

# 스타트업에서 Go언어 개발자로 살아남기



안녕하세요! 여러분 🖐️

## 박현상 · HyunSang Park

- Software Engineer @TeamGRIT, Inc.
- Community Organizer @GDG Golang Korea

Email. [hyun.sang@teamgrit.kr](mailto:hyun.sang@teamgrit.kr)

Blog. [hyunsang.dev](http://hyunsang.dev)

GitHub. [@dev-hyunsang](https://github.com/dev-hyunsang)

Linkedin. [@hyunsangpark](https://www.linkedin.com/in/hyunsangpark)

# Table of Contents

- 우연한 계기로 Go언어를 사용하게 된 계기
  - Go언어를 공부하는 방법
  - Go언어로 프로젝트하면 좋을 주제들
- 회사에서는 어떻게 Go언어를 사용하고 있는지
- Go언어로 커리어를 쌓기 위해서는?
  - 커리어를 쌓기 위한 필요한 능력
  - Go언어를 사용하고 있는 회사들
- 개발자로서, 그리고 한 사람으로서.
- Q&A

# 우연한 계기로 Go언어를 사용하게 된 계기

2021년으로 거슬러 올라가...



프론트엔드 개발자가 되고 싶은데  
뭔가 나랑 안 맞아...

무슨 분야의 개발자 되어야 할까?

무슨 프로그래밍 언어를 써야할까?

프로그래밍의 매력이 없네...?

이 프로그래밍 언어를 배워서 과연 커리어를  
성공적으로 쌓을 수 있을까?

이 프로그래밍 언어를 쓰는 회사가 있을까?

# 우연한 계기로 Go언어를 사용하게 된 계기

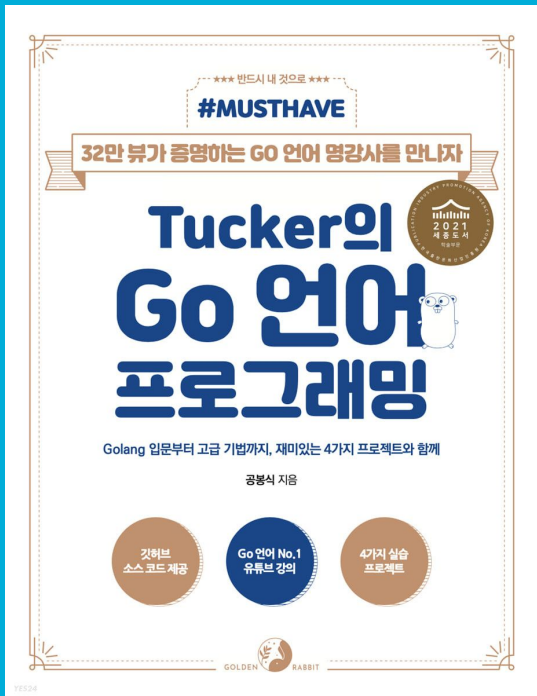
2021년으로 거슬러 올라가...

페이스북으로 우연히 연락하던 팀그릿의 강성일 대표님과 이야기한 이후 몇 개월 동안 Go언어를 공부하고 나서 꾸준한 관심을 가지고 있다는 점을 인상 깊게 보셔서 좋은 기회로 합류하게 되었습니다.



# 우연한 계기로 Go언어를 사용하게 된 계기

## Go언어를 공부하는 방법



Tucker의 Go언어 프로그래밍을 읽고 많은 도움이 되었어요!

이 외에도 Tucker님의 유튜브를 보면 많은 것들을

### Golang Korean Community

Go에 관련된 수다를 추구하는 개발자들의 세상

[golangkorea.github.io](https://golangkorea.github.io)

### Reducing boilerplate with go generate

OCT 16, 2016 - 5 MINUTE READ - [COMMENTS](#) - [번역](#)

Posted by [Timo Boll](#)

• Series: [Advent 2015](#)

Reducing boilerplate with go generate 를 번역한 글입니다 Go는 대단한 언어입니다. 단순하고, 파워풀하며, 훌륭한 도구들을 가지고 있고, 우리 중 많은 이들은 매일 사용하는 것을 즐깁니다. 하지만 강한 타입의 언어들에서 일상적으로 발생하게 되는, 이것저것을 연결하기 위해서 필수로 사용해야 하는 boilerplate를 쓰게 됩니다. 이 포스트에서 다음의 세가지 포인트를 다룰 것입니다: 코드 생성(code generation)을 사용하여 boilerplate를 줄이도록 도와주는 Go 도구들을 만들 수 있어야 하는 이유는 무엇입니까. Go 코드 생성을 위한 블록 구성 요소는 무엇입니까.

[Read On →](#)

Golang Korea Community 블로그에서도 많은 것들을 공부할 수 있어요.

# 우연한 계기로 Go언어를 사용하게 된 계기

## Go언어를 공부하는 방법

[https://go.dev/doc/effective\\_go](https://go.dev/doc/effective_go)

[nomadcoders.org](https://nomadcoders.org)

Documentation > Effective Go

### Effective Go

**Table of Contents**

- Introduction
- Examples
- Formatting
- Commentary
- Names
  - Package names
  - Getters
- Interface names
- MixedCaps
- Semicolons
- Control structures
  - if
  - Redeclaration and reassignment
  - For
  - Switch
  - Type switch
- Functions
  - Multiple return values
  - Named result parameters
- Defer
- Data
  - Allocation with new
  - Constructors and composite literals
  - Allocation with make
  - Arrays
  - Slices
  - Two-dimensional slices
  - Maps
  - Printing
  - Append
  - Initialization

- Constants
- Variables
- The init function
- Methods
  - Pointers vs. Values
- Interfaces and other types
  - Interfaces
  - Conversions
  - Interface conversions and type assertions
  - Generality
  - Interfaces and methods
- The blank identifier
  - The blank identifier in multiple assignment
- Unused imports and variables
- Import for side effect
- Interface checks
- Embedding
- Concurrency
  - Share by communicating
- Goroutines
- Channels
  - Channels of channels
  - Parallelization
- A leaky buffer
- Errors
  - Panic
  - Recover
- A web server


#### Introduction

Go is a new language. Although it borrows ideas from existing languages, it has unusual properties that make effective Go programs different in character from programs written in its relatives. A straightforward translation of a C++ or Java program into Go is unlikely to produce a satisfactory result—Java programs are written in Java, not Go. On the other hand, thinking about the problem from a Go perspective could produce a successful but quite different program. In other words, to write Go well, it's important to understand its properties and idioms. It's also important to know the established conventions for programming in Go, such as naming, formatting, program construction, and so on, so that programs you write will be easy for other Go programmers to understand.

This document gives tips for writing clear, idiomatic Go code. It augments the [language specification](#), the [Tour of Go](#), and [How to Write Go Code](#), all of which you should read first.


Note added January, 2022: This document was written for Go's release in 2009, and has not been updated significantly since. Although it is a good guide to understand how to use the language itself, thanks to the stability of the language, it says little about the libraries and nothing about significant changes to the Go ecosystem since it was written, such as the build system, testing, modules, and polymorphism. There are no plans to update it, as so much has happened and a large and growing set of documents, blogs, and books do a fine job of describing modern Go usage. Effective Go continues to be useful, but the reader should understand it is far from a complete guide. See [issue 28782](#) for context.

중급



노마드 코인  
Go로 암호화폐 만들기

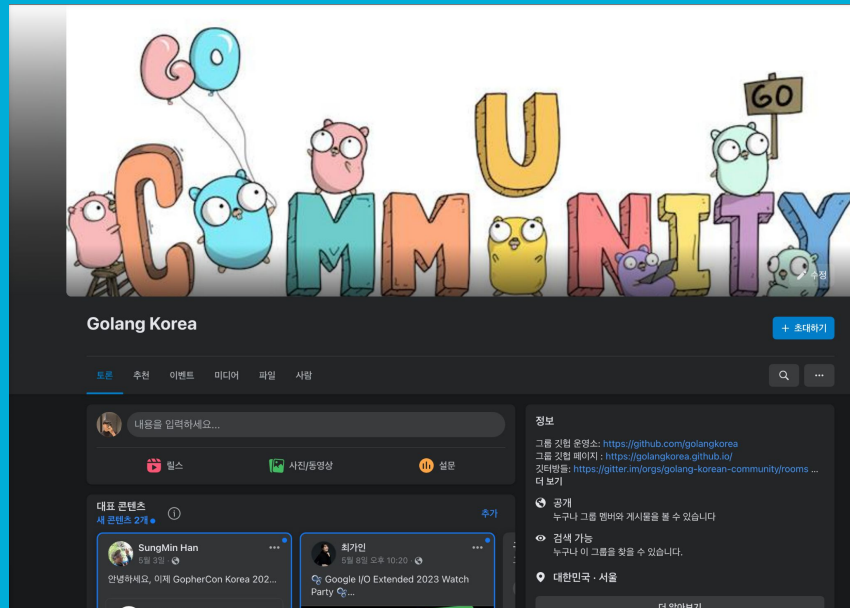
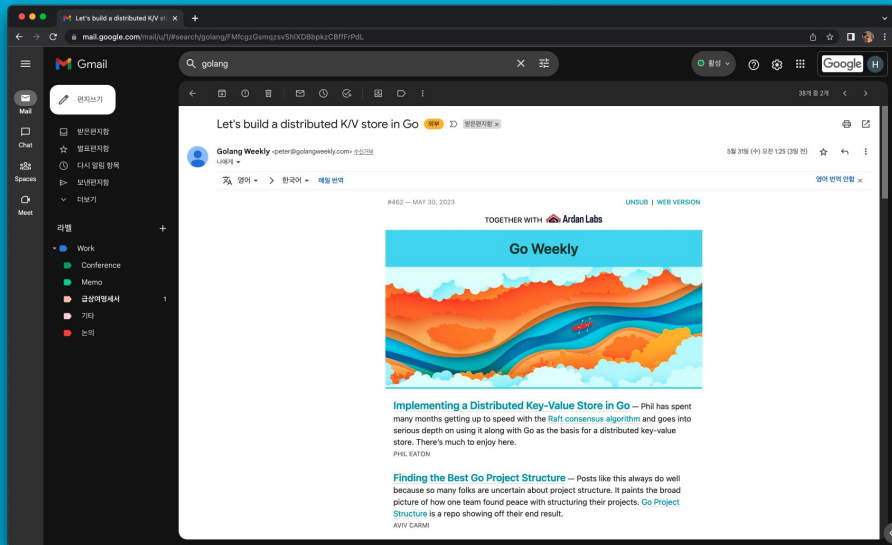
중급



쉽고 빠른 Go 시작하기  
Go for Beginners

# 우연한 계기로 Go언어를 사용하게 된 계기

Go언어를 공부하기 위한 자료 얻기





# Go언어로 프로젝트하면 좋을 주제들

## Back-End

1. **ORM(Object Relational Mapping, 객체-관계 매핑)**
  - a. entgo, gorm, sqlx
2. **JWT(JSON Web Token) + Redis**
  - a. 사용자 인증 관련 회원가입, 로그인, 로그아웃, 인증
3. **gRPC(gRPC Remote Procedure Calls)**
4. **Serverless(serverless.com)**

## Open-Source

1. **CNCF(Cloud Native Computing Foundation)**
  - a. Kubernetes
2. **Docker**
  - a. compose, ETC...

**[mingrammer/go-web-framework-stars](https://github.com/mingrammer/go-web-framework-stars)**

# 미디어 스타트업 '팀그릿'의 Go언어 사용률

## Top languages

● JavaScript ● Go ● Vue ● Python  
● TypeScript

### Scorpion-v2

Private

Media Processing and Storage System

● Go ☆ 0 🍷 0 🔄 0 🔗 0 Updated last week



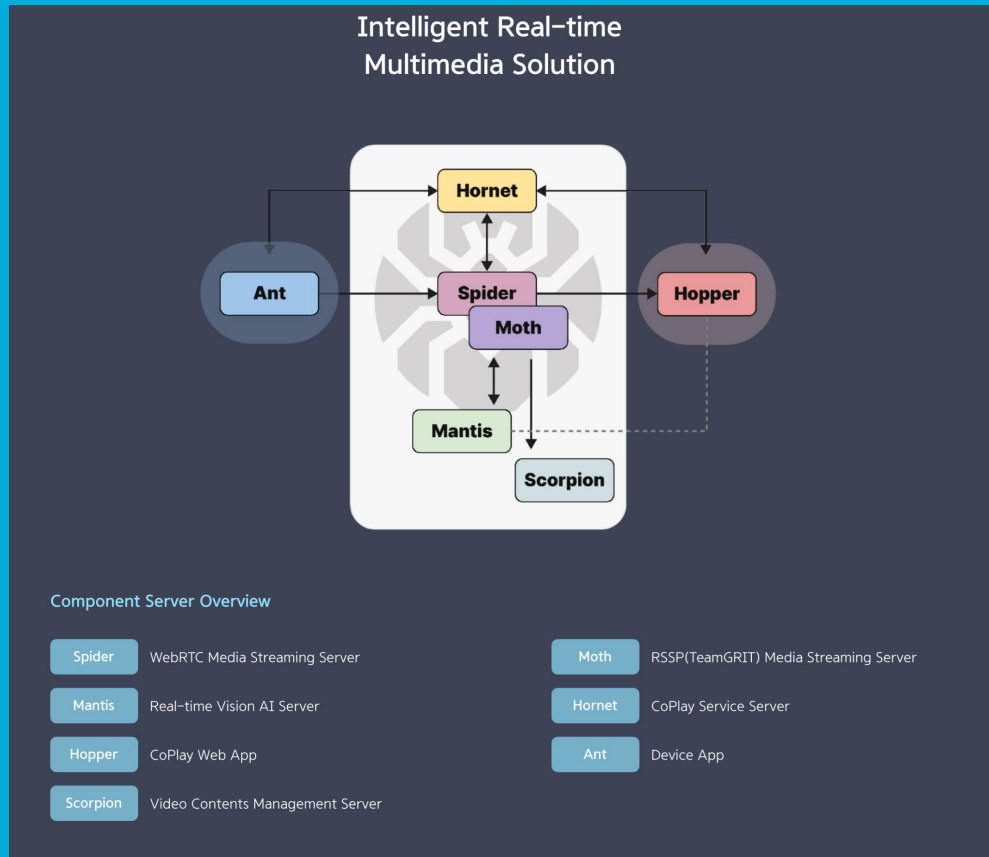
### coplay-service

Private

● Go ☆ 1 🍷 0 🔄 0 🔗 0 Updated 2 hours ago

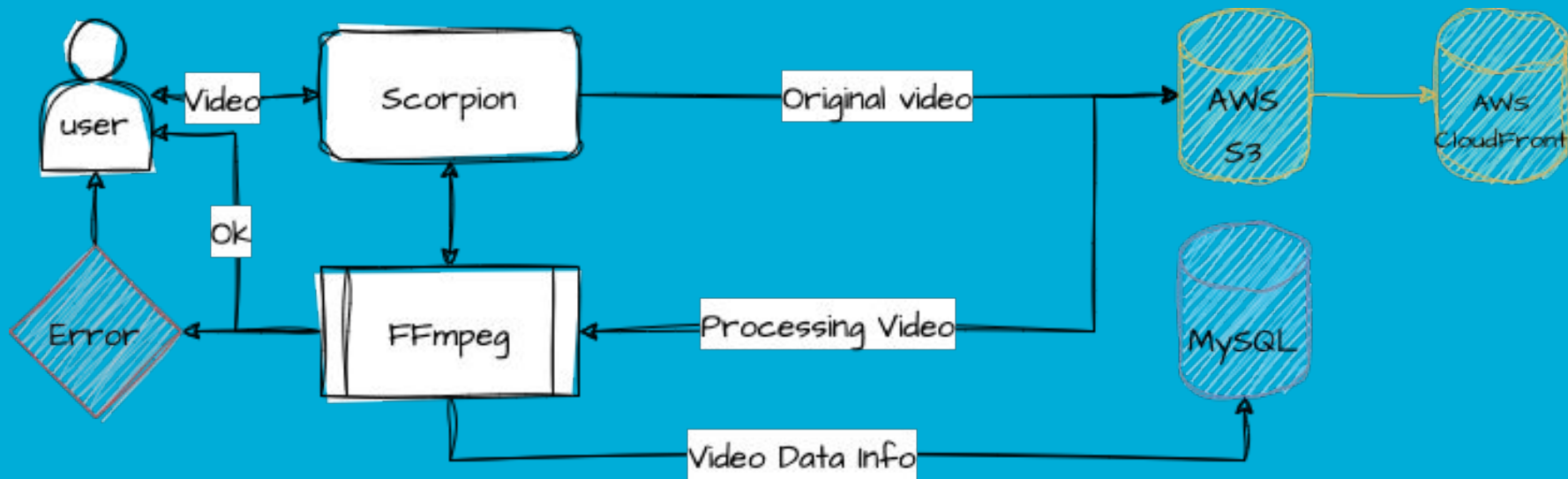


# 미디어 스타트업 '팀그릿'의 Go언어 사용률



# 미디어 스타트업 '팀그릿'의 Go언어 사용

## 스콜피온 (Scorpion) - Video Content Management System



# Go언어 사용하는 회사

42dot

## Backend Engineer (Connected Car Service)

Gangnam-gu, Seoul (Yangjae Station)

**We are looking for the best**

42dot은 도심형 모빌리티 서비스의 A부터 Z까지 전 과정을 아우르는 모빌리티 통합 플랫폼 UMOS (Urban Mobility Operating System)를 개발합니다.

Connected car service backend engineer는 커넥티드 차량에서 필요한 다양한 서비스를 개발합니다.

### 책임 (RESPONSIBILITIES)

OTA(Over-the-Air) 서비스 백엔드 개발 및 운영

자율주행 차량의 원격 제어 서비스 백엔드 개발 및 운영

자율주행 제어기(AKI)에서 필요로 하는 다양한 커넥티드 서비스를 위한 백엔드 개발 및 운영

### 자격요건 (QUALIFICATIONS)

능숙한 Go 언어 프로그래밍 역량

코딩(Secure coding) 역량

소프트웨어 아키텍처 설계 역량 및 확장성(scalability) 설계에 대한 이해

daangn

### 이런 분을 찾고 있어요

- 하나 이상의 프로그래밍 언어에 능숙하신 분
- 문제를 정의하고 해결책을 찾아가는 과정을 즐겨워하시는 분
- 대용량 트래픽 처리를 위한 애플리케이션 아키텍처 구성이 가능하신 분
- 테스트 코드 작성 및 코드 리뷰의 중요성을 알고 실천하시는 분
- REST, gRPC 등의 통신 모델을 이해하고 사용성 높은 API 설계 및 개발이 가능하신 분

### 이런 분이면 더 좋아요!

- SSO, OAuth2, OIDC 등을 활용한 인증/인가 시스템 개발 경험이 있으신 분
- 분산 처리 시스템 또는 마이크로서비스 아키텍처를 이해하고 경험하신 분
- 2개 이상의 다른 종류의 데이터 저장소를 혼합하여 시스템을 설계해 보신 분
- 대용량 서비스에 대해 경험하신 분
- NoSQL + RDBMS과 같이 2개 이상의 다른 종류의 데이터 저장소를 혼합하여 시스템을 설계해 보신 분
- Go 언어를 이용하여 개발해보신 분

# Go언어 사용하는 회사

## Coupang

### Senior Staff Engineer, Play Backend Enginee...

Coupang · 대한민국 서울 (대면근무)

간편지원

저장

...

technical background, you have a high degree of customer obsession to ensure we are building the right products for our customers.

Come innovate with the Coupang Play team!

#### Essential Qualifications

- 10+ years of professional software development exp
- 8+ years of programming experience with at least or Java or Go including object-oriented design.
- 8+ years of experience contributing to the architect design patterns, reliability and scaling) of new and c

#### What You Will Do

- Manage access controls, permissions, and CVE patches
- Maintaining the install , update , and upgrade pages
- Work on Dependency updates, License management and submissions to partners for validations/certifications
- Build and maintain the infrastructure used for creating the various installation methods

#### What You Will Bring

- Familiarity with multi-stage build systems
- Production experience with building Docker images and docker build tooling.
- Experience with Debian and RHEL based systems, and building/packaging archives such as .deb and .rpm
- Experience using Continuous Integration systems (e.g., GitLab CI, Jenkins, Travis).
- Experience with Ruby in production (Golang is a nice bonus)
- Familiarity with building and packaging cloud native applications.

GitLab

개발자로서, 그리고 한 사람으로서.

하고 싶은 것들이 너무 많음...ㅠ

사내에서는 조직문화, 테크니컬 라이팅 등등...

개인적으로 다양한 분들과 다양한 이야기를 나누면서  
성장하기,  
사진전 개최, 사진집 제작, 이것저것...

# 개발자로서, 그리고 한 사람으로서.

'왜 자살하지 않는가?' 카뮈의 질문에 나는 대답한다. 가슴이 설레어 잠을 이루지 못하는 밤이 있다. 이루어지기만 한다면 너무 좋아서 두 주먹을 불끈 쥐고 뛰어오를 것 같은 일이 있다. 누군가 못 견디게 그리워지는 시간이 있다. 더 많은 것을 주고 싶지만 그렇게 할 수가 없어 미안한 사람들이 있다. 설렘과 황홀, 그리움, 사랑의 느낌... 이런 것들이 살아 있음을 기쁘게 만든다. 나는 더 즐겁게 일하고, 더 열심히 놀고, 더 많이 더 깊게 사랑하고 싶다. 더 많은 사람들과 손잡고 더 아름다운 것을 더 많이 만들고 싶다. 미래의 어느 날이나 피안彼岸의 세상에서가 아니라, '지금' 바로 '여기'에서 그렇게 살고 싶다. 떠나는 것이야 서두를 필요가 없다. 더 일할 수도 더 놀 수도 누군가를 더 사랑할 수도 타인과 손잡을 수도 없게 되었을 때, 그때 조금 아쉬움을 남긴 채 떠나면 된다.

유시민의 어떻게 살 것인가 중.



# Q&A

부족한 발표 들어주셔서  
감사합니다.

[hyun.sang@teamgrit.kr](mailto:hyun.sang@teamgrit.kr)