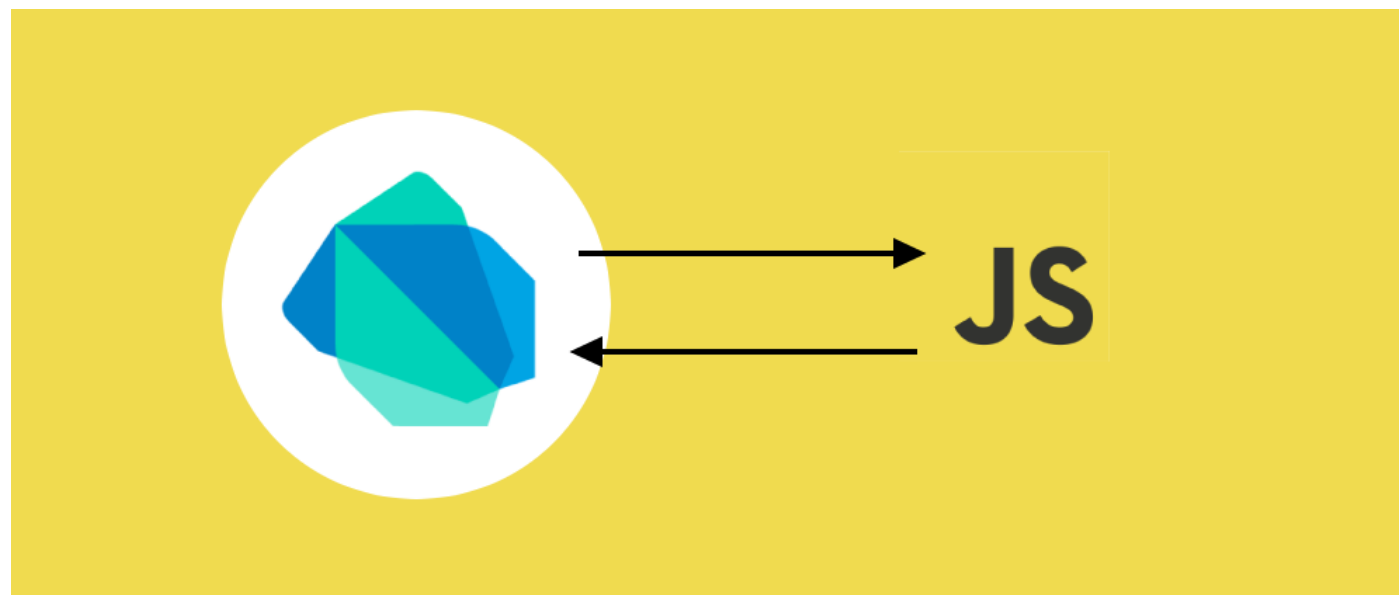


Dart 언어 소개





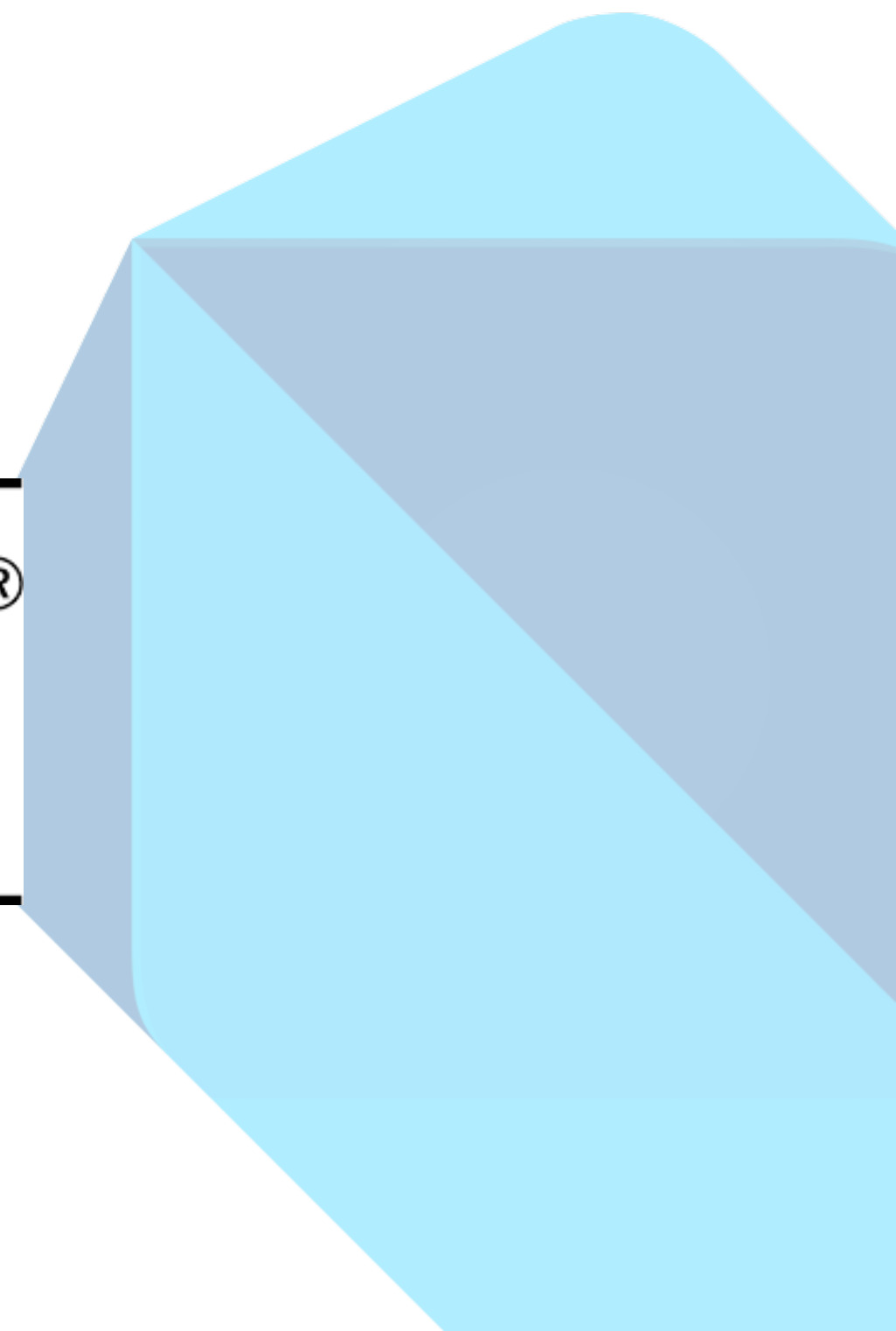
COME ^{TO} THE
DART
— **SIDE**



출처

How to use JavaScript libraries in your Dart applications
<https://www.slideshare.net/StphaneEsteGracias/dart-on-server-meetup-18052017>

W3C®





vs

Google

구글 vs 애플

- 광고 기반 회사
- 웹을 주력으로 함
- 주요 프로덕트
구글 드라이브, 지메일 등
웹 애플리케이션



























- 하드웨어 판매 기반 회사
- 앱을 주력으로 함
- 주요 프로덕트
아이폰, 아이패드 등 하드웨어
(의 판매 증대를 위한 앱스토어
인프라 구축)



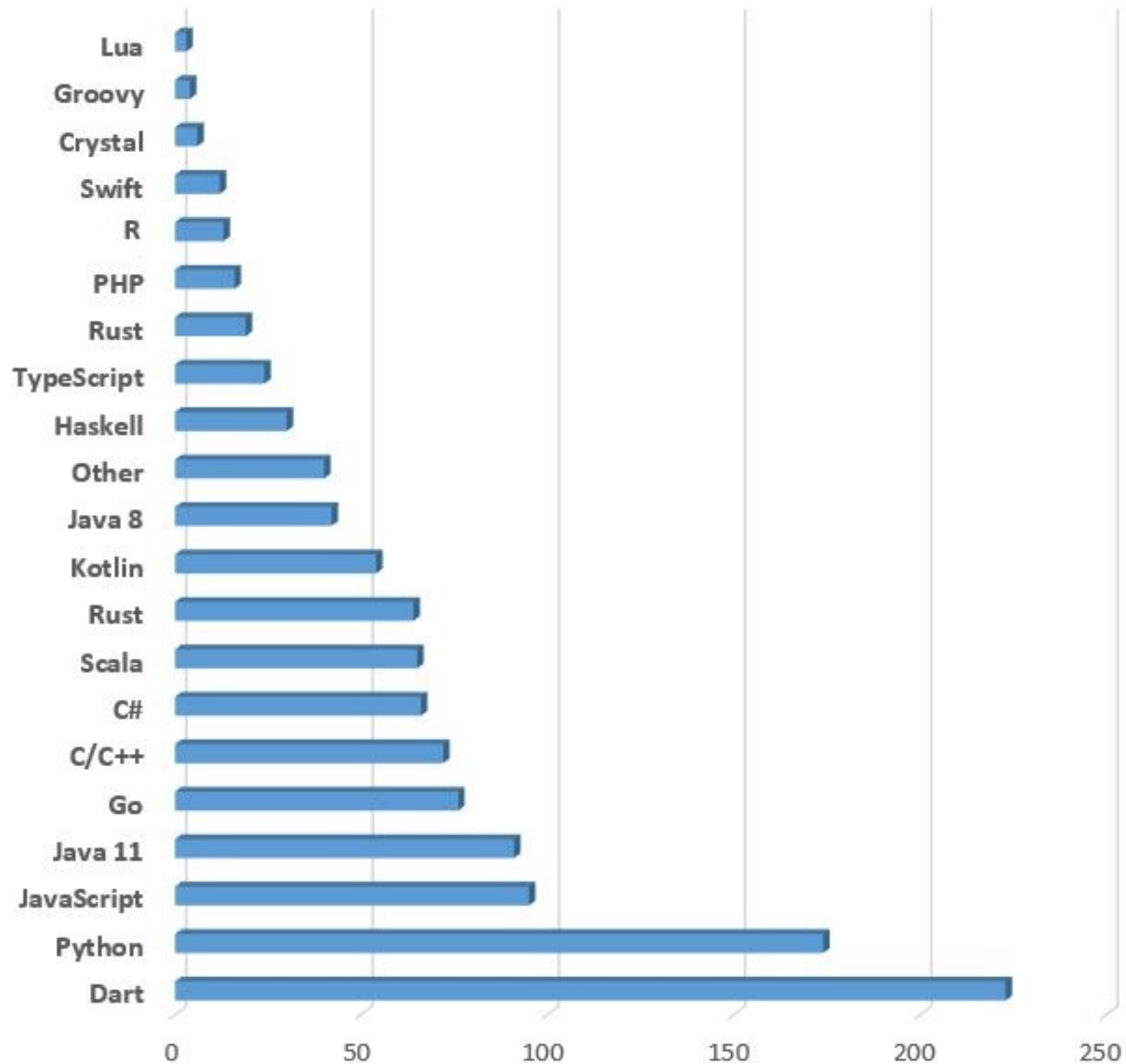
Dart

Worst Programming Languages to Learn in 2018 Rankings

Ranked from Worst to Best Languages to Learn




	Overall Rankings	Community Engagement	Job Market	Growth and Trends
1	 Dart	 Dart	 Dart	 Objective-C
2	 Objective-C	 CoffeeScript	 Rust	 CoffeeScript
3	 CoffeeScript	 Objective-C	 Elm	 Dart
4	 Erlang	 Lua	 Lua	 Perl
5	 Lua	 Elm	 Erlang	 Erlang
6	 Clojure	 Clojure	 Clojure	 Clojure
7	 Perl	 Elixir	 Kotlin	 Ruby
8	 Elm	 Erlang	 Elixir	C# C#
9	 Elixir	 Kotlin	 R	 Lua
10	 Haskell	 Perl	 Perl	 C
11	 Rust	 Scala	 Haskell	 Haskell
12	 Scala	 TypeScript	 CoffeeScript	 Rust
13	 C	 Haskell	 Scala	 Elm
14	 Ruby	 Rust	 TypeScript	 Elixir
15	 Go	 R	 Objective-C	 Scala
16	 TypeScript	 Swift	 C	 Swift
17	 Swift	 Go	 Go	 Go
18	 Kotlin	 C	 Swift	 R
19	 R	 Ruby	C# C#	 TypeScript
20	C# C#	C# C#	 Ruby	 Kotlin

Which programming language will be relevant to you in 2019 and you want to hear more about?



출처 : <https://jaxenter.com/poll-results-dart-word-2019-154779.html>



-  Mobile
-  Web
-  Desktop
-  Embedded

Dart Packages

- pub 이라는 패키지 관리자 제공
- <https://pub.dev>
- pubspec.yaml 기반으로 동작

상수

```
final constName = '상수';  
const constName = '상수';
```

변수

```
var varName = '변수';
```

```
string varName = '타입 지정한 변수';
```

```
dynamic varName = '유연한 타입';
```

함수

```
void exec( int input ) { /* 리턴 값이 없는 함수 */ }
```

```
exec( int input ) { /* 가능하나 권장하지 않음 */ }
```

```
int exec( int input ) { /* 리턴 형식이 지정됨 */ }
```

함수의 호출

```
void message( string to, string content ) { ... }
```

// 전통적인 방식의 값 전달

```
message( 'mail@example.com', 'Hello' );
```

// 옵션명 호출 Optional named parameters 방식의 전달

```
message( to: 'mail@example.com', content: 'Hello' );
```

시작점

// 시작점 entry point 에 해당하는 main 함수가 필요

```
void main() { ... }
```

// 화살표 함수가 존재하고 main 은 간략히 표현하는 경우가 많음

```
void main() => runApp(MyApp());
```