


# 가정용 버스 전광판 만들어 본 썰

2020. Homin Lee



```
func main() {  
    ctx, cancel := context.WithTimeout(  
        context.Background(),  
        1*time.Hour,  
    )  
    defer cancel()  
    wg.Add(2)  
    go playMusic(ctx)  
    go doArbitraryCoding(ctx)  
    wg.Wait()  
}
```

[twitch.tv/suapapa](https://twitch.tv/suapapa)

# 무엇을?



# 어떻게?

- 경기버스정보서비스
  - [gbis.go.kr](http://gbis.go.kr)
- RaspberryPi
- e-Paper display
- Golang



왜?

# 결과물

- gbis-frame
  - <https://github.com/suapapa/gbis-frame>
- epd7in5 e-Paper device driver
  - [https://github.com/suapapa/go\\_devices](https://github.com/suapapa/go_devices)



# 버스 정보 API 테스트

# 공공데이터포털 API키 발급

공공데이터포털: <https://www.data.go.kr/>

- 경기도-정류소 조회 서비스 (REST)
- 경기도-버스노선 조회 서비스 (REST)
- 경기도-버스 도착 정보 조회 서비스 (REST)

참고) 사이트에 승인되었다고 나와도 실제로 쓸 수 있을때까지는 -영업일기준- 0.5일 정도가 걸리는 느낌.



# 공공데이터포털 API키 발급 확인

The screenshot shows the 'DATA GO.KR' public data portal. The main navigation bar includes '데이터셋', '제공신청', '활용사례', '정보공유', and '이용안내'. The left sidebar contains '마이페이지' (My Page) with sub-links for '오픈API', '개발계정', '활용현황', '운영계정', and '인증키 발급현황'. The '마이페이지' section is expanded, showing '개발계정' (Developer Account) details. The '개발계정' section displays three status boxes: '신청' (Application) with 0 items, '활용' (Usage) with 5 items, and '중지' (Suspension) with 4 items. Below these, a table lists the developer accounts, with the first three rows highlighted in red. The table columns are: [승인] 정류소 조회 서비스, 신청일:2020-02-03 [중지신청] 만료:2020-02-03, 서비스유형:SOAP, 분류:교통및물류 > 도로, 제공기관:경기도; [승인] 정류소 조회 서비스, 신청일:2020-02-03 [활용신청] 만료예정:2022-02-03, 서비스유형:REST, 분류:교통및물류 > 도로, 제공기관:경기도; [승인] 버스노선 조회 서비스, 신청일:2020-02-02 [활용신청] 만료예정:2022-02-02, 서비스유형:REST, 분류:교통및물류 > 도로, 제공기관:경기도; [승인] 버스 도착 정보 조회 서비스, 신청일:2020-01-29 [활용신청] 만료예정:2022-01-29, 서비스유형:REST, 분류:교통및물류 > 도로, 제공기관:경기도.

서비스명	신청일	상태	만료예정	서비스유형	분류	제공기관
[승인] 정류소 조회 서비스	2020-02-03	[중지신청]	2020-02-03	SOAP	교통및물류 > 도로	경기도
[승인] 정류소 조회 서비스	2020-02-03	[활용신청]	2022-02-03	REST	교통및물류 > 도로	경기도
[승인] 버스노선 조회 서비스	2020-02-02	[활용신청]	2022-02-02	REST	교통및물류 > 도로	경기도
[승인] 버스 도착 정보 조회 서비스	2020-01-29	[활용신청]	2022-01-29	REST	교통및물류 > 도로	경기도

# 정류소 조회 서비스 테스트

- 정류소의 단축번호(mobileNo) 입력
- 정류소ID(stationID)등을 반환.
- 07479 -> 206000678

```
suapapa@suapapa-B70SH-AJ76B2W: ~/ws/suapapa/go-gbis
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
suapapa@suapapa-B70SH-AJ76B2W:~/ws/suapapa/go-gbis$ curl --include --request GET
"http://openapi.gbis.go.kr/ws/rest/busstationservice?serviceKey=$SERVICEKEY&key
word=07479"
HTTP/1.1 200 OK
Date: Thu, 06 Feb 2020 05:51:01 GMT
Set-Cookie: PHAROSVISITOR=000027880170190d36a53b09c0a867c8; HttpOnly
Content-Type: application/xml;charset=UTF-8
Content-Length: 586

<?xml version="1.0" encoding="UTF-8" standalone="yes"?><response><comMsgHeader><
errMsg>NORMAL SERVICE.</errMsg><returnCode>00</returnCode></comMsgHeader><msgHea
der><queryTime>2020-02-06 14:51:01.300</queryTime><resultCode>0</resultCode><res
ultMessage>정상적으로 처리되었습니다.</resultMessage></msgHeader><msgBody><busSt
ationList><centerYn>N</centerYn><districtCd>2</districtCd><mobileNo> 07479</mobi
leNo><regionName>성남</regionName><stationId>206000678</stationId><stationName>H
스퀘어</stationName><x>127.10945</x><y>37.40185</y></busStationList></msgBody></
response>suapapa@suapapa-B70SH-AJ76B2W:~/ws/suapapa/go-gbis$
```

# 버스노선 조회 서비스 테스트

- 노선ID(routeID)를 입력
- 버스번호(routeName)등을 반환
- 200000085 -> 98

```
suapapa@suapapa-B70SH-AJ76B2W: ~/ws/suapapa/go-gbis
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
suapapa@suapapa-B70SH-AJ76B2W:~/ws/suapapa/go-gbis$ curl --include --request GET
"http://openapi.gbis.go.kr/ws/rest/busrouteservice/info?serviceKey=$SERVICEKEY&
routeId=200000085"
HTTP/1.1 200 OK
Date: Thu, 06 Feb 2020 05:56:01 GMT
Set-Cookie: PHAROSVISITOR=0000278801701911caa251fbc0a867c8; HttpOnly
Content-Type: application/xml;charset=UTF-8
Content-Length: 1140

<?xml version="1.0" encoding="UTF-8" standalone="yes"?><response><comMsgHeader><
errMsg>NORMAL SERVICE.</errMsg><returnCode>00</returnCode></comMsgHeader><msgHea
der><queryTime>2020-02-06 14:56:01.314</queryTime><resultCode>0</resultCode><res
ultMessage>정상적으로 처리되었습니다.</resultMessage></msgHeader><msgBody><busRo
uteInfoItem><companyId>4103100</companyId><companyName>수원여객</companyName><co
mpanyTel>031-244-5341</companyTel><districtCd>2</districtCd><downFirstTime>05:45
</downFirstTime><downLastTime>23:35</downLastTime><endMobileNo>37707</endMobileN
o><endStationId>233001703</endStationId><endStationName>한화꿈에그린</endStation
Name><peekAlloc>8</peekAlloc><regionName>수원, 화성</regionName><routeId>20000008
5</routeId><routeName>98</routeName><routeTypeCd>13</routeTypeCd><routeTypeName>
일반형 시내버스</routeTypeName><startMobileNo>1070</startMobileNo><startStationId
>200000165</startStationId><startStationName>이목동차고지.이목동입구</startStati
onName><upFirstTime>04:50</upFirstTime><upLastTime>22:10</upLastTime><nPeekAlloc
>12</nPeekAlloc></busRouteInfoItem></msgBody></response>suapapa@suapapa-B70SH-AJ
76B2W:~/ws/suapapa/go-gbis$
```

# 버스 도착 정보 조회 서비스 테스트

- stationID로 검색하면,
- routeID들의 도착 정보들을 반환
- gbis.go.kr 에서;
  - 경기버스정보 홈
  - 공유서비스
  - 테스트
  - 버스 도착정보 목록조회
  - <http://www.gbis.go.kr/gbis2014/public/Service.action?cmd=tBusArrivalStation>

고 코딩

# package main 구성

- main.go : 플래그 파싱 및 주요 함수 진입
- const.go : 각 서비스 엔드포인트 주소 등의 상수들
- struct.go : 서비스 반환값을 파싱할 구조체
- businfo.go : 서비스 API 호출 함수들
- config.go : 서비스 키 등의 설정
- draw.go : 이미지로 그리는 프로그램
- eink.go : e-ink 패널에 출력하는 프로그램
- util.go : 자잘한 유틸 함수들



**KEEP  
CALM  
AND  
COPY  
PASTE**

# curl-to-Go

curl로 테스트한 내용을 바로 고 코드로 옮겨 줍니다.

- <https://mholt.github.io/curl-to-go/>

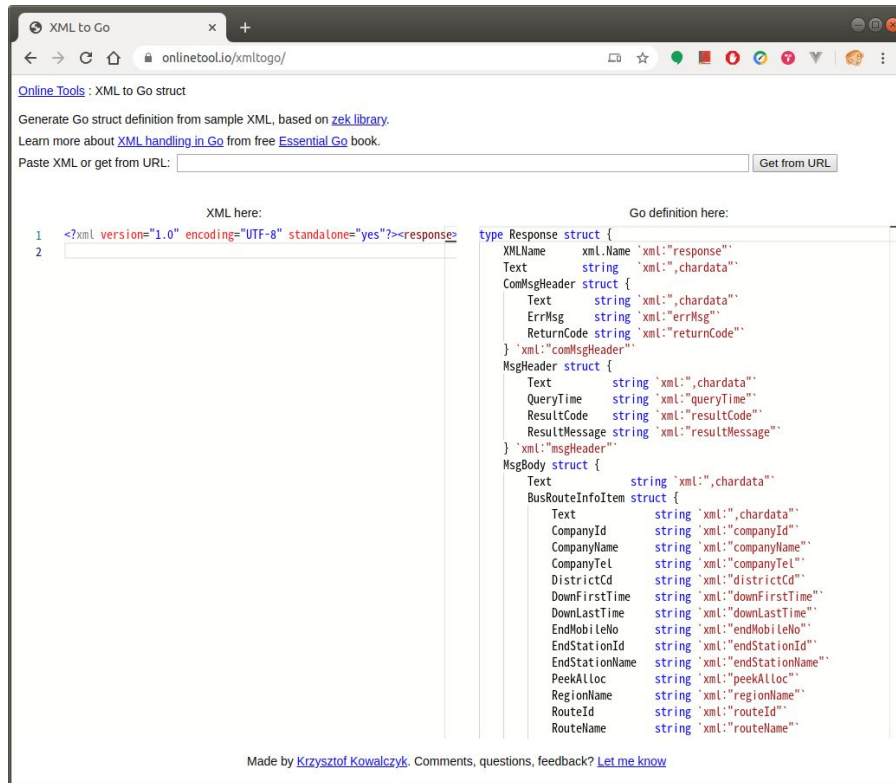




# xml to Go struct

API의 응답 xml을 담을 구조체를 생성해 줍니다.

- <https://www.onlinetool.io/xmltogo/>



The screenshot shows the 'XML to Go' online tool interface. The browser address bar displays 'onlinetool.io/xmltogo/'. The page title is 'XML to Go struct'. Below the title, there are instructions: 'Generate Go struct definition from sample XML, based on [zek library](#). Learn more about [XML handling in Go](#) from free [Essential Go](#) book.' A text input field labeled 'Paste XML or get from URL:' is present, with a 'Get from URL' button to its right. The 'XML here:' section contains a sample XML snippet: 

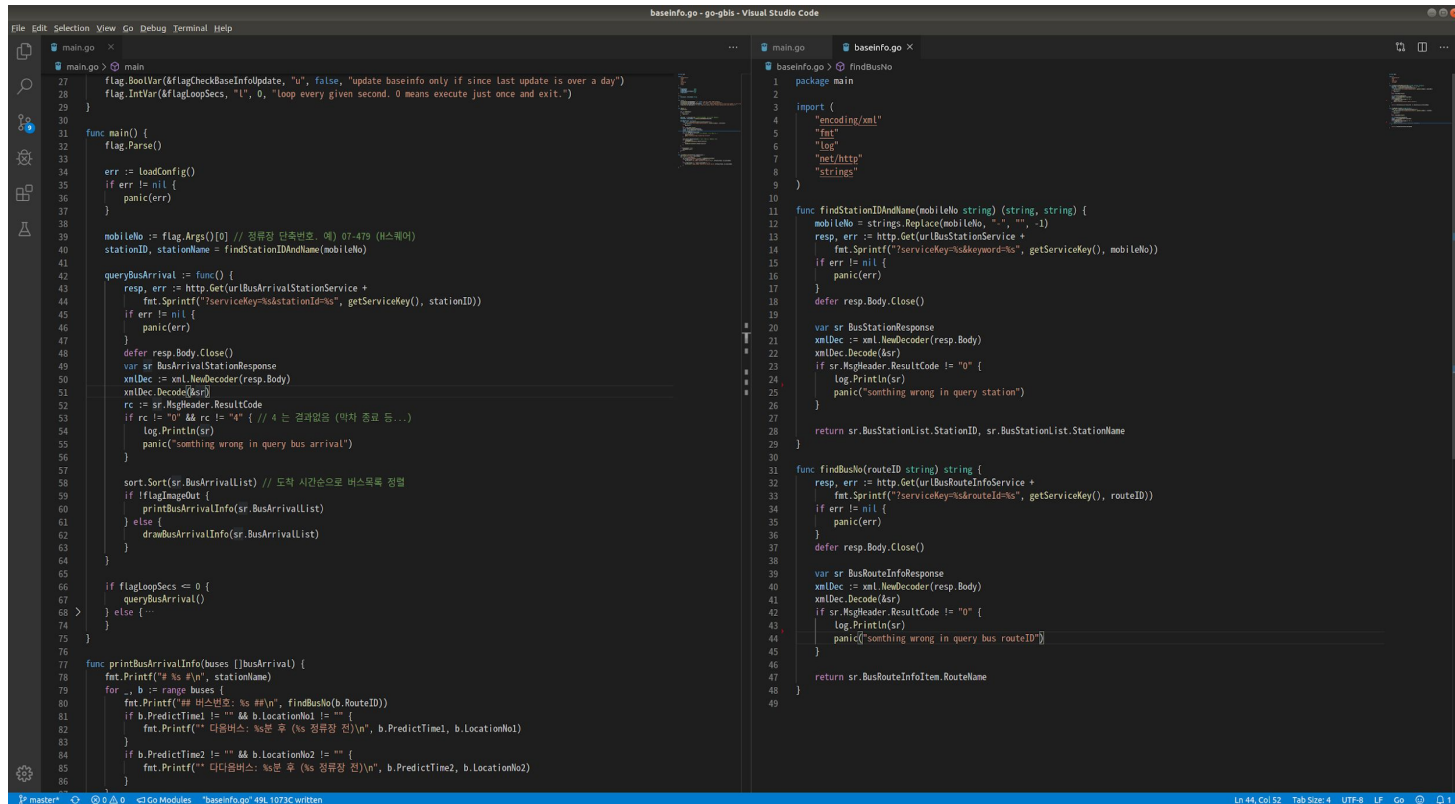
```
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?><response>
2
```

 The 'Go definition here:' section displays the generated Go struct definition: 

```
type Response struct {
    XMLName      xml.Name `xml:"response"`
    Text          string   `xml:",chardata"`
    ComMsgHeader struct {
        Text      string `xml:",chardata"`
        ErrMsg     string `xml:"errMsg"`
        ReturnCode string `xml:"returnCode"`
    } `xml:"comMsgHeader"`
    MsgHeader struct {
        Text      string `xml:",chardata"`
        QueryTime string `xml:"queryTime"`
        ResultCode string `xml:"resultCode"`
        ResultMessage string `xml:"resultMessage"`
    } `xml:"msgHeader"`
    MsgBody struct {
        Text      string `xml:",chardata"`
        BusRouteInfoItem struct {
            Text      string `xml:",chardata"`
            CompanyId string `xml:"companyId"`
            CompanyName string `xml:"companyName"`
            CompanyTel string `xml:"companyTel"`
            DistrictCd string `xml:"districtCd"`
            DownFirstTime string `xml:"downFirstTime"`
            DownLastTime string `xml:"downLastTime"`
            EndMobileNo string `xml:"endMobileNo"`
            EndStationId string `xml:"endStationId"`
            EndStationName string `xml:"endStationName"`
            PeekAlloc string `xml:"peekAlloc"`
            RegionName string `xml:"regionName"`
            RouteId string `xml:"routeId"`
            RouteName string `xml:"routeName"`
        }
    }
}
```

 At the bottom, a footer note reads: 'Made by [Krzysztof Kowalczyk](#). Comments, questions, feedback? [Let me know](#)'.

# 에디터는 VSCode + Go 플러그인 추천 (개취)



```
baseinfo.go - go@bils - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

main.go x
main
27 flag.BoolVar(&flagCheckBaseInfoUpdate, "u", false, "update baseinfo only if since last update is over a day")
28 flag.IntVar(&flagLoopSecs, "l", 0, "loop every given second. 0 means execute just once and exit.")
29 }
30
31 func main() {
32     flag.Parse()
33
34     err := LoadConfig()
35     if err != nil {
36         panic(err)
37     }
38
39     mobileNo := flag.Args()[0] // 정류장 단속번호. 예) 01-479 (H스퀘어)
40     stationID, stationName = findStationIDAndName(mobileNo)
41
42     queryBusArrival := func() {
43         resp, err := http.Get(urlBusArrivalStationService +
44             fmt.Sprintf("?serviceKey=%s&stationId=%s", getServiceKey(), stationID))
45         if err != nil {
46             panic(err)
47         }
48         defer resp.Body.Close()
49         var sr BusArrivalStationResponse
50         xmlDec := xml.NewDecoder(resp.Body)
51         xmlDec.Decode(&sr)
52         rc := sr.MsgHeader.ResultCode
53         if rc != "0" && rc != "4" { // 4는 결과없음 (백자 종료 등...)
54             log.Println(sr)
55             panic("something wrong in query bus arrival")
56         }
57
58         sort.Sort(sr.BusArrivalList) // 도착 시간순으로 버스목록 정렬
59         if !flagImageOut {
60             printBusArrivalInfo(sr.BusArrivalList)
61         } else {
62             drawBusArrivalInfo(sr.BusArrivalList)
63         }
64     }
65
66     if flagLoopSecs == 0 {
67         queryBusArrival()
68     } else {
69         // ...
70     }
71 }
72
73 func printBusArrivalInfo(buses []BusArrival) {
74     fmt.Printf("%s %s\n", stationName)
75     for _, b := range buses {
76         fmt.Printf("# 버스번호: %s #\n", findBusNo(b.RouteID))
77         if b.PredictTime1 != "" && b.LocationNo1 != "" {
78             fmt.Printf("다음버스: %s분 후 (%s 정류장 전)\n", b.PredictTime1, b.LocationNo1)
79         }
80         if b.PredictTime2 != "" && b.LocationNo2 != "" {
81             fmt.Printf("다음버스: %s분 후 (%s 정류장 전)\n", b.PredictTime2, b.LocationNo2)
82         }
83     }
84 }
85
86
baseinfo.go x
baseinfo
1 package main
2
3 import (
4     "encoding/xml"
5     "fmt"
6     "log"
7     "net/http"
8     "strings"
9 )
10
11 func findStationIDAndName(mobileNo string) (string, string) {
12     mobileNo = strings.Replace(mobileNo, "-", "", -1)
13     resp, err := http.Get(urlBusStationService +
14         fmt.Sprintf("?serviceKey=%s&keyword=%s", getServiceKey(), mobileNo))
15     if err != nil {
16         panic(err)
17     }
18     defer resp.Body.Close()
19
20     var sr BusStationResponse
21     xmlDec := xml.NewDecoder(resp.Body)
22     xmlDec.Decode(&sr)
23     if sr.MsgHeader.ResultCode != "0" {
24         log.Println(sr)
25         panic("something wrong in query station")
26     }
27
28     return sr.BusStationList.StationID, sr.BusStationList.StationName
29 }
30
31 func findBusNo(routeID string) string {
32     resp, err := http.Get(urlBusRouteInfoService +
33         fmt.Sprintf("?serviceKey=%s&routeId=%s", getServiceKey(), routeID))
34     if err != nil {
35         panic(err)
36     }
37     defer resp.Body.Close()
38
39     var sr BusRouteInfoResponse
40     xmlDec := xml.NewDecoder(resp.Body)
41     xmlDec.Decode(&sr)
42     if sr.MsgHeader.ResultCode != "0" {
43         log.Println(sr)
44         panic("something wrong in query bus routeID")
45     }
46
47     return sr.BusRouteInfoItem.RouteName
48 }
49
```

# HOW TO BUILD A MINIMUM VIABLE PRODUCT

## NOT LIKE THIS

---



1

2

3

4

## LIKE THIS

---



1

2

3

4

5

image by [blog.fastmonkeys.com](http://blog.fastmonkeys.com) original idea: spotify product team

# 기본 패키지들로 CLI 프로그램 작성

버스 정류장 번호(단축번호)를 넣으면  
버스들의 도착정보를 마크다운 형식으로  
출력하도록 만들어 봄.

- 예의 H스퀘어 정류장의 단축번호,  
07479는 직접 버스정류장에서  
확인해 보거나 카카오맵 등에서 버스  
정류장을 찍어보면 알 수 있음.

```
suapapa@suapapa-B70SH-AJ76B2W: ~/ws/suapapa/go-gbis
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
suapapa@suapapa-B70SH-AJ76B2W:~/ws/suapapa/go-gbis$ ./go-gbis 07479
# H스퀘어 #
## 버스번호: 390 ##
* 다음버스: 5분 후 (3 정류장 전)
## 버스번호: 9007 ##
* 다음버스: 5분 후 (4 정류장 전)
## 버스번호: 101 ##
* 다음버스: 24분 후 (20 정류장 전)
* 다다음버스: 41분 후 (31 정류장 전)
suapapa@suapapa-B70SH-AJ76B2W:~/ws/suapapa/go-gbis$
```

# 외부 패키지 사용 (module)

사용하는 시점의 외부 패키지를 박제하기 위해 다음과 같이 모듈을 초기화 함.

- go mod init [github.com/suapapa/gbis-frame](https://github.com/suapapa/gbis-frame)
- go.mod, go.sum 추가 후 빌드하여 내용 갱신 필요

참고

- <https://github.com/golang/go/wiki/Modules>

# 외부패키지 사용 - 이미지 파일로 출력 (gg)

H스퀘어

 390

다음버스  
6분 후 (5 전)

 9007

다음버스  
16분 후 (14 전)

 101

다음버스  
18분 후 (14 전)  
다다음버스  
42분 후 (31 전)

- <https://github.com/fogleman/gg>
- 이미지 출력을 위한 플래그 추가: -i
- draw.go에 관련 코드 분리
- 좌표에 가이드 라인 그려가며 수작업으로 배치;;

리소스:

- 배민 도현체: <http://font.woowahan.com/dohyeon/>
- Icons - Marerial Design, 버스 아이콘:  
<https://material.io/resources/icons/>
  - Inkscape로 svg=>png 변환 후 사용

# 외부 패키지 사용 - 리소스 내장하기 (go-bindata)

- <https://github.com/go-bindata/go-bindata>
- `go get -u github.com/go-bindata/go-bindata/...`
- `go-bindata -o resource.go _resource/`
- draw.go에서 Asset() 함수가 리턴하는 바이트 슬라이스에서 바로 리소스를  
사요하도록 코드 수정
- 바이너리 사이즈
  - -200KB (원 리소스 사이즈: 500KB → 변환 후: 300KB)
- 실행속도
  - +2secs in RPi B+

e-Paper에 그리기



# 재료 - 라즈베리파이

## OS설치

- SD카드 (4기가면 충분)
- Raspbian Buster Lite
  - <https://www.raspberrypi.org/downloads/raspbian/>

## 최초 설정 시 다음의 추가 장비를 추천

- HDMI 모니터
- USB 키보드

### Raspberry Pi Boards



**Raspberry Pi 4 Model B**

Your tiny, dual-display, desktop computer

[More info >](#)



**Raspberry Pi 3 Model A+**

Our third-generation single-board computer, now in the A+ format

[More info >](#)



**Raspberry Pi 3 Model B+**

The final revision of our third-generation single-board computer

[More info >](#)



**Raspberry Pi 3 Model B**

Our third-generation single-board computer

[More info >](#)



**Raspberry Pi 2 Model B**

The Raspberry Pi 2 Model B is the second-generation Raspberry Pi

[More info >](#)



**Raspberry Pi 1 Model B+**

The Model B+ is the final revision of the original Raspberry Pi

[More info >](#)



**Raspberry Pi 1 Model A+**

The Model A+ is the low-cost variant of the Raspberry Pi

[More info >](#)



**Raspberry Pi Zero W**

Single-board computer with wireless and Bluetooth connectivity

[More info >](#)

# 라즈베리파이 설정

- 키보드 설정이 영국식(gb)로 되어 있음 미국식으로 바꿀것(us)
  - `/etc/default/keyboard`
    - `XKBLAYOUT="us"`
- `sudo raspi-config`
  - network => Wi-fi => SSID/password 입력
  - Interface => ssh => yes
  - Interface => spi => yes
- SPI 버퍼 늘리기 4096 -> 48000 (= 800\*480/8)
  - `/boot/cmdline.txt` 앞에 추가
    - `spidev.bufsiz=48000`
  - `cat /sys/module/spidev/parameters/bufsiz`

# RPi에서 돌아갈 고 프로그램 빌드 및 설치

빌드 (arm-linux 타겟으로 크로스컴파일):

- GOARCH=arm GOOS=linux go build

설치는 파일 하나만 밀어넣으면 끝:

- scp gbis-frame pi@192.168.0.5:~/

# 재료 - e-Paper패널 (epd7in5 V2)

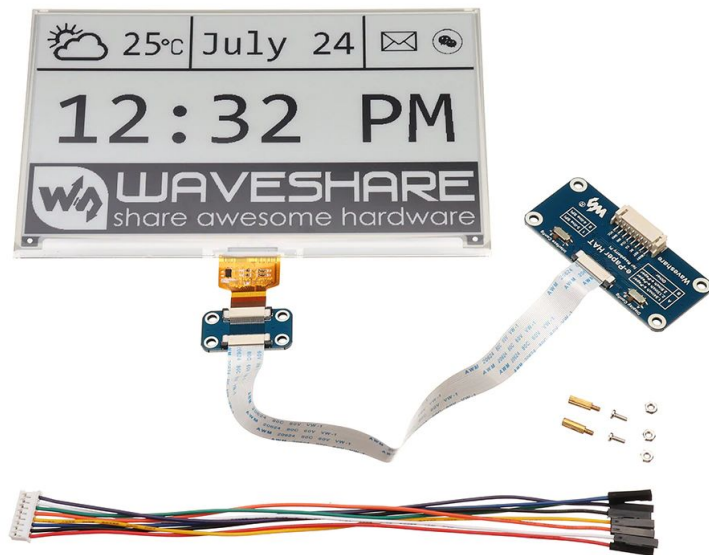
- 리소스:

[https://www.waveshare.com/wiki/7.5inch\\_e-Paper\\_HAT](https://www.waveshare.com/wiki/7.5inch_e-Paper_HAT)

- 해상도:

- ~~640x384(v1)~~
- 800x480(v2)

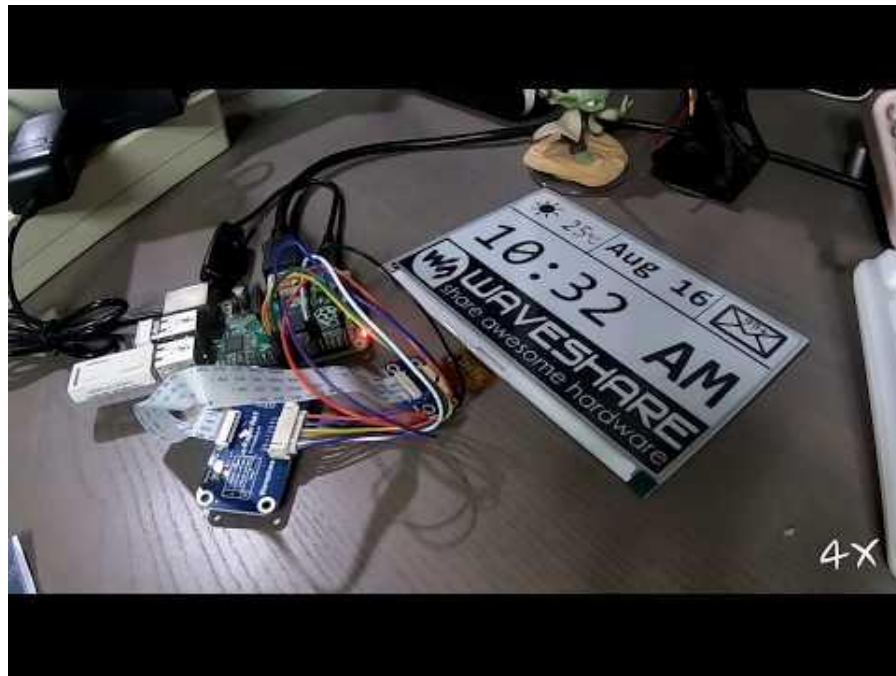
- 인터페이스: SPI



# e-Paper 테스트 - 파이썬

모듈 업체에서 제공해주는 파이썬으로 된 테스트 스크립트로 디스플레이가 작동하는지 봅니다.

- <https://github.com/waveshare/e-Paper>
- python3  
RaspberryPi&JetsonNano/python/examples/epd\_7in5\_V2\_test.py



4X

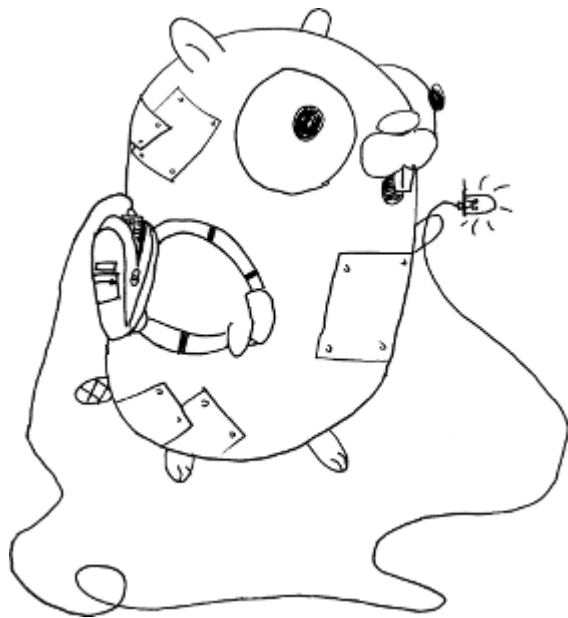
# 외부 패키지 사용 - e-Paper 패키지 작성

파이썬 코드를 고로 옮김:

- ~~github.com/suapapa/go\_devices/epd7in5~~
- github.com/suapapa/go\_devices/epd7in5v2

참고:

- <https://periph.io/>



# e-Paper에 그리기

라이브러리를 잘 준비해서 실제로 그리는 코드는 무척 단순합니다.

- `dev, _ := epd7in5v2.NewSPIHat(s)`
- `img := dc.Image() // gg 패키지의 draw context => image.Image`
- `dev.Draw(img.Bounds(), img, image.ZP)`

e-Paper에 그리는 기능을 위해 flag, -e 를 추가했습니다.

- `./gbis-frame -e 07479`

데몬화

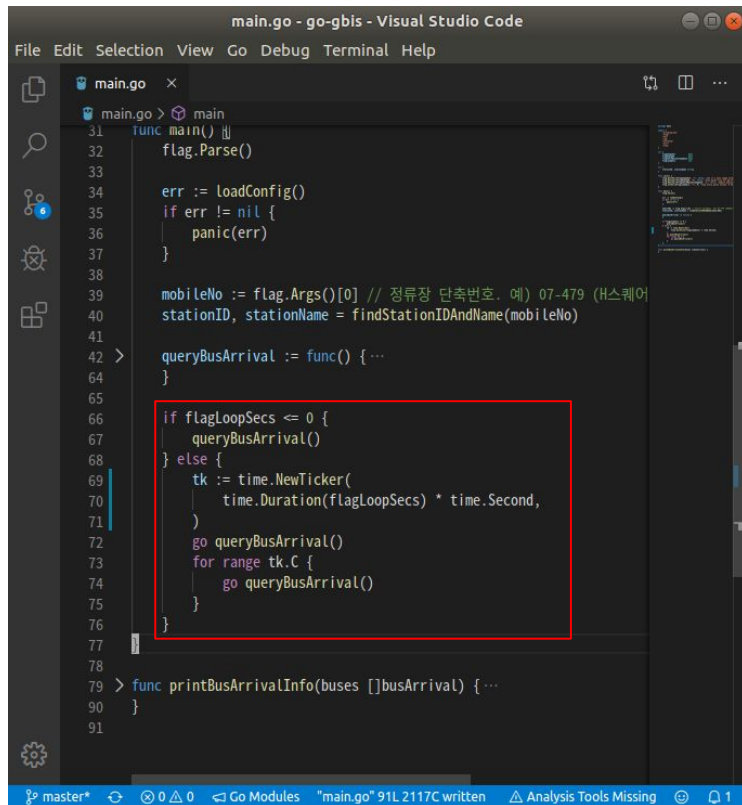


# 죽지 않게 하기

time.Ticker를 사용해 일정  
시간마다 주기적으로 패널을  
갱신하도록 함.

플래그 -l 추가. 예) 60초마다  
패널에 출력

- ./gbis-frame -e -l 60  
07479



```
main.go - go-gbis - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

main.go x
main.go > main
31 func main() {
32     flag.Parse()
33
34     err := loadConfig()
35     if err != nil {
36         panic(err)
37     }
38
39     mobileNo := flag.Args()[0] // 정류장 단축번호. 예) 07-479 (H스퀘어)
40     stationID, stationName = findStationIDAndName(mobileNo)
41
42     queryBusArrival := func() { ...
43     }
44
45     if flagLoopSecs <= 0 {
46         queryBusArrival()
47     } else {
48         tk := time.NewTicker(
49             time.Duration(flagLoopSecs) * time.Second,
50         )
51         go queryBusArrival()
52         for range tk.C {
53             go queryBusArrival()
54         }
55     }
56
57     func printBusArrivalInfo(buses []busArrival) { ...
58     }
59 }
```

master 0 0 0 Go Modules "main.go" 91L 2117C written Analysis Tools Missing 1

# 부팅 시 실행 - Linux systemd service 설정

## register

```
$ sudo vi /lib/systemd/system/gbis-frame.service
[Unit]
Description=gbis panel
[Service]
Type=simple
Restart=always
RestartSec=5s
ExecStart=/home/pi/gbis-frame -l 60 -e 07479
[Install]
WantedBy=multi-user.target
```

## log

```
$ tail -f /var/log/syslog
```

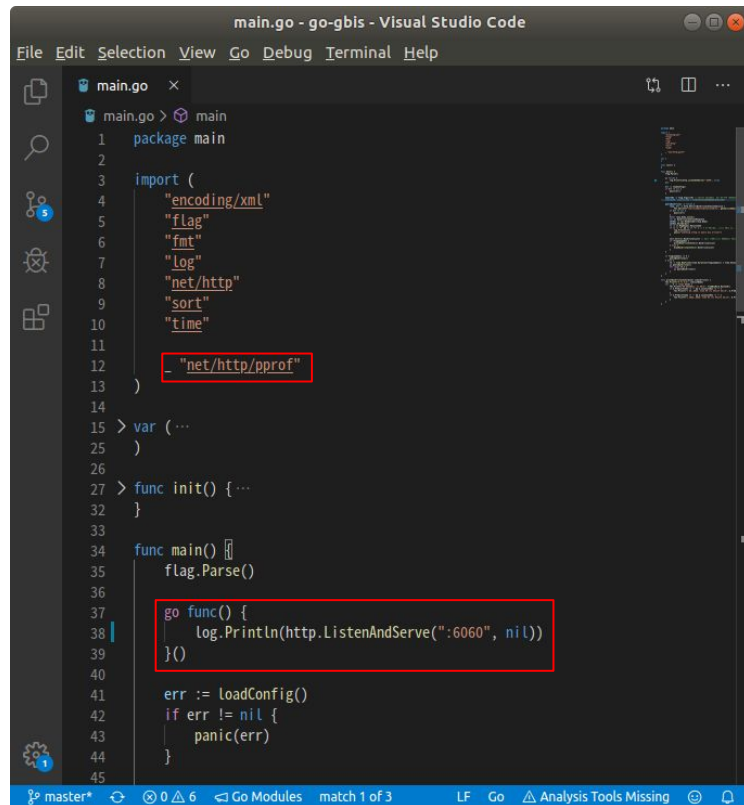
## enable

```
$ sudo systemctl enable gbis-frame
```

프로파일링

# 프로파일링 1/2

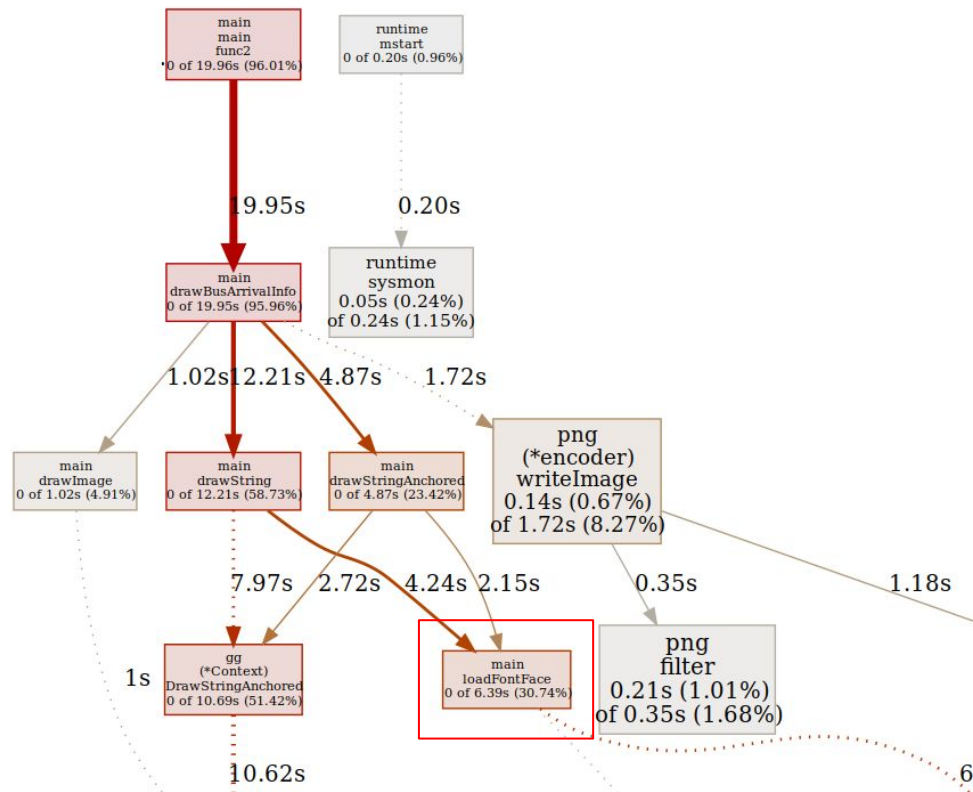
- net/http/pprof
- <https://golang.org/pkg/net/http/pprof/>
- T> gbis-frame -l 30 -i out.png
- H> go tool pprof -http=:8080  
http://192.168.0.5:6060/debug/pprof/profile?seconds=60



```
main.go - go-gbis - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

main.go x
main.go > main
1 package main
2
3 import (
4     "encoding/xml"
5     "flag"
6     "fmt"
7     "log"
8     "net/http"
9     "sort"
10    "time"
11
12    _ "net/http/pprof"
13 )
14
15 > var ( ...
16 )
17
18 > func init() { ...
19 }
20
21 func main() {
22     flag.Parse()
23
24     go func() {
25         log.Println(http.ListenAndServe(":6060", nil))
26     }()
27
28     err := loadConfig()
29     if err != nil {
30         panic(err)
31     }
32 }
33
34 master* 0 6 Go Modules match 1 of 3 LF Go Analysis Tools Missing
```

## 프로파일링 2/2

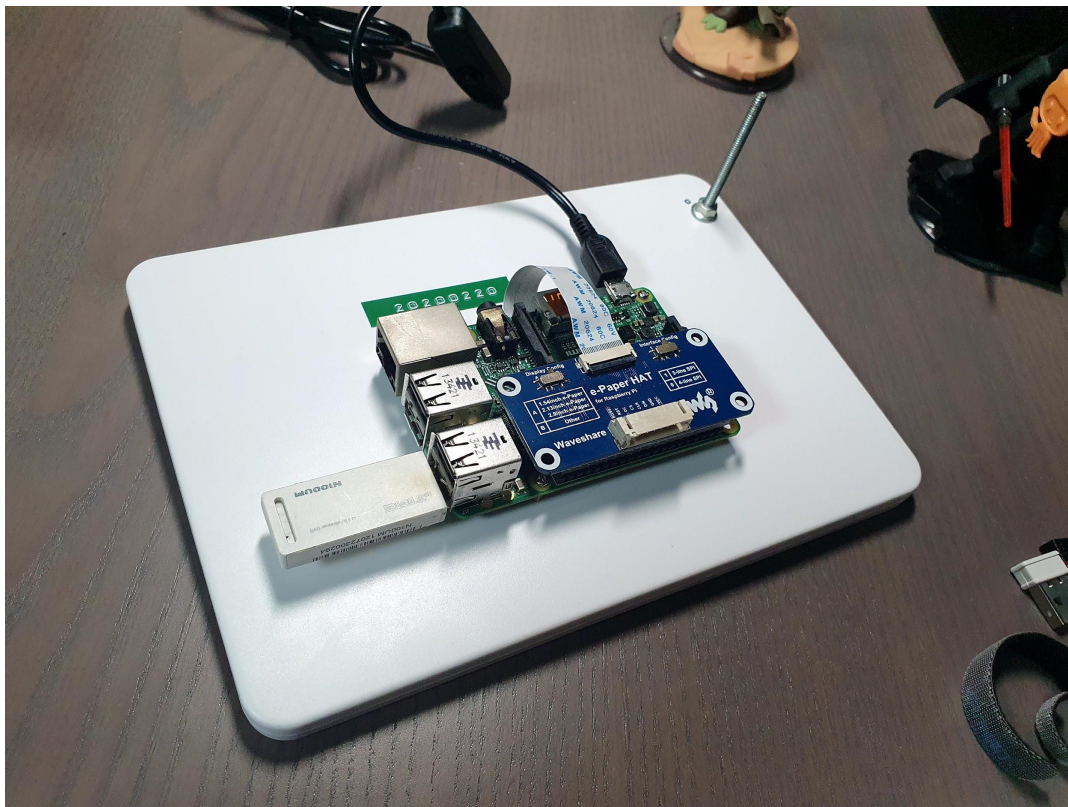


마무리

# 케이스에 넣어 완성



# 뒷모습



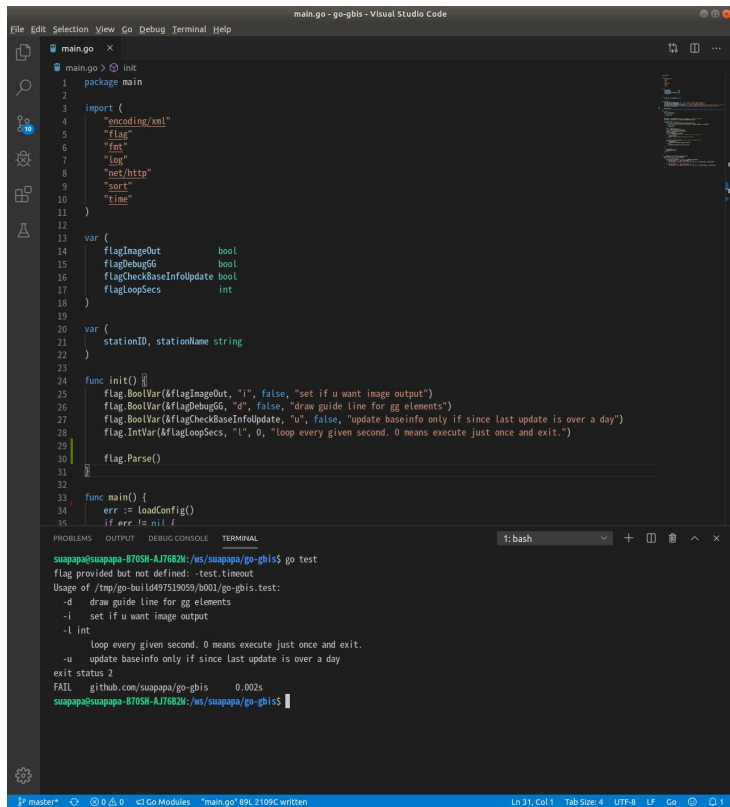


## 결과물 - 동영상



뱀다리

# 헤매던 부분 - flag.Parse() 버그?



```
1 package main
2
3 import (
4     "encoding/xml"
5     "flag"
6     "log"
7     "net/http"
8     "sort"
9     "time"
10 )
11
12
13 var (
14     flagImageOut    bool
15     flagDebugGG     bool
16     flagCheckBaseInfoUpdate bool
17     flagLoopSecs    int
18 )
19
20 var (
21     stationID, stationName string
22 )
23
24 func init() {
25     flag.BoolVar(&flagImageOut, "i", false, "set if u want image output")
26     flag.BoolVar(&flagDebugGG, "d", false, "draw guide line for gg elements")
27     flag.BoolVar(&flagCheckBaseInfoUpdate, "u", false, "update baseinfo only if since last update is over a day")
28     flag.IntVar(&flagLoopSecs, "l", 0, "loop every given second. 0 means execute just once and exit.")
29
30     flag.Parse()
31 }
32
33 func main() {
34     err := loadConfig()
35     if err != nil {
36         log.Fatal(err)
37     }
38 }
39
40 func loadConfig() error {
41     // ...
42 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
1: bash
$ go test
flag provided but not defined: -test.timeout
Usage of /tmp/go-build497519059/b001/go-gbis.test:
  -d draw guide line for gg elements
  -i set if u want image output
  -l int
      loop every given second. 0 means execute just once and exit.
  -u update baseinfo only if since last update is over a day
exit status 2
FAIL    github.com/suapapa/go-gbis    0.002s
$
```

go test 시 이상한 flag 오류 발생

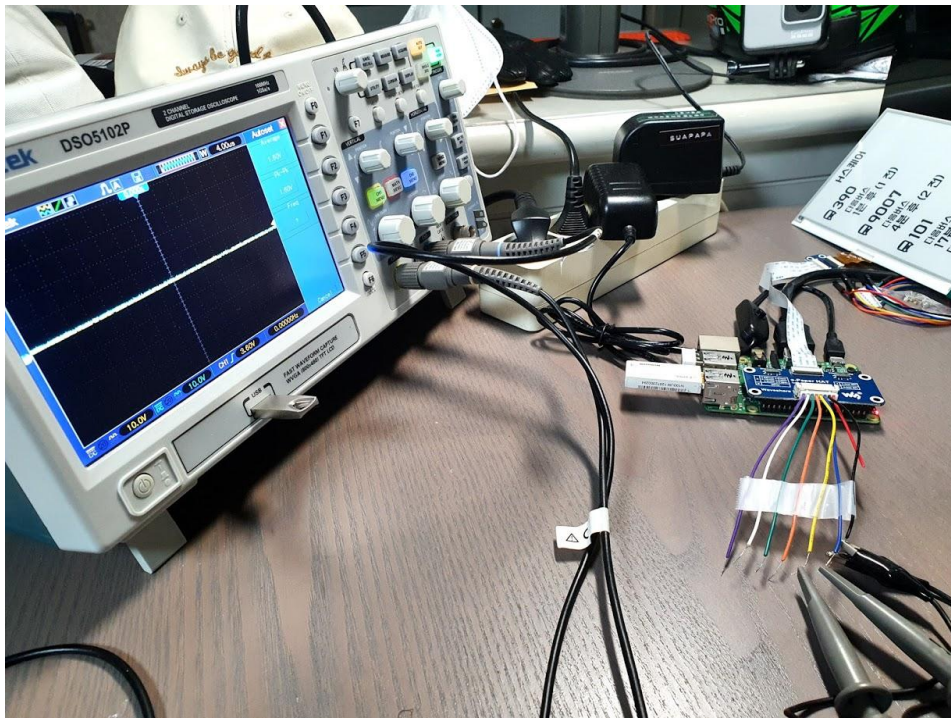
해결:

- flag.Parse() 가 init()에 있으면 안됨.  
main()에 있어야 함

# 라이브러리가 안 될 때는 소스를 봅시다;

테스트 파이썬 스크립트에서 동작하던 것이 고에선 동작하지 않음.

- 동일 동작의 프로그램을 각 언어로 만듦.
- 비교분석...
- 라이브러리를 열어보니 이미 deprecated 되고 새 저장소로 옮겼다는 주석;
  - [golang.org/x/exp/io](http://golang.org/x/exp/io)
  - [periph.io/x/periph](http://periph.io/x/periph)



# Go 1.13 부터 추가되서 좋았던 부분

- 숫자를 쓸 때 \_ 로 가독성을 높일 수 있음
  - `annualSalary := 100_000_000`
- 바이너리 형식으로도 숫자 표현이 가능해짐
  - `0x4C == 0b0100_1100`

감사합니다