


4 닥트 3.0 신규 문법 요약

작성자: 이 상용 



레코드 (Record)

어원 및 배경

레코드는 라틴어 "recordari(기록하다)"에서 유래되었으며, 여러 값을 하나의 묶음으로 취급하는 자료 구조라는 의미를 가진다.

기존 닥트에서는 여러 값을 반환할 때 List, Map, Class 등으로 구조를 만들어야 했다. 하지만 단순 조합 반환에는 너무 과하고 불편했다.

역사 및 개념

Dart 3.0부터 record type이라는 이름으로 복수 값의 그룹을 간단히 표현할 수 있는 구조가 도입되었다. 이는 Tuple 개념과 유사하다.

복수 타입을 단일 구조로 표현

- 위치 기반 positional record: (String, int)
- 이름 기반 named record: ({String name, int age})

기본 문법

```
(String, int) user = ('Alice', 25);  
print(user.$1); // 'Alice'  
print(user.$2); // 25
```

```
({String name, int age}) user = (name:  
'Bob', age: 30);  
print(user.name); // 'Bob'
```

구조 분해 (Destructuring)

어원 및 배경

destruct는 "해체하다"라는 뜻으로, 구조를 부분 요소로 분리하여 개별 변수로 추출하는 행위를 의미한다.

JS, Python, Kotlin 등에서는 이미 리스트나 오브젝트의 속성을 직접 꺼내는 구조 분해가 존재했고, Dart도 이에 발맞추어 기능을 도입했다.

개념

리스트, 맵, 클래스의 값을 변수로 직접 추출 가능

선언 구조를 그대로 따라가면 해체할 수 있음

기본 문법

final [a, b] = [1, 2]; // 리스트

final {'name': name} = {'name': '민지'}; // 맵

final (x: x, y: y) = point; // record/class

switch 문법 개선



어원 및 배경

switch는 선택을 뜻하며, 여러 조건 중 하나에 따라 분기 처리하는 제어문이다.

기존 switch 문은 단순한 값 분기에 불과했지만, Dart 3.0에서는 다음 기능이 추가되며 강력해졌다:



추가된 기능

- 표현식 사용
- 패턴 매칭
- 완전 검사 (exhaustiveness)
- 가드 절 (guard clause)



개념 예시

```
String result = switch (val) {  
  'A' => 'Excellent',  
  'B' => 'Good',  
  _ => 'Fail',  
};
```



기본 문법 요약

- `=>`: 표현식 반환
- `case (T a, T b):` 구조 매칭
- `when` 조건: 가드 절
- `default` 또는 `_`: 모든 케이스 커버

```
naony: dartt_haltu ns  
laed.switch ewrtirg oukt mech:atertuatertberilin  
ig ""  
  
ebanage >- esting jt usinveretp<f }  
ottterr's guarcd clauces: r""  
  
ornatent>preteharis -res-clrettirn)  
  
^:sten_2;;  
ag2  
  
nanced.switch matlerg_mattemingl );  
ast;  
  
ood last;  
rgns+o;  
  
llawarge 'stlern hit_ marngling)"";  
  
ore_maching fingerclent x<;  
ard_*{;  
  
agg"azes lartk,*);  
/yags  
args  
  
nzenaced switch ssatm:mert'3);  
  
ewenil marnging"";
```

클래스 제한자 (Class Modifiers)



어원: modifier는 "수식어", 즉 클래스의 속성, 사용 범위를 제한하는 키워드이다.

배경: 다트는 기존에 클래스의 상속과 구현을 제어할 수단이 부족했다. 캡슐화, 라이브러리 보호, 강제성 부여가 필요했고, 이를 위해 다양한 제한자가 도입되었다.

예시:

```
base class Animal {}  
final class Dog extends Animal {} // 가능  
class Cat extends Animal {} // 에러
```

간단한 실습 예제

문제 1: 레코드 타입과 구조 분해를 활용한 사용자 정보 저장

이름(String), 나이(int), 회원 여부(bool)를 포함한 레코드를 선언하고 구조 분해로 각각 출력해 보세요.

```
void main() {
    (String name, int age, bool isMember)
    user = ('민지', 20, true);
    print(user.$1); // 민지
    print(user.$2); // 20
    print(user.$3); // true
}
```

문제 2: 리스트와 Map에서 구조 분해 사용하기

리스트와 맵을 각각 구조 분해하여 변수로 출력해 보세요.

```
void main() {
    final [a, b] = [10, 20];
    final {'name': name} = {'name': '해린'};

    print(a); // 10
    print(b); // 20
    print(name); // 해린
}
```

문제 3: switch 표현식을 활용한 요일 번역기 만들기

입력된 한글 요일을 영어 요일로 변환하는 코드를 작성하세요.

```
void main() {
    String dayKor = '수요일';
    String dayEng = switch (dayKor) {
        '월요일' => 'Mon',
        '화요일' => 'Tue',
        '수요일' => 'Wed',
        '목요일' => 'Thu',
        _ => 'Unknown'
    };

    print(dayEng); // Wed
}
```

응용 실습 예제

문제 1: 패턴 매칭과 가드 절을 활용한 튜플 검사기 만들기

레코드 형태 (int, int)의 값을 받아 첫 값이 1이고 두 번째 값이 0보다 크면 "통과", 아니면 "실패"를 출력하세요.

```
void main() {
  (int a, int b) record = (1, 5);

  switch (record) {
  case (1, _) when record.$2 > 0:
    print("통과");
    break;
  default:
    print("실패");
  }
}
```

문제 3: mixin class를 사용해 기능 추가

LoggerMixin을 mixin class로 정의하고, Service 클래스에 로그 기능을 추가하여 출력하세요.

```
mixin class LoggerMixin {
  void log(String msg) {
    print(" 로그: $msg");
  }
}

class Service with LoggerMixin {
  void run() {
    log("서비스 시작");
  }
}

void main() {
  Service s = Service();
  s.run(); // 로그: 서비스 시작
}
```



문제 2: sealed 클래스와 switch exhaustiveness 사용

sealed 클래스를 정의하고, 해당 클래스 타입에 따라 switch문에서 조건별로 다르게 출력하세요.

```
sealed class Animal {}

class Dog extends Animal {}
class Cat extends Animal {}

void describe(Animal animal) {
  switch (animal) {
  case Dog():
    print("강아지");
  case Cat():
    print("고양이");
  }
}

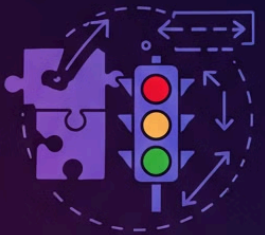
void main() {
  describe(Dog()); // 강아지
}
```



Dart 3.0



Records



Destructing
switch statgements



Eocle porce switch sag'reffft eroj
thjus, tre'ay trastb, bul suc
angganm oäding)
proiertil.



Destructig's tatements



pr eh'iod and tundortte
or cosentie
simpeirnte!



Class madierrs
for issats udriesie.

핵심 정리 요약

항목	개념 및 핵심 요약
✓ 레코드	여러 값을 하나의 묶음으로 표현하는 새로운 타입, Tuple 유사
✓ 구조 분해	복합 구조를 변수로 직접 해체할 수 있는 문법, 리스트/맵/클래스 대상
✓ switch 확장	표현식 반환, 패턴 매칭, 가드 절, 엄격한 검사 도입
✓ 클래스 제한자	캡슐화/의도 제어 목적의 base, sealed, interface, mixin class 도입