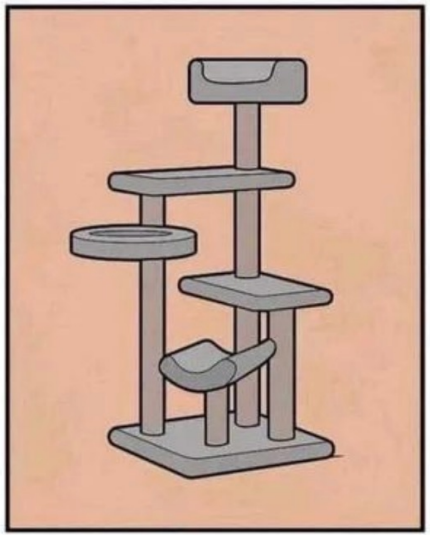
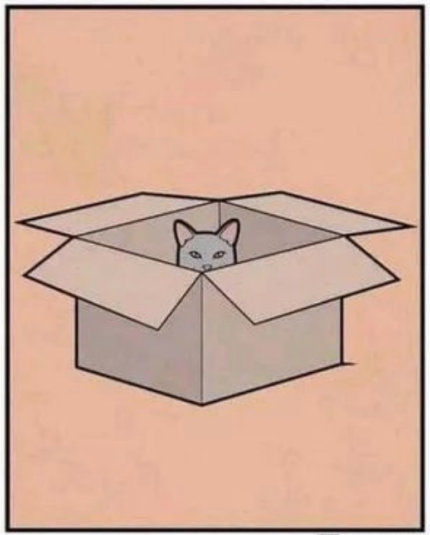
Infrastructure from Code

김민준, Cloud Club

AUSG

만약 당신이

Product features	User needs
	

Migrating your infrastructure to terraform

```
Error: Invalid for_each argument

  on /home/vscode/.terraform.cache/modules/launchp
   4:    for_each = var.keyvaults

The given "for_each" argument value is unsuitable:
must be a map, or set of strings, and you have pro
string.


Error: Invalid for_each argument

  on /home/vscode/.terraform.cache/modules/launchp
   4:    for_each = var.resource_groups

The given "for_each" argument value is unsuitable:
must be a map, or set of strings, and you have pro
string.
```


terraform plan
terraform apply -auto-approve
150 resources changed
150 resources changed???
imgflip.com


Welcome to AWS IAM !

시사상식사전

있어빌리티

요약 '있어 보인다'는 표현과 능력이라는 뜻을 가진 영단어 'ability'를 합쳐 만든 신조어로 실상은 별 거 없지만 뭔가 있어 보이게 자신을 잘 포장하는 하는 능력을 뜻한다.

외국어 표기	있어+bility

'있어 보인다'와 능력을 뜻하는 영단어 'ability'를 합쳐 만든 신조어이다. 실상은 별 거 없지만 사진이나 영상을 통해 뭔가 있어 보이게 자신을 잘 포장하는 능력을 뜻한다. 익명성이 보장되는 온라인 게시판이나 자신의 SNS에 고가의 상품 로고를 부각시킨 사진이나 해외 휴양지에서의 휴가 사진, 고급 호텔에서 식사를 하는 사진 등을 게시하는 사람들의 행위를 '있어빌리티'라고 표현할 수 있다. 이러한 현상과 비슷한 경제학 용어로는 '베블런 효과(Veblen effect)'가 있다. 과시적 소비를 뜻하는 베블런 효과는 소비재의 가격이 상승하는데도 수요가 증가하는 현상을 의미한다.

잘 오셨어요!

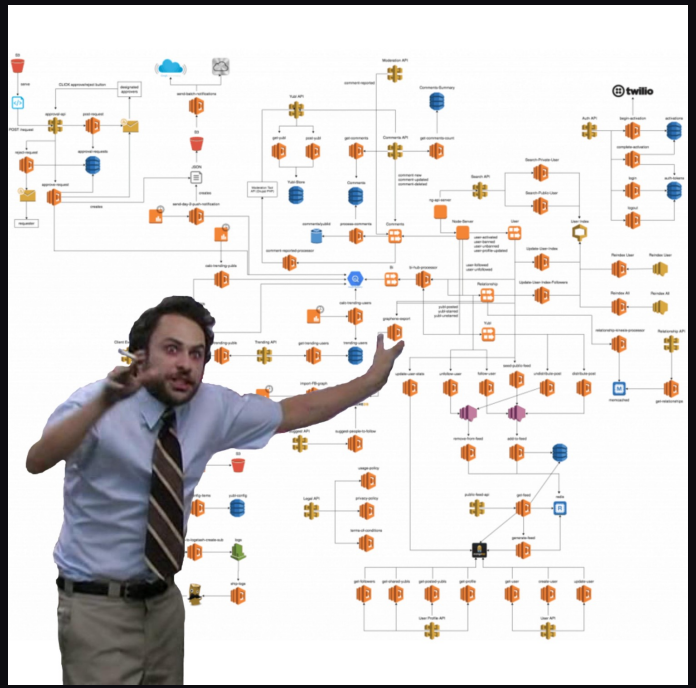
김민준

- 2년차 Software Engineer
  - Next.js / Nest.js
- 전북대학교 컴퓨터공학
- 관심사
  - 해커톤
  - 동아리 활동
  - 지식 공유

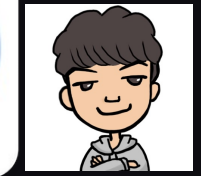
어쩌다 이 주제를?

해커톤

→ 인프라 공부해야겠다..



CLOUD CLUB



Cloud Club

→ 서버리스 맛있다!

→ IaC 알면 편하다고?

Infrastructure as Code

→ 어렵네(귀찮네)..

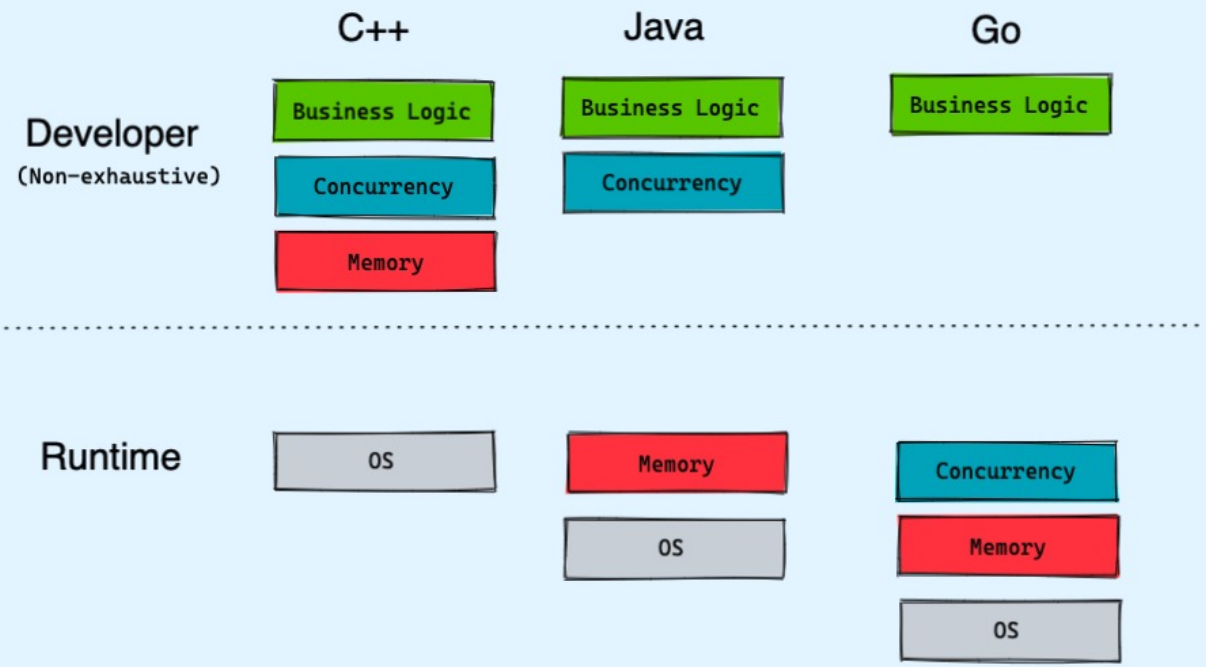
→ 더 쉬운 방법은 없을까?

프로그래밍 언어와 클라우드 인프라는
단일 패러다임으로 수렴될 것이며,
필요한 모든 리소스가 자동으로 프로비저닝되고,
이를 실행하는 환경에 의해 최적화될 것이다.

by Shawn "swyx" Wang

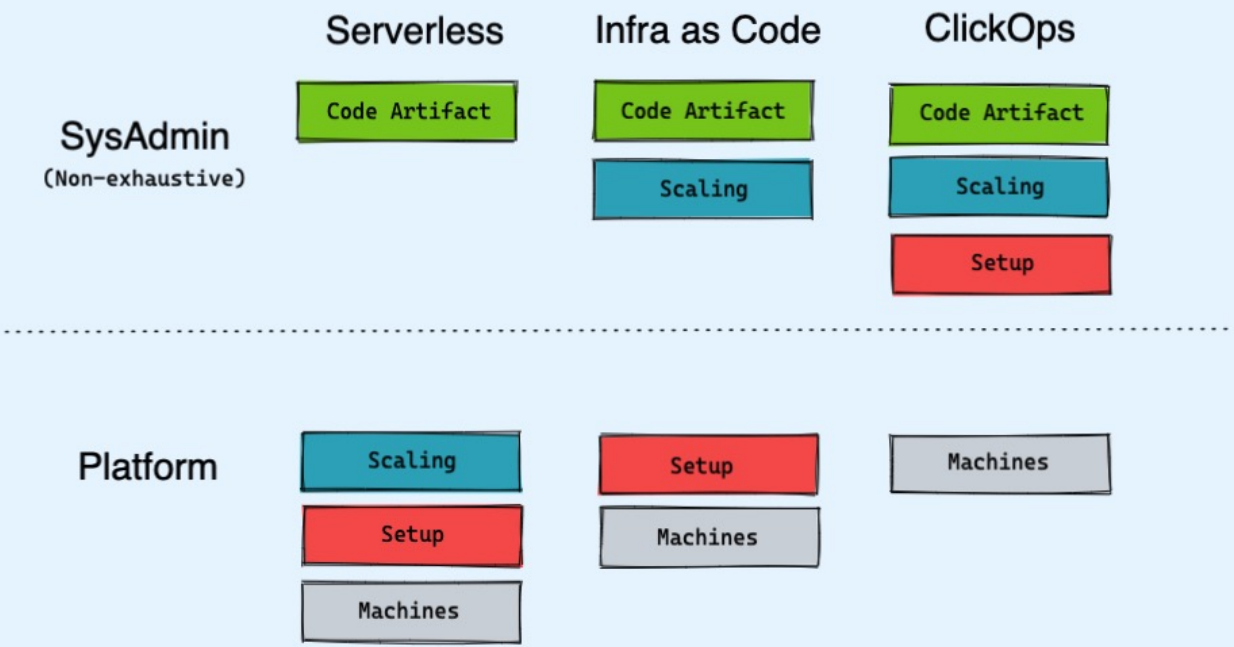
"Just Write Business Logic"

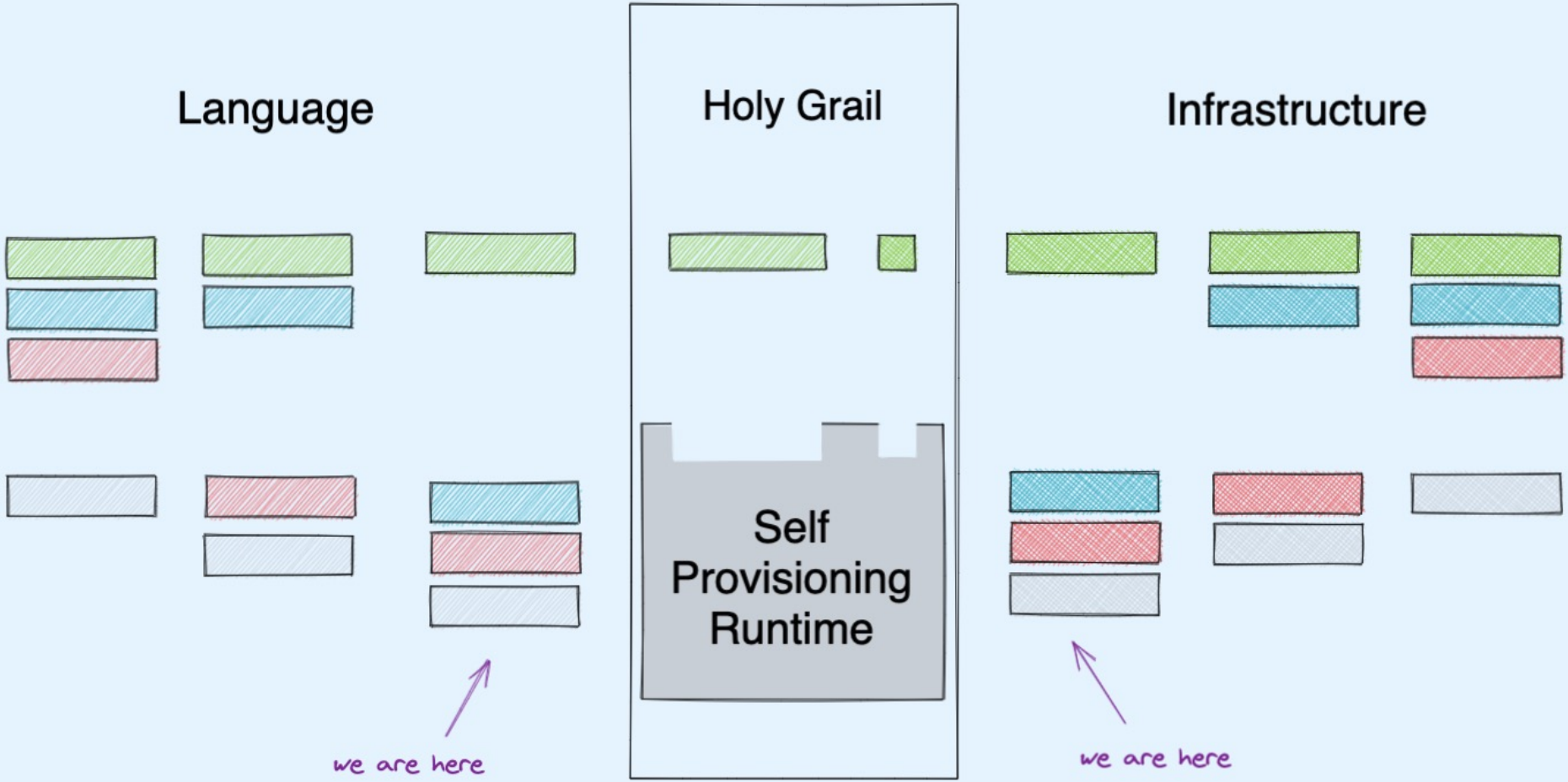
in Programming Languages



"Just Deploy Business Logic"

in Infrastructure




Language
Holy Grail
Infrastructure
Self
Provisioning
Runtime
we are here
we are here

Infrastructure from Code

→ 실제 로직에서 요구 사항을 추론하여 최적의 클라우드 인프라를 자동으로 프로비저닝

스스로 인프라를 구축

오직 애플리케이션 코드만

IaC의 장점 포함

자동화
일관성
재사용성
버전관리
.
.

IfC 특

셀프 업그레이드

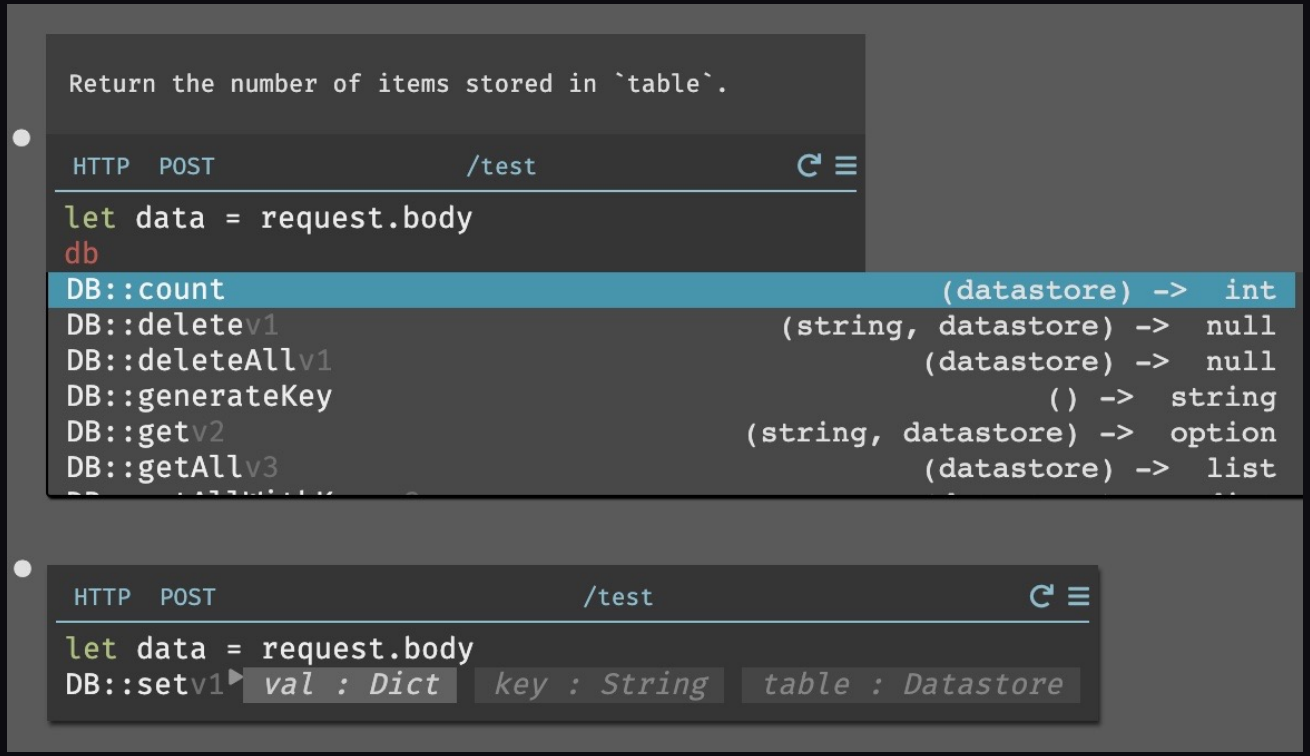
늘 최적화된 인프라

귀찮은 구성 자동 매핑

IAM
VPC
ACL
.
.

네 가지 IfC 접근 방식

언어


darklang
Return the number of items stored in `table`.
HTTP POST /test
let data = request.body
db
DB::count (datastore) -> int
DB::deletev1 (string, datastore) -> null
DB::deleteAllv1 (datastore) -> null
DB::generateKey () -> string
DB::getv2 (string, datastore) -> option
DB::getAllv3 (datastore) -> list
HTTP POST /test
let data = request.body
DB::setv1 val : Dict key : String table : Datastore

SDK


```
import { api } from "@ampt/api";
import { data } from "@ampt/data";

// define a public api and create a new router
const publicApi = api("public").router("/api");
publicApi.get("/hello", async (event) => {
  // ingest data to database
  await data.set('foo', 'bar');
  const results = await data.get('foo');
  return event.status(200).body({ message: 'Hello from the public api! ${results}' });
});
```

주석

klotho

TS webapi.ts

```ts
import * as shelter from './pets/shelter/shelter';

const app: any = express();
app.use(shelter.router)

/*
 * @klotho::expose {
 *  target = "public"
 *  id = "petsApp"
 * }
 */
app.listen(3000, async () => {
  console.log(`App listening locally`);
});
```

SDK + 주석


```rust
use rocket::{get, routes, State};
use sqlx::PgPool;

struct MyState(PgPool);

#[get("/hello")]
fn hello(state: &State<MyState>) -> &'static str {
    // Do things with `state.0`...
    "Hello, Postgres!"
}

#[shuttle_service::main]
async fn rocket(
    #[shared::Postgres] pool: PgPool
) -> shuttle_service::ShuttleRocket {
    let state = MyState(pool);

    Ok(
        rocket::build()
            .manage(state)
            .mount("/", routes![hello])
    )
}
```

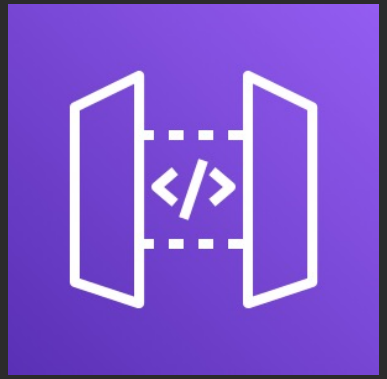
언어

SDK

주석

SDK + 주석

```
/**
 * @klotho::expose {
 *   id = "todo-api"
 *   target = "public"
 * }
 */
app.listen(3000, () => {
  console.log('Server is up and running at port 3000');
});
```



```
/**
 * @klotho::persist {
 *   id = "tasks"
 * }
 */
const tasks = new Map<string, string>();
```

```
/**
 * @klotho::persist {
 *   id = "tasks"
 * }
 */
const tasks = new Map<string, string>();

/**
 * @klotho::expose {
 *   id = "todo-api"
 *   target = "public"
 * }
 */
app.listen(3000, () => {
  console.log('Server is up and running at port 3000');
});
```

klotho

Todo API Demo – Klotho

No Silver Bullet – IfC의 한계

개발자들의 신뢰 문제

→ 믿고 비즈니스 문제에 집중

저수준 설계 작업은 여전히 중요

→ 솔루션 아키텍트가 여전히 필요

유지 보수의 어려움

→ 추가적인 도구나 방법 필요

IfC is Coming

요약

개발자는 프로그램의 로직에 집중해야 한다.

오직 로직 코드만으로 인프라를 만드는 IfC가 있다.

다양한 접근 방식들이 시도되고 있고, 계속 발전할 것이다.

참고문헌

github.com/Me1e

Infrastructure from Code (IfC) by Ampt

The future of cloud development by Jeremy Daly

The Self Provisioning Runtime by Shawn "swyx" Wang

State of Infrastructure-from-Code 2023 by Ala Shiban

What Does The Future Hold For Serverless? by Allen Helton

The Current State of Infrastructure From Code by Allen Helton

Infrastructure From Code - My First Impression by Allen Helton

Overcoming Infrastructure as Code obstacles: It's time for Infrastructure from Code by Asher Sterkin

Overcoming Infrastructure as Code obstacles: It's time for Infrastructure from Code by IDG Connect

AWS re:Invent 2022 - Unleash developer productivity with infrastructure from code (COM301) by Jeremy Daly

AWS Community Day Nordics 2023 - Infrastructure FROM Code: Logical evolution of cloud by Emarah Samdan

AUSG