

How we use Golang in heybeauty

헤이뷰티 주원영

getogrand@gmail.com / GitHub @getogrand

발표자 소개

- UI Developer (implements Serverable)
- API 서버 리뉴얼
 - RoR ➡ Golang
- iOS 앱 리뉴얼 (지금 바로 앱스토어 'HEYBEAUTY' 검색 & 설치!)
 - Native (Obj-C) ➡ React SPA in WebView (Swift)
- Android 앱 리뉴얼 (Ongoing...)
 - Native (Java) ➡ React SPA in WebView (Kotlin)

헤이뷰티

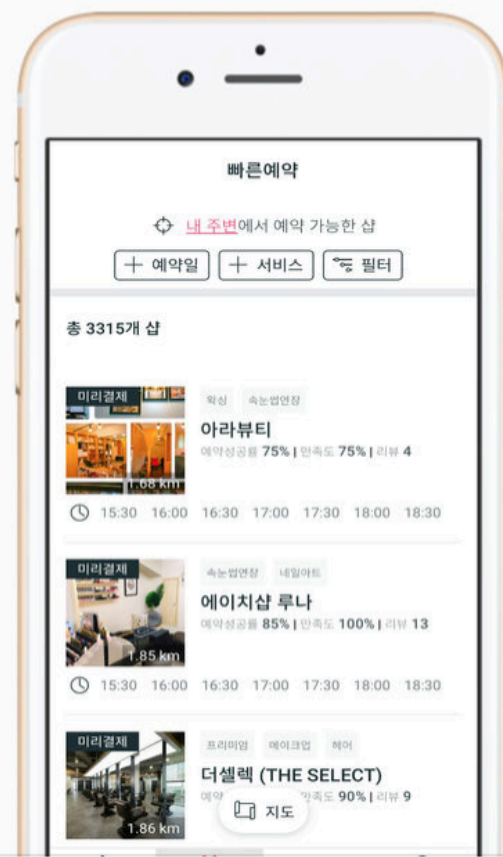
- 뷰티샵 (네일, 헤어, 마사지, 왁싱 등) 예약 서비스
- 전화 없이, 손쉽게 예약!

완전히 새로워진 디자인



빠른예약

빠르게 원하는 샵을 찾아 예약까지!



서비스 선택



가격 범위

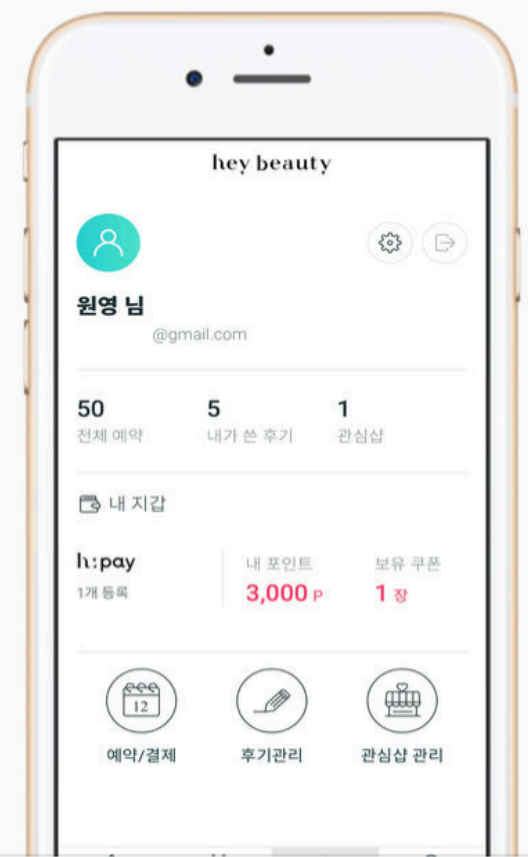
15,000 원 ~ 75,000 원

적용하기

원하는 가격 범위, 우수 매장, 태그를 골라 샵을 볼 수 있어요

hpay로 간편결제

카드를 등록하지 않아도 결제할 수 있도록 일반결제 기능이 추가되었어요



“우릴 속였어. 이건 광고잖아!”

-Homer Simpson

hey beauty

이제 진짜 시작합니다.

“I might be wrong”

-Radiohead

hey beauty

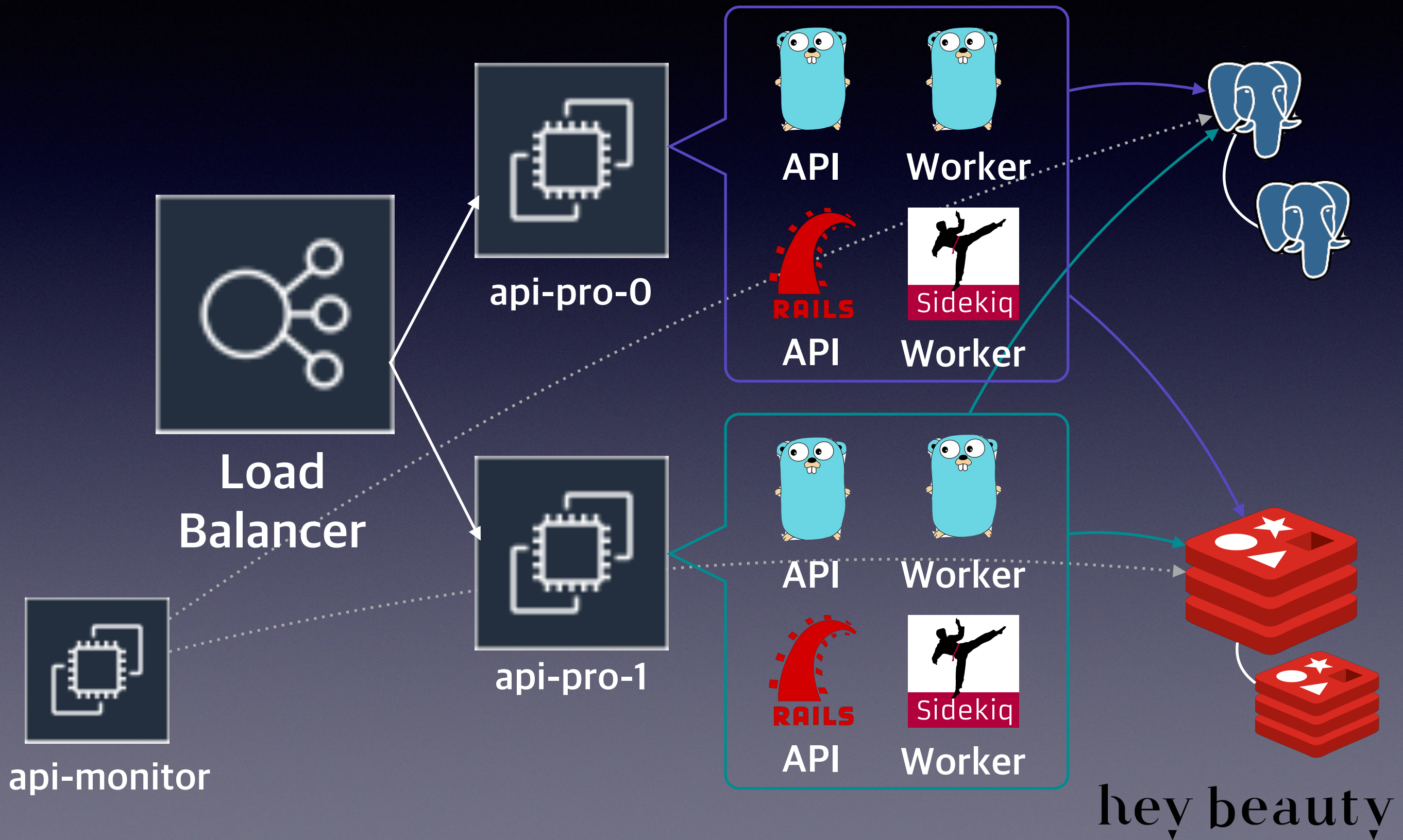


당신의 눈(과 지식)으로
제 발표를 무시하세요.

헤이뷰티 서비스

- 미리결제
 - 신용카드
 - hpay (간편결제)
 - 일반결제
- 쿠폰
- 포인트
- 방문결제
- 예약
 - 승인 대기
 - 승인 완료
 - 서비스 완료
 - 취소
 - 임박 취소 패널티 부과
 - 노쇼
 - 패널티 부과

헤이뷰티의 서버 구성



“가장 보통의 존재”

-언니네 이발관

hey beauty

가장 보통의 서비스.

hey beauty

가장 보통의 서버 구성.

가장 보통의 API.

hey beauty

제일 중요하고, 제일 어려운 것.

Interlude

hey beauty

배포

- Lint
- Test
- Build
- Load Balancing
- Upload
- Daemonize
- Clean

\$ make production

헤이뷰티의 Makefile을 통한 배포 자동화

<https://asciinema.org/a/221510>



{17:01}[2.6.0]~/go/src/github.com/hey-beauty/server-api-up:master ✓ ↻ make production

make lint test

🔍 Check Dependencies for Makefile

✅ Dependencies for Makefile are fulfilled

📁 Lint Source Files

golangci-lint run

📁 Run Tests

go test -v .

make DEPLOY=pro0 deploy-nolint-notest

🔍 Check Dependencies for Makefile

✅ Dependencies for Makefile are fulfilled

📁 Build API Server Executable

GOOS=linux go build -o hbapi main.go

📁 Build hop Utility Executable

GOOS=linux go build -o hop ./cmd/hop/main.go

📁 Deregister 'pro0' from ELB

aws elbv2 deregister-targets \

📁 Stop API Server

ssh api-pro-0 "sudo systemctl stop hbapid"

📁 Upload API Server Executable

scp -r ./hbapi api-pro-0:~/server/hbapi
hbapi

📁 Upload Environment File

scp -r ./production.env api-pro-0:~/server/productio
production.env

📁 Upload hop Utility Executable

scp -r ./hop api-pro-0:~/server/hop
hop

📁 Build Worker Executable

GOOS=linux go build -o hbworker worker/cmd/main.go

📁 Stop Worker

ssh api-pro-0 "sudo systemctl stop hbworkerd"

📁 Upload Worker Executable

scp ./hbworker api-pro-0:~/server/hbworker
hbworker

📁 Start Worker

ssh api-pro-0 "sudo systemctl start hbworkerd"

📁 Start API Server

ssh api-pro-0 "sudo systemctl start hbapid"

📁 Register 'pro0' to ELB

aws elbv2 register-targets \

📁 Clean Output Files

rm ./hbapi
rm ./hop
rm ./hbworker

make DEPLOY=pro1 deploy-nolint-notest

hey beauty

\$ make production

- **\$ make lint**
- **\$ make test**
- \$ DEPLOY=pro1 make setup
- \$ DEPLOY=pro1 make restart
- \$ DEPLOY=pro1 make stop
- \$ DEPLOY=pro1 make run

Makefile 후기

- 장점

- 모든 linux / macOS 에 기본 탑재 및 호환
- Golang 세계의 컨벤션 (그저 빛 ✨ T.J. ✨ 그저 빛)

- 단점

- 생각보다 어렵다
- 문서도 읽기 어려운 편
- 아는 사람이 별로 없다 (뇌피셜)
- 스크립트가 아니라 매크로에 가깝다

Makefile 후기

- 다음 번에는?
 - 전 안 쓸거예요
- 대안들
 - <https://github.com/markbates/grift>
 - <https://github.com/magefile/mage>

Interlude

hey beauty

헤이뷰티의 API 서버 어플리케이션 아키텍처

주요 프레임워크/라이브러리

- Web framework: github.com/labstack/echo
- ORM: <https://github.com/go-pg/pg>
- Worker: <https://github.com/RichardKnop/machinery>

주요 프레임워크/라이브러리 후기

- Web framework: github.com/labstack/echo
 - 아주 좋음
- ORM: <https://github.com/go-pg/pg>
 - 쓰지 마세요
- Worker: <https://github.com/RichardKnop/machinery>
 - 쏘쏘

주요 프레임워크/라이브러리 후기

- 그냥 buffalo 쓸 걸...
- <https://github.com/gobuffalo/buffalo>
- Golang on Rails
- Routing, Templating, Testing, Hot Code Reload, Frontend Pipeline, ORM, Worker, Toolbox(scaffold generator), etc...
- 이렇게 좋은 거 있는지 몰랐어요 ㅠㅠ

“I have no interest in playing the benchmark game, and neither should you.”

-Mark Bates (Author of buffalo framework)



























디렉토리 구성

 main.go
 main_of_test.go

 routes

 handlers

 models



-  alimtalk
-  aws
-  cmd/hop
-  config
-  fixtures
-  handlers
-  httpclient
-  iamport
-  infobip
-  kakao
-  logger
-  migrations
-  models
-  notification
-  redis
-  reqctx
-  routes
-  scratch
-  setup
-  slack
-  sms
-  smsauth
-  systemd
-  utils
-  vendor
-  worker

hey beauty

Models as One Package

- alimtalk_token.go
- assigned_coupon.go
- assigned_coupon_test.go
- balance.go
- balance_test.go
- balance_transaction.go
- balance_transaction_test.go
- blacklist.go
- block_time.go
- block_time_test.go
- brand.go
- brand_test.go
- call_log.go
- call_log_test.go
- campaign.go
- category.go
- category_test.go
- color.go
- color_test.go
- content.go
- content_test.go
- contract_plan.go
- coupon.go
- coupon_test.go
- curation.go
- curation_test.go
- customer.go
- customer_test.go
- design.go
- design_portfolio.go
- device.go
- device_test.go
- employee.go
- employee_review.go
- employee_test.go
- employment.go
- employment_schedule.go
- employment_test.go
- employment_timeslot.go
- faq.go
- gift_certificate.go
- gift_certificate_test.go
- go_pg_test.go
- interest_free_policy.go
- main_service.go
- marketing_plan.go
- models.go
- models_test.go
- naver_agreement.go
- package.go
- packaged_service.go
- partner_coupon.go
- payment.go
- payment_method.go
- payment_test.go
- peon.go
- plan_payment.go
- portfolio.go
- portfolio_tag.go
- portfolio_tagging.go
- portfolio_test.go
- post.go
- pricing.go
- product.go
- product_type.go
- purchased_package.go
- question.go
- received_sms.go
- region.go
- registered_invitation.go
- registered_invitation_test.go
- reservation.go
- reservation_notification.go
- reservation_notification_alimtalk_t
- reservation_notification_factory.go
- reservation_test.go
- reservations_timeslot.go
- schedule.go
- service.go
- service_category.go
- service_tag.go
- service_tagging.go
- settlement.go
- shop.go
- shop_pricing.go
- shop_review.go
- shop_search.go
- shop_tag.go
- shop_tagging.go
- shop_test.go
- sns_blacklist.go
- style.go
- timeslot.go
- transaction.go
- user.go
- user_push.go
- user_test.go
- version.go

Middlewares / Routes

- Bind Request Context: JWT, userID, etc...
- Bind Start Time Context: measuring duration
- Recover Panic
- Remove Trailing Slash
- CORS
- Log Request and Response – Debugging Experience 
-  Masking Required
- Gzip

```
func Route(e *echo.Echo) {  
    e.HTTPErrorHandler = errorHan  
    e.Logger = logger.Logger  
    e.Use(bindRequestContext)  
    e.Use(bindStart)  
    e.Use(middleware.Recover())  
    e.Use(middleware.RemoveTrailingSlash)  
    e.Use(middleware.CORS())  
    e.Use(bindUserContext)  
    skipPaths := []string{"/ping"  
    e.Use(logger.Middleware(skipP  
  
    RouteAlimtalk(e)  
    RouteAssignedCoupon(e)  
    RouteBalance(e)  
    RouteBalanceTransation(e)  
    RouteBrand(e)  
    RouteCuration(e)  
    RouteDevice(e)  
    RouteEmployment(e)  
    RouteMainService(e)  
    RouteMetaInfo(e)  
    RouteMobileApp(e)  
    RoutePayment(e)  
    RoutePaymentMethod(e)  
    RoutePing(e)  
    RouteProduct(e)  
    RouteRegisteredInvitation(e)  
    RouteReservation(e)  
    RouteReview(e)  
    RouteShop(e)  
    RouteSMSAuth(e)  
    RouteUser(e)  
    RouteRRWeb(e)  
  
    e.Use(middleware.Gzip())  
}
```

hey beauty

JSON Serialization on Response

```
type reservationIndexItem struct {
    ID          int32    `json:"id"`
    StartTS     int32    `json:"startTs"`
    Status      string   `json:"status"`
    ServiceAmount int32    `json:"serviceAmount"`
    IsPrepay    bool     `json:"isPrepay"`
    IsReviewed  bool     `json:"isReviewed"`
    IsRebookable bool    `json:"isRebookable"`
    ServiceID   int32    `json:"serviceId"`
    ServiceName string   `json:"serviceName"`
    ShopID      int32    `json:"shopId"`
    ShopName    string   `json:"shopName"`
}

type reservationIndexResult struct {
    TotalCount    int    `json:"totalCount"`
    Reservations []*reservationIndexItem `json:"reservations"`
}
```


Response/Request Serialization/Deserialization

- DTO(Data Transfer Object)
 - <https://martinfowler.com/eaCatalog/dataTransferObject.html>
 - Repetitive but Explicit, Easy to understand and Safe
 - Validate() error

API Documentation

- <https://github.com/lord/slate>
- Just Markdown



SLATE

REPLACE THIS WITH YOUR LOGO
IN SOURCE/IMAGES/LOGO.PNG

Q Search

Introduction

Authentication

App

Assigned Coupon

Balance

Show Balance

Purchase Balance with Register...

Purchase Balance with Instant ...

Add Balance using Gift Certific...

Balance Transaction

Brand

Curation

Device

Employment

Main Services

Mobile App

Payment

Payment Method

Product

Balance

Show Balance

유저의 포인트 현황을 조회하는 API 입니다 .

Responses:

JSON Field	Type	Description
nonRefundableAmount	int32	환불 불가 포인트 (헤이뷰티에서 지급 된 포인트)
refundableAmount	int32	환불 가능 포인트 (유저가 구매한 포인트)
total	int32	유저 총 포인트: 환불 불가 포인트 + 환불 가능 포인트

Purchase Balance with Registered Payment Method

기존에 등록해놓은 카드로 포인트를 구매하는 API 입니다.

할부 기능을 지원합니다. 무이자 할부 정보는 고정적이지 않으며 , 그 때마다 [나이스페이 안내](#)를 참조하세요 .

```
curl 'https://dapi.heybeauty.kr/balances' \  
-X GET \  
-H "Authorization: Bearer eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.e
```


Database Migration

- The 'hop' CLI Utility
 - Wrapped <https://github.com/go-pg/migrations>

```
ubuntu@hb-production-0:~/apps/hb/current$ hop migration
```

디비 마이그레이션 관련 툴들

Usage:

```
hop migration [flags]
hop migration [command]
```

Available Commands:

create	creates a migration file
down	reverts last migration
reset	reverts all migrations
set_version	set version without running migration
up	runs all available migrations
version	shows current db version

Flags:

```
-h, --help  help for migration
```

Use "hop migration [command] --help" for more information about a command.

└─ migrations

└─ sqlmod

- 1_initial.go
- 2_add_arn_to_devices.go
- 3_add_jti_to_users.go
- 4_add_is_ad_agree_to_users.go
- 5_create_registered_invitations
- 6_add_invitation_code_to_user
- 7_add_is_pw_change_required
- 8_add_card_quota_to_payment
- 9_create_interest_free_policies
- 10_add_pay_type_to_reservatio
- 11_remove_is_benefit_provide
- 12_create_marketing_plans.go
- 13_create_contract_plans.go
- 14_create_designs.go
- 15_create_joint_table_designs
- 16_change_shops_expose_to_r

Worker

- worker
 - cmd
 - main.go
 - worker.go

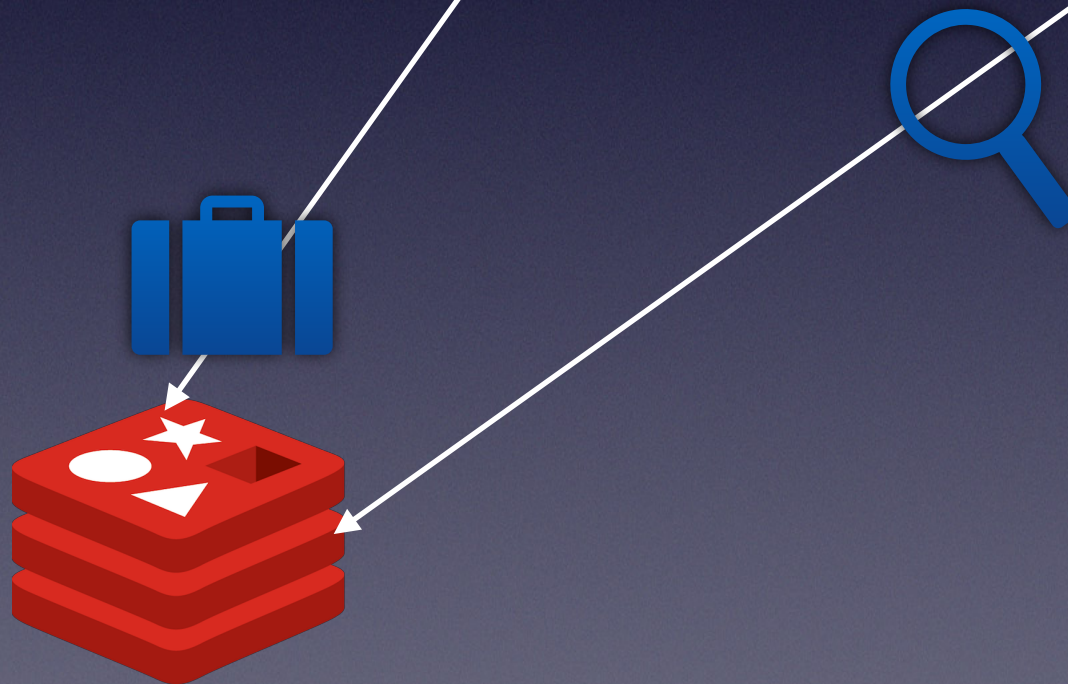
```
func main() {  
    if err := setup.App(); err != nil {  
        log.Fatal(err)  
    }  
    worker.Launch()  
}
```

```
func Launch() {  
    worker := Server.NewWorker("server-api-worker", 8)  
    worker.SetErrorHandler(func(err error) {  
        // make raven client  
        ravenClient, rerr := raven.New(conf.SentryDSN)  
        if rerr != nil {  
            logger.Logger.Errorf("%+v", errors.Wrapf(rerr, "new raven client"))  
            return  
        }  
        ravenClient.CaptureMessageAndWait(fmt.Sprintf("%+v", err), nil, nil)  
    })  
    if err := worker.Launch(); err != nil {  
        log.Fatalf("launch worker: %v", errors.WithStack(err))  
    }  
}
```

hey beauty

Worker

VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
48800	2568	1800	S	0.0	0.1	0:00.09	`- /lib/systemd/systemd --system
327M	44740	12832	S	0.0	2.2	3:36.87	`- /home/ubuntu/server/hbapi
123M	29228	9788	S	6.8	1.4	25:48.77	`- /home/ubuntu/server/hbworker



hey beauty

Scratch

- Fast Feedback Loop ➡ Rapid Development
- Missing REPL or ` \$ rails console `
- Learning Test (TDDBE, 토비의 스프링)
- \$ go run main.go scratch

```
var subcommand string
if len(os.Args) > 1 {
    subcommand = os.Args[1]
}

if subcommand == "scratch" {
    if err := scratch.Run(); err != nil {
        log.Fatal(errors.Wrapf(err, "run scratch"))
    }
} else {
    runServer(conf)
}
```

```
package scratch

import (
    "github.com/davecgh/go-spew/spew"
    "github.com/hey-beauty/server-api-up/utils/krttime"
)

func Run() error {
    spew.Dump(krttime.Now())
    return nil
}
```

Thanks to Daniel Choi, Gunwan Park

hey beauty




Interlude

hey beauty

테스트

라고 쓰고 디펜던시 목킹(mocking)이라고 읽는다

Mocking What?

-  `time.Now()`
-  3rd party API (SNS Auth, SMS, Push, etc...)
-  Database
 - Local Test Database and Fixtures
 - Transaction Rollback

Mocking How?

gomock/mockgen
+
Dependency Injection

“Dependency Injection is not a virtue in Ruby”

-DHH (David Heinemeier Hansen, Author of Ruby on Rails)
(<https://dhh.dk/2012/dependency-injection-is-not-a-virtue.html>)



hey beauty

“Dependency Injection is not a virtue in ~~Ruby~~
Monkey-patchable Language”

Monkey patching in Golang

<https://github.com/bouk/monkey>


```
func TestUser_SendPush(t *testing.T) {
    publishPushAsyncCalled := false

    var d *Device
    monkey.PatchInstanceMethod(
        reflect.TypeOf(d),
        "PublishPushAsync",
        func(_ *Device, data *aws.PlatformMessageData) error {
            publishPushAsyncCalled = true
            return nil
        },
    )
    defer monkey.UnpatchInstanceMethod(reflect.TypeOf(d), "PublishPushAsync")

    var user User
    err := DB.Model(&user).
        Column("user.*", "Devices").
        Where(`"user".id = ?`, 40797).
        Select()
    assert.NoError(t, err)

    err = user.SendPush(&aws.PlatformMessageData{
        Message: "User.SendPush test (/w open_coupon_box action)",
        Action:   aws.PlatformMessageActionOpenCouponBox,
    })
    assert.NoError(t, err)

    assert.True(t, publishPushAsyncCalled)
}
```


Interlude

hey beauty

Error Handling in Golang

Exception
is considered harmful

<https://www.joelonsoftware.com/2003/10/13/13/>

hey beauty

- It's REALLY hard to use well
 - Silence or Kaboom
- Dereliction of Duty as a Programmer

“Errors are values”

-Rob Pike (Co-creator of Golang, UTF-8)

hey beauty

It's just one of cases(flows)

hey beauty


```
if err := doSomething(); err != nil {  
    return err  
}
```

hey beauty

network is unreachable

????????????????????????????????????

hey beauty


```
if err := doSomething(userID); err != nil {  
|   return fmt.Errorf("do something with user %v: %v", userID, err)  
}
```

hey beauty

sms delivery to '01012341234':
network is unreachable

- The Go Programming Language Style
- Must include information to debug
- It works, but...

<https://github.com/pkg/errors>

hey beauty


```
if err := doSomething(userID); err != nil {  
    return errors.Wrapf(err, "do something with user %v", userID)  
}
```

hey beauty


```

alimtalk delivery to [07012341234] status ((*infobip.ResultMessageStatus)(0xc0
011107d0)({
  GroupID: (int32) 5,
  GroupName: (string) (len=8) "REJECTED",
  ID: (int32) 51,
  Name: (string) (len=10) "MISSING_TO",
  Description: (string) (len=20) "Missing destination.",
  Action: (string) (len=19) "Check to parameter."
})
)
github.com/hey-beauty/server-api-up/alimtalk/infobip.(*Client).SendBulk
/Users/wonyoungju/go/src/github.com/hey-beauty/server-api-up/alimtalk/
infobip/infobip.go:122
github.com/hey-beauty/server-api-up/alimtalk/infobip.(*Client).Send
/Users/wonyoungju/go/src/github.com/hey-beauty/server-api-up/alimtalk/
infobip/infobip.go:53
github.com/hey-beauty/server-api-up/alimtalk.Send
/Users/wonyoungju/go/src/github.com/hey-beauty/server-api-up/alimtalk/
alimtalk.go:34
github.com/hey-beauty/server-api-up/alimtalk.SendGOB
/Users/wonyoungju/go/src/github.com/hey-beauty/server-api-up/alimtalk/
alimtalk.go:54
runtime.call64
/usr/local/Cellar/go/1.11.4/libexec/src/runtime/asm_amd64.s:523
reflect.Value.call

```

hey beauty

- Wrap error with stack trace!
- Adding context to an error
- Retrieving the cause of an error
- Go 2 Error Inspection?
 - <https://go.googlesource.com/proposal/+refs/changes/97/159497/3/design/XXXXX-error-values.md>

Using pkg/errors with Sentry

`raven.CaptureErrorAndWait()` is **shit**

hey beauty


```

func errorHandler(errToHandle error, c echo.Context) {
    code, message := utils.CodeAndMessageFromError(errToHandle)
    msg := echo.Map{
        "message": message,
    }
    if err := c.JSON(code, msg); err != nil {
        panic(errors.Wrapf(err, "echo context json response"))
    }

    ////////////////
    // Report Error //
    ////////////////
    // check pre-conditions
    if conf.Env != "prod" {
        fmt.Printf("%+v\n", errToHandle)
        return
    }
    if !(500 <= code && code <= 599) {
        return
    }
    // make raven client
    ravenClient, err := raven.New(conf.SentryDSN)
    if err != nil {
        panic(errors.Wrapf(err, "new raven client"))
    }
    req := c.Request()
    ravenClient.SetHttpContext(raven.NewHttp(req))
    // try to set raven user context
    authHeader := req.Header.Get("Authorization")
    if authHeader != "" {
        token, err := parseTokenFromRequestWithClaims(req)
        if err == nil { // CAUTION: err == nil NOT err != nil
            claims := token.Claims.(*handlers.UserClaims)
            ravenClient.SetUserContext(&raven.User{
                ID: fmt.Sprintf("%d", claims.UserID),
            })
        }
    }
}

ravenClient.CaptureMessageAndWait(fmt.Sprintf("%+v", errToHandle), nil, nil)
}

```


Sentry Shit

- Horrible Documentation
- Horrible API
- Horrible Implementation
- Horrible Issue Handling
 - <https://github.com/getsentry/sentry-javascript/issues/1735>


```

func errorHandler(errToHandle error, c echo.Context) {
    code, message := utils.CodeAndMessageFromError(errToHandle)
    msg := echo.Map{
        "message": message,
    }
    if err := c.JSON(code, msg); err != nil {
        panic(errors.Wrapf(err, "echo context json response"))
    }

    ////////////////
    // Report Error //
    ////////////////
    // check pre-conditions
    if conf.Env != "prod" {
        fmt.Printf("%+v\n", errToHandle)
        return
    }
    if !(500 <= code && code <= 599) {
        return
    }
    // make raven client
    ravenClient, err := raven.New(conf.SentryDSN)
    if err != nil {
        panic(errors.Wrapf(err, "new raven client"))
    }
    req := c.Request()
    ravenClient.SetHttpContext(raven.NewHttp(req))
    // try to set raven user context
    authHeader := req.Header.Get("Authorization")
    if authHeader != "" {
        token, err := parseTokenFromRequestWithClaims(req)
        if err == nil { // CAUTION: err == nil NOT err != nil
            claims := token.Claims.(*handlers.UserClaims)
            ravenClient.SetUserContext(&raven.User{
                ID: fmt.Sprintf("%d", claims.UserID),
            })
        }
    }

    ravenClient.CaptureMessageAndWait(fmt.Sprintf("%+v", errToHandle), nil, nil)
}

```


사용자 친화적인 서버 에러 메시지

login: password validation: 비밀번호가 일치하지 않습니다 ❌
비밀번호가 일치하지 않습니다 ✅

*UserFriendlyError


```
type UserFriendlyError struct {  
    Message string  
}
```

```
func (e *UserFriendlyError) Error() string {  
    return e.Message  
}
```

```
func NewUserFriendlyErrorf(format string, args ...interface{}) *UserFriendlyError {  
    return &UserFriendlyError{  
        Message: fmt.Sprintf(format, args...),  
    }  
}
```



```
func errorHandler(errToHandle error, c echo.Context) {  
    code, message := utils.CodeAndMessageFromError(errToHandle)  
    msg := echo.Map{  
        "message": message,  
    }  
    if err := c.JSON(code, msg); err != nil {  
        panic(errors.Wrapf(err, "echo context json response"))  
    }  
}
```

```
////////////////////  
// Report Error //
```



```
func CodeAndMessageFromError(err error) (int, string) {
    var code int
    var message string
    if httpErr, ok := err.(*echo.HTTPError); ok && httpErr != nil {
        code = httpErr.Code
        switch v := httpErr.Message.(type) {
        case string:
            message = v
        case error:
            message = v.Error()
        }
    } else {
        cause := errors.Cause(err)
        if uferr, ok := cause.(*UserFriendlyError); ok {
            code = http.StatusBadRequest
            message = uferr.Message
        } else {
            code = http.StatusInternalServerError
            message = err.Error()
        }
    }
    return code, message
}
```


Interlude

hey beauty

돈금 없지만,
TDD와는 다르다! TDD와는!

TDD?

- Test Driven Development: By Example – Kent Beck
- <https://dhh.dk//2014/tdd-is-dead-long-live-testing.html>
- <https://www.facebook.com/notes/kent-beck/rip-tdd/750840194948847>
- <https://martinfowler.com/articles/is-tdd-dead/>

- I love pizza != I love Pineapple pizza
- I love hamburger != I love Loxxeria hamburger
- I do write test code != I do TDD

hey beauty

Test with code is not TDD

TDD (Test Driven Development)

Kent Beck's HUUUUUGE marketing shit

Don't get me wrong

hey beauty

TFD (Test First Development)

TDD is dead.
Long live TFD.

I do not TFD 😂

It's fucking hard to learn!

hey beauty

I write test code
(after implementation)

Q&A

hey beauty

One more thing...

“Talk is cheap. Show me the code”

-Linus Torvalds

(<https://lkml.org/lkml/2000/8/25/132>)

hey beauty

Q&A

hey beauty

Thank you

Contact – getogrand@gmail.com

hey beauty

One more thing...

We are hiring!

Taste the Golang in Production 🤪
Send resume to **dev@heybeauty.me**

hey beauty