

Go with Flutter

다이어리 서비스 만들기

박제창 (Dreamwalker)

GDG Golang Korea



Speaker

- 박제창 @Dreamwalker
- Dreamus Company
- **GDG Golang Korea**
- **Flutter Seoul**
- **Github:** [JAICHANGPARK](https://github.com/JAICHANGPARK)



2030 부산세계박람회 유치
SK도 함께 노력하겠습니다



FLO

국내 음악플랫폼 중
최다곡 서비스





핸즈온



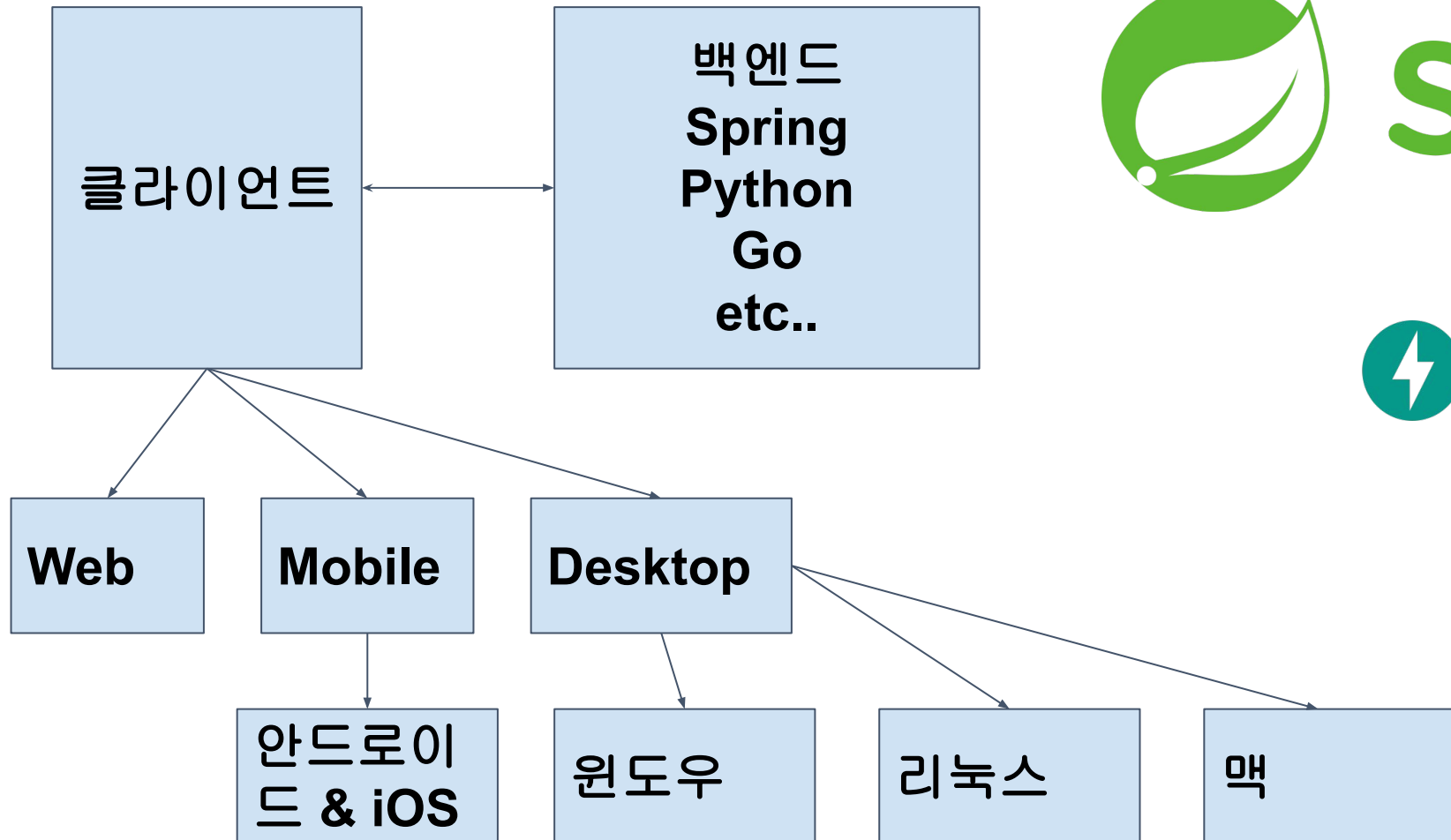


질문
플러터
사용해보신분?
(그럼 Go는...?)



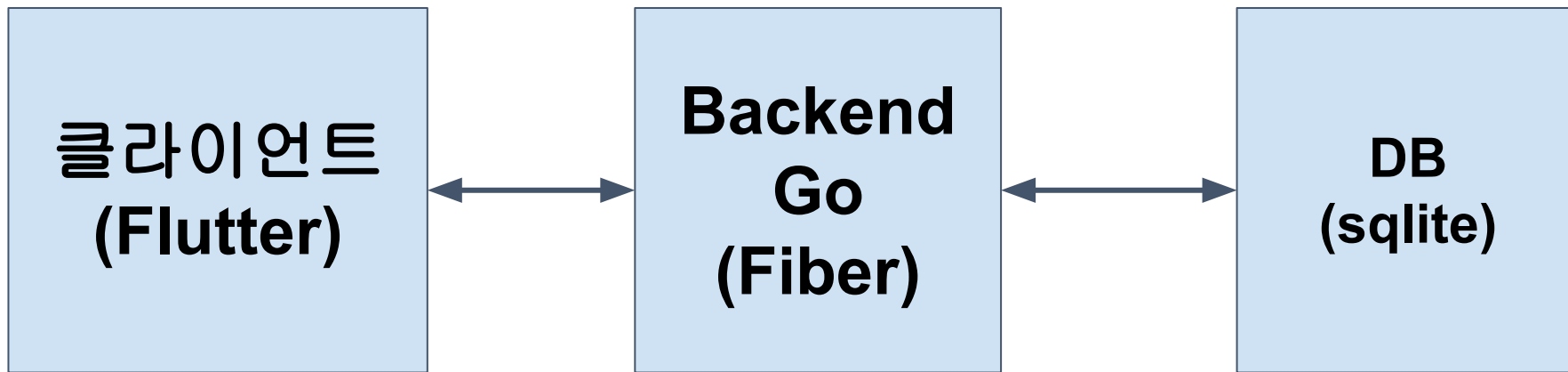
대중적인 서비스..

Go to Busan 2023



어떻게 만들어볼 것인가?

Go to Busan 2023



Flutter



- **CRUD → REST API**
 - Create → POST
 - Read → GET
 - Update → PUT
 - Delete → Delete
- **유사 ToDo List**



Live



jaichang@PARKui-MacBookPro backend %
go version
go version go1.20.4 darwin/arm64

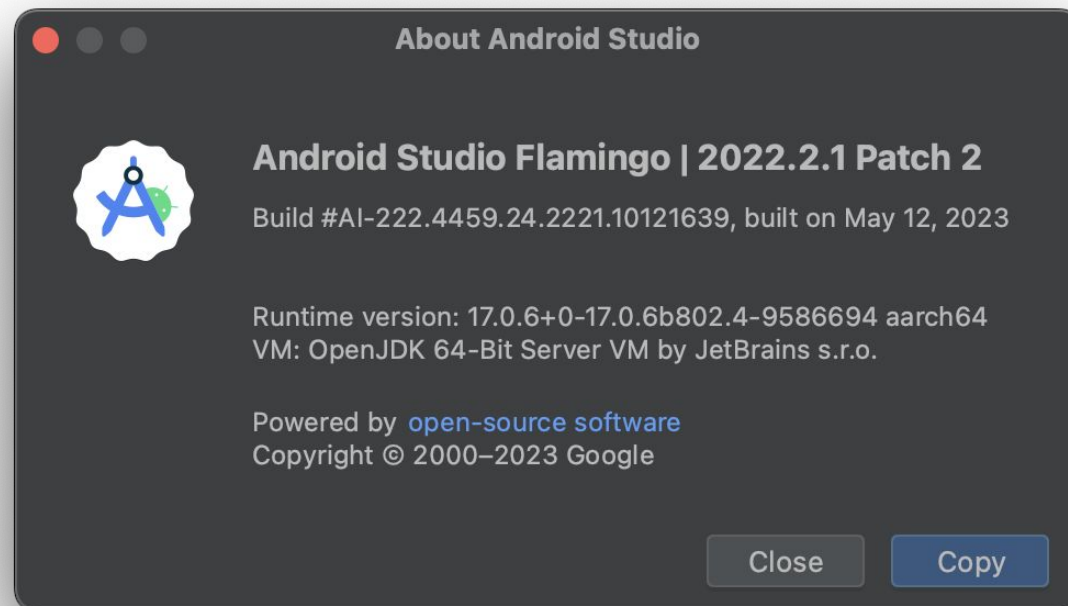
[✓] Flutter (Channel master, 3.11.0-18.0.pre.65, on macOS 13.4 22F66 darwin-arm64, locale ko-KR)

- Flutter version 3.11.0-18.0.pre.65 on channel master at /Users/jaichang/development/flutter
- Upstream repository <https://github.com/flutter/flutter.git>
- Framework revision de368fca94 (22 minutes ago), 2023-06-03 04:19:12 +0800
- Engine revision 02d6fbb68b
- Dart version 3.1.0 (build 3.1.0-160.0.dev)
- DevTools version 2.24.0



[✓] VS Code (version 1.78.2)

Android Studio Flamingo | 2022.2.1 Patch 2



```
import
```

```
"database/sql"
```

```
"fmt"
```

```
"log"
```

```
"strconv"
```


```
"github.com/gofiber/fiber/v2"
```

```
_ "github.com/mattn/go-sqlite3"
```

```
"github.com/projectdiscovery/gologger"
```

```
// myDB 변수 생성
var (
    myDB *sql.DB
)

// 테이블 생성하기
func createTable() {
    query := `
        CREATE TABLE IF NOT EXISTS diary (
            id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
            uuid TEXT,
            note TEXT,
            dt TEXT,
            timestamp INTEGER
        ); `
    if _, err := myDB.Exec(query); err != nil {
        checkErr(err)
    }
}
```



데이터베이스 초기화 및 테이블 생성

```
//sqlite3 drive설정
db, err := sql.Open("sqlite3", "mydb.db")
defer db.Close()
myDB = db
// 테이블 생성하기
createTable()
```

다이어리 모델

```
type Diary struct {  
    Id          string `json:"id" xml:"id" form:"id"`  
    Uuid        string `json:"uuid"`  
    Note        string `json:"note" xml:"note" form:"note"`  
    Dt          string `json:"dt"`  
    Timestamp   int    `json:"timestamp"`  
}
```

다이어리 응답 모델

```
type ResponseModel struct {  
    Code      int      `json:"code"`  
    Message   string   `json:"message"`  
    Diary     []Diary  `json:"items"`  
}
```


Fiber 초기화 및 시작

```
app := fiber.New()  
app.Listen(":3000")
```

다이어리 기록 쓰기

```
app.Post("/diary", func(c *fiber.Ctx) error {
    item := new(Diary)
    //클라이언트로 들어온 Body 데이터 Pasing 하기
    if err := c.BodyParser(item); err != nil {
        return err
    }
    //데이터베이스에 정보 추가하기
    insertDiaryItem(item.Uuid, item.Note, item.Dt, item.Timestamp)
    return c.JSON(item)
})
```

다이어리 기록 쓰기

```
// 데이터베이스에 정보를 추가하기 위한 함수 (Insert)
func insertDiaryItem(uuid string, note string, dt string, timestamp int)
{
    query := `INSERT INTO diary VALUES(NULL,?,?,?,?)`
    if _, err := myDB.Exec(query, uuid, note, dt, timestamp); err != nil
    {
        checkErr(err)
    }
}
```

다이어리 기록 읽기

```
app.Get("/diary", func(c *fiber.Ctx) error {
    search := c.Query("search")
    //클라이언트로 온 검색 쿼리를 기반으로 데이터베이스 정보 조회
    rows, err := myDB.Query("SELECT * FROM diary WHERE Date(dt) = Date(?)", search)
    if err != nil {
        //조회에 오류가 있을 경우 처리
        log.Panicln(err)
    }
    // Slice 생성
    tmps := make([]Diary, 0)
    // 조회 결과 순회
    for rows.Next() {
        var result Diary
        rows.Scan(&result.Id, &result.Uuid, &result.Note, &result.Dt, &result.Timestamp)
        tmps = append(tmps, result)
    }
    return c.JSON(ResponseModel{200, strconv.Itoa(len(tmps)), tmps})
})
```



다이어리 기록 삭제하기

```
app.Delete("/diary", func(c *fiber.Ctx) error {
    uuid := c.Query("uuid")
    log.Println(uuid)
    deleteDiartItem(uuid)
    return c.JSON("200")
})

func deleteDiartItem(uuid string) {
    query := `DELETE FROM diary WHERE uuid=?;`
    if _, err := myDB.Exec(query, uuid); err != nil {
        checkErr(err)
    }
}
```



클라이언트 - 다이어리 데이터 모델

```
class Items {  
    String? id;  
    String? uuid;  
    String? note;  
    String? dt;  
    int? timestamp;  
  
    Items({this.id, this.uuid, this.note, this.dt, this.timestamp});  
  
    factory Items.fromJson(Map<String, dynamic> json) {  
        var id = json['id'];  
        var uuid = json['uuid'];  
        var note = json['note'];  
        var dt = json['dt'];  
        var timestamp = json['timestamp'];  
        return Items(id: id, uuid: uuid, note: note, dt: dt, timestamp: timestamp);  
    }  
}
```



클라이언트 - 다이어리 데이터 모델

```
class ResponseModel {  
    int? code;  
    String? message;  
    List<Items>? items;  
  
    ResponseModel({this.code, this.message, this.items});  
  
    factory ResponseModel.fromJson(Map<String, dynamic> json) {  
        return ResponseModel(  
            code: json['code'],  
            message: json['message'],  
            items: List.from(json['items']).map((e) => Items.fromJson(e)).toList(),  
        );  
    }  
}
```



Android Emulator의 경우

127.0.0.1를

10.0.2.2로

클라이언트 - 다이어리 데이터 가져오기

```
fetchDiary() async {  
    final uri = Uri.http(  
        '10.0.2.2:3000',  
        '/diary',  
        {  
            'search': _selectedDateTime.toString(),  
        },  
    );  
    final resp = await http.get(uri);  
    final items = ResponseModel.fromJson(  
        jsonDecode(  
            utf8.decode(resp.bodyBytes),  
        ),  
    );  
    setState(() {  
        diaryItems.clear();  
        diaryItems.addAll(items.items ?? []);  
    });  
}
```



클라이언트 - 다이어리 데이터 쓰기

```
final resp = await http.post(
  Uri.parse(url + "/diary"),
  body: {
    "uuid": const Uuid().v4(),
    "note": textEditingController.text,
    "dt": _selectedDateTime.toString(),
    "timestamp": DateTime.now().millisecondsSinceEpoch.toString(),
  },
);

// 컨트롤러 초기화
textEditingController.clear();
```

클라이언트 - 다이어리 데이터 삭제하기

```
Future deleteDiary(String uuid) async {  
    final uri = Uri.http(  
        '10.0.2.2:3000',  
        '/diary',  
        {  
            'uuid': uuid,  
        },  
    );  
    final resp = await http.delete(uri);  
}
```



데이터베이스 쿼리 정리

테이블 생성하기	myDB.Exec	<pre>CREATE TABLE IF NOT EXISTS diary (id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, uuid TEXT, note TEXT, dt TEXT, timestamp INTEGER);</pre>
정보 추가하기	myDB.Exec	<pre>INSERT INTO diary VALUES(NULL,?,?,?,?)</pre>
정보 조회하기	myDB.Query	<pre>SELECT * FROM diary WHERE Date(dt) = Date(?)</pre>
정보 삭제하기	myDB.Exec	<pre>DELETE FROM diary WHERE uuid=?;</pre>

Summary

Go 지금이라도 해볼만합니다.
시작해보세요!

<https://github.com/JAICHANGPARK/go-to-busan-hands-on>



Q & A



Thank you