Dart 빠른 참조

타입, 함수, 클래스, 비동기, null 안전성, 컬렉션

기본

Hello World

```
void main() {
  print('Hello, Dart!');
}
```

변수

```
var name = 'Dart';       // type inferred
String lang = 'Dart';    // explicit type
final pi = 3.14;         // runtime constant
const max = 100;         // compile-time constant
```

문자열 보간

```
var name = 'World';
print('Hello, $name!');
print('1 + 1 = ${1 + 1}');
```

타입

내장 타입

int	64비트 정수
double	64비트 부동소수점
num	int와 double의 상위 타입
String	UTF-16 문자열
bool	true 또는 false
List	순서 있는 컬렉션 (배열)
Set	순서 없는 고유 컬렉션
Map	키-값 쌍
dynamic	모든 타입, 정적 검사 비활성화
void	반환값 없음

타입 검사 & 캐스팅

```
if (obj is String) print(obj.length);
var s = obj as String;   // cast
print(obj.runtimeType);  // runtime type
```

함수

함수 문법

```
int add(int a, int b) => a + b;
void greet({required String name}) {
  print('Hello, $name');
}
```

매개변수

int f(int a)	필수 위치 매개변수
int f([int a = 0])	기본값이 있는 선택적 위치 매개변수
f({required int a})	필수 명명 매개변수
f({int a = 0})	기본값이 있는 선택적 명명 매개변수

클로저 & 테어오프

```
var square = (int n) => n * n;
[1, 2, 3].map((e) => e * 2);
[1, 2, 3].forEach(print);  // tearoff
```

제어 흐름

조건문

```
if (x > 0) { print('pos'); }
else if (x == 0) { print('zero'); }
else { print('neg'); }
var result = x > 0 ? 'pos' : 'neg';
```

반복문

```
for (var i = 0; i < 5; i++) { }
for (var item in list) { }
while (x > 0) { x--; }
do { x--; } while (x > 0);
```

Switch & 패턴 매칭

```
switch (color) {
  case 'red':   print('R'); break;
  case 'blue':  print('B'); break;
  default:      print('?');
}
```

클래스

클래스 정의

```
class Point {
  final double x, y;
  Point(this.x, this.y);
  double distanceTo(Point p) =>
    sqrt(pow(x - p.x, 2) + pow(y - p.y, 2));
}
```

명명 & 팩토리 생성자

```
class Point {
  double x, y;
  Point(this.x, this.y);
  Point.origin() : x = 0, y = 0;
  factory Point.fromJson(Map j) =>
    Point(j['x'], j['y']);
}
```

상속

```
class Animal { void speak() {} }
class Dog extends Animal {
  @override
  void speak() => print('Woof');
}
```

믹스인 & 확장

믹스인

```
mixin Flyable {
  void fly() => print('Flying');
}
class Bird with Flyable {}
```

확장 메서드

```
extension StringX on String {
  String capitalize() =>
    '${this[0].toUpperCase()}${substring(1)}';
}
print('hello'.capitalize()); // Hello
```

추상 & 구현

```
abstract class Shape {
  double area();
}
class Circle implements Shape {
  final double r;
  Circle(this.r);
  @override double area() => pi * r * r;
}
```

Async/Await

Future

```
Future<String> fetchData() async {
  var res = await http.get(uri);
  return res.body;
}
```

Stream

```
Stream<int> count(int n) async* {
  for (var i = 0; i < n; i++) {
    yield i;
  }
}
```

에러 처리

```
try {
  var data = await fetchData();
} on HttpException catch (e) {
  print('HTTP error: $e');
} catch (e) {
  print('Error: $e');
}
```

컬렉션

List 연산

```
var nums = [1, 2, 3];
nums.add(4);
nums.where((n) => n > 2);     // [3, 4]
nums.map((n) => n * 2);        // [2,4,6,8]
var sorted = nums..sort();
```

Map 연산

```
var m = {'a': 1, 'b': 2};
m['c'] = 3;
m.containsKey('a');  // true
m.entries.map((e) => '${e.key}=${e.value}');
```

스프레드 & 컬렉션 If/For

```
var all = [0, ...nums];
var nav = ['Home', if (isAdmin) 'Admin'];
var sq = [for (var i in nums) i * i];
```

Null 안전성

nullable 타입

int?	nullable int (null 가능)
int	non-nullable int (null 불가)
!	null 단언 연산자
?.	null 인식 접근
??	null이면 기본값
??=	null이면 할당
late	지연 초기화

Null 안전성 예시

```
String? name;            // nullable
int len = name?.length ?? 0;
late final String title; // set before use
name ??= 'default';     // assign if null
```

Dart 빠른 참조

일반 패턴

값이 있는 Enum

```
enum Color {
  red('FF0000'), green('00FF00');
  final String hex;
  const Color(this.hex);
}
```

레코드 & 구조 분해

```
(String, int) userInfo() => ('Alice', 30);
var (name, age) = userInfo();
print('$name is $age');
```

봉인 클래스

```
sealed class Shape {}
class Circle extends Shape { final double r; Circle(this.r); }
class Rect extends Shape { final double w, h; Rect(this.w,
this.h); }
```