovního týdne');
```

```
    break;
  default:
    print('Jiný den');
}
```

Cykly

```
// For cyklus
for (int i = 0; i < 5; i++) {
  print(i);
}

// For-in cyklus
var jmena = ['Jan', 'Petr', 'Marie'];
for (var jmeno in jmena) {
  print(jmeno);
}

// While cyklus
int i = 0;
while (i < 5) {
  print(i);
  i++;
}

// Do-while cyklus
int j = 0;
do {
  print(j);
  j++;
} while (j < 5);

// ForEach
jmena.forEach((jmeno) => print(jmeno));
```

Asynchronní programování

Future

```
// Základní Future
Future<String> nactiData() async {
  await Future.delayed(Duration(seconds: 2));
  return 'Data načtena';
}

// Použití async/await
void main() async {
```

```
  print('Načítám...');
  String data = await nactiData();
  print(data);
}

// Zpracování chyb
Future<void> zpracujData() async {
  try {
    var data = await nactiData();
    print(data);
  } catch (e) {
    print('Chyba: $e');
  } finally {
    print('Hotovo');
  }
}

// Future.then()
nactiData().then((data) {
  print(data);
}).catchError((error) {
  print('Chyba: $error');
});
```

Stream

```
// Stream generátor
Stream<int> pocitadlo() async* {
  for (int i = 1; i <= 5; i++) {
    await Future.delayed(Duration(seconds: 1));
    yield i;
  }
}

// Poslouchání streamu
void main() async {
  await for (var cislo in pocitadlo()) {
    print(cislo);
  }
}

// StreamController
import 'dart:async';

void prikladStreamController() {
  var controller = StreamController<int>();

  // Poslech
  controller.stream.listen((data) {
    print('Přijato: $data');
```

```
  });

  // Přidání dat
  controller.add(1);
  controller.add(2);
  controller.add(3);

  controller.close();
}
```

Výjimky

```
// Vyhození výjimky
void zkontrolujVek(int vek) {
  if (vek < 0) {
    throw Exception('Věk nemůže být záporný');
  }
}

// Zachycení výjimky
void main() {
  try {
    zkontrolujVek(-5);
  } on Exception catch (e) {
    print('Chyba: $e');
  } catch (e, stackTrace) {
    print('Neočekávaná chyba: $e');
    print('Stack trace: $stackTrace');
  } finally {
    print('Vždy se provede');
  }
}

// Vlastní výjimka
class MojeVyjimka implements Exception {
  String zprava;

  MojeVyjimka(this.zprava);

  @override
  String toString() => 'MojeVyjimka: $zprava';
}
```

Generika

```
// Generická třída
class Box<T> {
```

```
  T obsah;

  Box(this.obsah);

  T getObsah() => obsah;
}

// Použití
var intBox = Box<int>(42);
var stringBox = Box<String>('Ahoj');

// Generická funkce
T prvniPrvek<T>(List<T> seznam) {
  return seznam[0];
}

// Omezení typu
class Srovnavac<T extends Comparable> {
  T max(T a, T b) {
    return a.compareTo(b) > 0 ? a : b;
  }
}
```

Důležité knihovny

dart:core

```
// Práce s čísly
int cislo = int.parse('42');
double des = double.parse('3.14');
String retezec = 42.toString();

// Práce s datumem
DateTime nyni = DateTime.now();
DateTime datum = DateTime(2024, 1, 15);
DateTime parsed = DateTime.parse('2024-01-15');

print(nyni.year);
print(nyni.month);
print(nyni.day);

// Duration
Duration hodina = Duration(hours: 1);
DateTime pozdeji = nyni.add(hodina);
```

dart:math

```dart
import 'dart:math';

// Matematické funkce
double odmocnina = sqrt(16);  // 4.0
double mocnina = pow(2, 3) as double;  // 8.0
double zaokrouhleni = 3.7.round() as double;  // 4.0

// Náhodná čísla
var random = Random();
int nahodneCislo = random.nextInt(100);  // 0-99
double nahodnyDouble = random.nextDouble();  // 0.0-1.0

// Konstanty
print(pi);  // 3.141592653589793
print(e);   // 2.718281828459045
```

dart:collection

```dart
import 'dart:collection';

// Queue
Queue<int> fronta = Queue();
fronta.add(1);
fronta.add(2);
fronta.removeFirst();

// HashMap
HashMap<String, int> mapa = HashMap();
mapa['klic'] = 100;

// LinkedHashMap (zachovává pořadí)
LinkedHashMap<String, int> serazena = LinkedHashMap();
```

Užitečné operátory

```dart
// Null-aware operátory
String? jmeno;
String zobrazeni = jmeno ?? 'Neznámé';  // Pokud null, použij defaultní

jmeno ??= 'Jan';  // Přiřaď pouze pokud je null

String? delka = jmeno?.length.toString();  // Bezpečný přístup

// Cascade notation
var osoba = Osoba('Jan', 25)
  ..predstavSe()
  ..nastavVek = 26;
```

```
// Type test operátory
if (osoba is Osoba) {
  print('Je to Osoba');
}

if (osoba is! String) {
  print('Není to String');
}
```

Flutter příklad

```
import 'package:flutter/material.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Dart + Flutter',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Moje první aplikace'),
        ),
        body: Center(
          child: Text(
            'Ahoj světe!',
            style: TextStyle(fontSize: 24),
          ),
        ),
      ),
    );
  }
}
```

Užitečné tipy

- Vždy používejte **null safety** - kontrolujte nullable typy pomocí ?
- Preferujte `final` a `const` pro neměnné proměnné
- Používejte `async/await` pro asynchronní operace místo `then()`
- Vyzkoušejte **DartPad** (dartpad.dev) pro rychlé experimenty
- Pro Flutter projekty používejte **Hot Reload** (`r`) a **Hot Restart** (R)

Odkazy

- Oficiální Dart web
- Dart průvodci
- Dart API dokumentace
- DartPad - online editor
- Flutter framework