이준영 Junyoung Lee

010-7210-2992

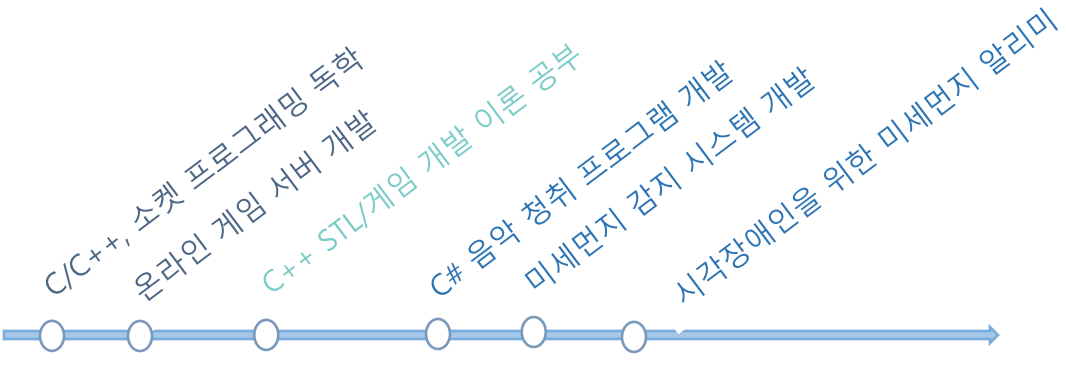
8byte@naver.com

https://github.com/64byte

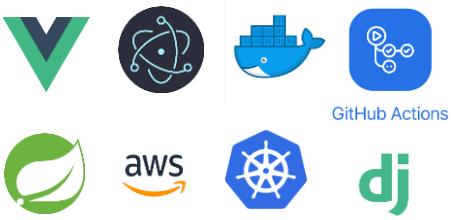
https://www.acmicpc.net/user/8byte

간략한 소개

대용량 트래픽 처리 및 다양한 서비스를 경험하고 싶은 신입 개발자입니다.
중학교 시절 온라인 게임 서버를 시작으로 프로그래밍에 흥미를 갖게 되었고,
웹을 통해 다양한 서비스를 실현할 수 있다는 것을 느끼고 난 지금은 웹서버
개발자를 꿈꾸고 있으며, 최근 쿠버네티스에 관심이 많습니다.
프로그램은 혼자 만들 수 없다는 걸 느끼며 개발하고 있습니다.
경력은 없지만, 경험은 있는 개발자라고 생각합니다.



현재 관심



보유기술

(관심 분야)

Programming Languages (프로그래밍 언어)

- Java 11, C, C++11, C# 7.0, Python3.8, Javascript, Vuejs, HTML5, CSS3

Database (데이터베이스)

- PostgreSQL, Maria DB(MySQL), Oracle

Environment (환경)

- aws Lightsail, aws EC2, Windows, Linux(Ubuntu, Rasbian), docker, k8s

Current Interest Fields (현재 관심 분야)

- Spring (boot, MQP, etc), Kubernates, aws, docker, Django, vue.js, Electron, Github action(CI)

Previous Interest Fields (이전 관심 분야)

- Go, Rust, Mongo DB, WPF, PHP, Xamarin, Android

프로젝트

Qualcomm Institute

2018. 01 ~ 2018. 02

IoT 플랫폼 개발 프로젝트(미세먼지 감시 시스템)

- 개요: 사용자가 미세먼지를 측정하고 스마트폰과 연동하여 측정 데이터를 웹서버에 저장하고 데이터는 안드로이드 애플리케이션 화면과 웹 화면을 통해 조회
- 팀 구성: 웹 개발 (2), 안드로이드 개발 (2), 엠베디드 개발 (1)
- 역할: 웹 개발
- 사용 스택: PHP, Slim framework, JWT, Javascript, D3.js, Maria DB
- 업무
  - 요구 사항 정의 (https://trello.com/b/H4h41yy8/%EC%9A%94%EA%B5%AC-%EC%82%AC%ED%95%AD)
  - 기능 정의 (https://trello.com/b/4OEbonfx/function)
  - 설계
  - 구현
    I. JWT 를 이용한 회원가입 이메일 OTP 인증 구현
    II. CSRF 토큰을 이용한 요청 위조 방지 구현
    III. D3.js 를 이용한 실시간 미세먼지 변화 그래프 조회 구현
    IV. Restful API 구현
    V. JWT 를 이용한 API 호출 사용자 검증
- 저장소 주소: https://github.com/64byte/teamb-iot

시각장애인을 위한 미세먼지 알리미 – *2018.04 ~ 2018.05*

- 개요: 시각장애인도 실내 미세먼지를 알 수 있도록 Clova TTS API 를 통해 정보 제공
- 팀 구성: 엠베디드 개발 1, 안드로이드 개발 1, 디자인 1
- 역할: 엠베디드 개발, 안드로이드 개발
- 사용 스택: Python, Java, Android
- 업무

- 요구 사항 정의
- 기능 정의
- 구현
  I. 라즈베리파이와 연결된 미세먼지 센서 및 온/습도 센서의 동작 구현
  II. 라즈베리파이와 안드로이드 블루투스 연결 구현
  III. Clova TTS Rest API 호출을 이용한 시각장애인용 안드로이드 어플리케이션 구현
  IV. 일반 사용자용 어플리케이션 구현

포트폴리오

쇼핑몰 REST API 서버 – *2021. 01 ~*

- 개요: JWT 인증 기반 쇼핑몰 Spring REST API 서버
- 사용 스택: Spring boot(Web, security, validation, jpa), postgresql, docker, JWT
- 저장소 주소: https://github.com/64byte/Web-Project/tree/main/backend

경험

미리디 – *2020. 06. 15 ~ 2020. 09. 07*

80 만 사용자가 사용하는 무료 디자인 툴 미리캔버스

- 문제해결/개선 사례 (1)의 기존 검색 API 속도 개선
- 검색한 아이템에 대해 선택되는 정도에 따라 인기순으로 검색하는 기능 구현
- 백오피스 관련 개선 및 새로운 기능 구현
- 기술 스택
  - Spring Boot(Java 11), PostgreSQL, Bitbucket, Bamboo, aws(EC2, RDS, Elastic Beanstalk, Cloud Watch, Cloud Search, Lambda, ...), NewRelic(APM)

퀄컴 연구소에서 프로젝트 진행 – *2018. 01 ~ 2018. 02*

- 프로젝트(1)에서 언급한 프로젝트 진행

문제해결/개선 사례

1. **API 서버에서 검색 엔진 서버에서 가져올 때의 속도 개선 – *2020. 08***

- 문제: API 요청 시에 검색 엔진 서버에서 검색 정보를 가져올 때 속도상의 문제가 있었음.
- 원인: API 서버에서 검색 엔진 서버에서 검색하는 아이템을 가져올 때도 여러 번 검색 엔진 서버로 검색을 함.
- 대안
  - 검색하는 아이템을 종류별로 분류한 뒤에, 검색 엔진 서버에 검색할 때, 한 번으로도 필요한 아이템을 가져오도록 개선하는 방법

    - 캐시 할 수 있는 요소를 캐시 전략에 따라 Redis 를 이용해 캐싱하여 검색 엔진 서버로 연결하는 횟수를 줄이는 방법
- 해결
    - 첫 번째 대안으로, 검색하는 아이템을 종류별로 분류하여 검색 엔진 서버에 필요한 아이템을 한 번으로 가져오는 방법으로 개선함.
- 평가: APM(NewRelic)에서 해당 API 의 응답 시간이 22% 정도를 차지하였으나, 2~3%로 줄여 성능을 개선함.

2. 여러 태스크가 하나의 컨테이너에서 데이터를 꺼낼 때 중복 사용하는 문제 – *2019. 05*

- 문제: C#의 Task 자료구조를 이용하여 다수의 태스크를 생성하여 하나의 컬렉션에서 데이터를 중복으로 사용하는 문제
- 원인: Task 자료구조가 단일 스레드에서 동작하는 줄 알았으나, Task 자료구조 명세서
  에 따르면 유기적으로 다중 스레드에서 동작하는 방식
- 대안: 스레드로부터 안전한 컬렉션 구현
- 해결
  1) lock 을 이용한 스레드 동기화
  2) Interlocked 으로 구현된 ConcurrentCollection 사용
- 평가: 처음 lock 을 이용한 스레드 동기화로 문제를 해결했지만, 태스크가 많아질수록 잦은 busy-waiting 으로 인한 성능 이슈로 인하여 Interlocked 으로 구현된 ConcurrentCollection 을 사용하여 성능을 향상함

학력

동의대학교 컴퓨터공학과 학사 졸업 – *2015.03 ~ 2019.02*

- 학점 4.44/4.50
- 단과대학 수석 졸업

건국고등학교 졸업 – *2008.02 ~ 2011.03*

언어

영어 (Intermediate)
한국어 (Native)

기타 인적사항

~~**OPIC (Intermediate High) – *2019.01.21 취득(만료, 갱신 예정)***~~
정보처리기사 – *2018.09.17 취득*
병장 만기 제대 – *2011.11.28 ~ 2013.08.27*

아래부터는 간략한 제 소개 글 및 생각이므로, 꼭 읽지 않으셔도 됩니다만,
저를 이해하시는 데 도움이 되리라 생각합니다.

1. 간략한 성장 과정

[재미를 나누기 위해 시작한 개발자의 꿈]
여느 학생과 마찬가지로 처음에는 단지 게임이 좋아서 나도 사용자들에게 재미라는 가치를 주는 게임을 만들고 싶어서 프로그래밍을 배웠습니다.
주먹구구식으로 프로그래밍을 배워 만든 첫 게임 서버는 제대로 돌아갈 리가 만무했고, 구현했던 기능 또한 제대로 돌아가지 않았습니다.
그렇게 첫 번째 소프트웨어는 실패작이 되었습니다. 다시 제대로 만들어 보고자, TCP/IP 소켓 프로그래밍과 게임 개발 관련 서적들을 모아 공부하면서 게임 서버를 만들어 갔습니다. 그렇게 하다 보니 어느 정도 괜찮은 게임 서버를 만들 수 있게 되었습니다.
제 인생에 있어서 프로그래밍은 그렇게 시작되었습니다. 그렇게 학창 시절에 프로그래밍에 빠져서 시간을 보내게 되었고 이를 즐기게 되었습니다.
비록 프로그래밍 공부 때문에 학교 공부를 등한시하게 되어, 대입에는 실패하게 되었지만 내가 하고 싶은 일은 일찍이 찾을 수 있게 되었습니다.
또한, 그때 당시를 되돌아보면서 얻은 것은 제대로 된 커리큘럼 없이 공부하는 것은 깊이 있는 지식을 얻는 게 힘들다는 것을 알게 되었습니다.
군대를 전역하고 나서 다시 대학교에 가서 프로그래밍을 제대로 배워야겠다는 생각이 들었고 컴퓨터공학과를 입학하게 되었습니다.

[나 자신을 알다]
대학에 입학한 것은 어떻게 보면 저 자신을 알게 된 계기였습니다.
입학하기 전에는 프로그래밍을 혼자서 주먹구구식으로 익혔기 때문에 코드를 작성했을 때, 컴퓨터상에서 그 코드가 어떻게 동작하는지만 알 뿐, 그 코드가 컴퓨터 내부에선 어떤 식으로 메모리에 로드되고 최적화되는지 몰랐습니다. 즉, 컴퓨터 프로그램을 만들지만, 컴퓨터 내부 구조에 대해선 무지했었습니다.
대학에서 디지털 논리회로, 컴퓨터구조, 그리고 컴퓨터시스템설계 강의를 들으면서 컴퓨터 지식에 대해 세밀하게 배울 수 있었고 조금이나마 코드를 어떻게 짜야 효율적으로 동작하는지 알 수 있었습니다.
또한, 윈도우즈 프로그래밍만 알던 저에게 모바일, 웹, 임베디드 등 다양한 플랫폼에 대한 강의는 또 다른 새로운 세계였습니다.
그 이전까지는 윈도우즈 프로그래밍만 알았고 윈도우즈 프로그래밍만 했었습니다.
웹 프로그래밍, 모바일네트워크, 그리고 임베디드시스템 및 실습 강의는 제게 IT 환경이 변해가고 있음을 알려주었습니다.
하나의 플랫폼을 고집하는 것은 더는 무의미하며 사용자에게 서비스를 제공하는 게 한계가 있음을 일깨워 주었습니다.
스마트폰, 태블릿이 보급되면서 하나의 플랫폼이 아닌 다양한 플랫폼 환경에서 통합적으로 실행될 수 있는 소프트웨어가 중요하며 개발자와 사용자 모두에게 편의를 줄 수 있다는 것이었습니다. 대학에서 이러한 강의를 들으면서 여러 플랫폼에서 소프트웨어를 만드는 능력을 배울 수 있었습니다.

이렇듯 대학에 입학하여 배운 지식은 이전의 제가 얼마나 우물 안의 개구리였는지를 알 수 있게 해주었습니다. 제가 대학에 오지 않았다면 컴퓨터구조나 IT 환경변화에 대해 자세하게 알 수 없었을 것입니다. 대학에서의 지식은 저에게 공학도의 능력을 한층 향상해 주었습니다

2. 개발자로서 개발 철학

[개발은 혼자 하는 것이 아니다.]

소규모의 프로그램이라면 개발은 혼자 해낼 수 있다고 생각합니다. 그러나 요즘 시대의 소프트웨어는 작은 규모의 프로그램이 아닙니다.

그러므로 개발은 혼자 할 수 없다고 생각합니다. 그러므로 협업의 중요성은 여러 번 강조해도 지나치지 않다고 생각합니다.

혼자만 알아볼 수 있는 코드를 작성하는 것보다, 모두가 쉽게 이해할 수 있는 코드를 작성하는 것이 중요하고, 혼자만 알아볼 수 있는 단어를 말하는 것보다, 모두가 쉽게 이해할 수 있는 단어로 대화하는 것이 중요하다고 생각합니다.

즉, 팀 내에서의 협업, 개발팀이 아닌 다른 팀과의 협업을 위해 이러한 부분을 인지하고 생각하면서 공유하며 일을 하는 게 필요하다고 생각합니다.

협업을 자주 해보진 않았지만, 이러한 생각 때문에 협업할 때는 코드를 읽기 쉽게 작성하도록 하고, 설명이 필요한 부분에는 코드에 대한 주석을 남기고 문서화를 하며 팀 내에서 관련 내용을 공유하려고 노력합니다.

[개발의 산출물은 추상적이지 않다.]

개발자는 코드를 작성하면서 개발합니다. 그 말은 우리의 작업은 기록물이 남는다는 말입니다.

그리고 그 코드가 어떻게 작동되는지는 컴퓨터가 판단하며, 우리는 그 성능을 추적할 수 있습니다.

그래서 컴퓨터의 내부적인 구조를 이해하며 사이드 이펙트가 없는지 코드를 작성하면서도 타인이 봐도 복잡하지 않은 코드를 작성해야 한다고 생각합니다.

즉, 우리는 스스로가 작성한 코드에 대해 책임감을 느껴야 한다고 생각합니다.

이를 위해선 내가 작성한 코드에 대해 항상 의심하고, 다른 동료의 코드도 읽어보면서 상호 피드백을 하는 것이 중요하다고 생각합니다.

그러다 보니 코드를 작성할 때에 많이 고민해왔고 하고 있습니다.

예를 들면, Exception 처리를 어떻게 해야 할 지, null 처리는 어떻게 해야 할지, 객체 지향적인 코드 작성 방법이 어떤 것인지 등에 관한 생각을 끊임없이 하고 github 에서 타인의 코드를 읽어봅니다.

3. 개발 분야를 웹 분야로 생각하게 된 계기와 한(하고) 있는 것들

[모든 것을 연결하는 웹]

기술이 발전할수록 PC 나 스마트폰뿐만 아니라 새로운 장치들이 생겨나고 있습니다. 먼 미래에는 더 많은 제품이 스마트하게 연결될 것으로 생각합니다.

이를 연결해주는 기술이 바로 웹이라고 생각합니다. 웹은 인터넷이 연결되어 있고 URI 만 있으면 언제 어디서든지 접속할 수 있습니다.

새로운 시대에는 웹 분야가 중추적인 역할을 할 것이라 믿어 의심치 않습니다.
그 뿐만 아니라 웹을 이용해 다양한 서비스를 실현할 수 있다고 생각합니다. 이러한 이유로 자연스레 웹 분야에 많은 관심을 두게 되었습니다.

웹 분야에 관해 관심을 가지다 보니 자연스레 상용 웹서버는 어떻게 이루어져 있는지 분석을 하게 되었습니다. 실제 서비스하고 있는 다양한 웹 사이트를 분석하면서 많은 것을 배울 수 있었습니다.
이전에는 단순히 웹서버가 하나의 컴퓨터에서만 동작하는 줄 알았으나 많은 트래픽을 감당하기 위해 서버를 분할하여 로더밸런싱을 통한 효율적인 트래픽 관리, 콘텐츠는 CDN 을 따로 두어 전송하는 방식, 그럴 뿐만 아니라 접근하는 데이터베이스 서버를 분할하여 관리하는 방식 등, 웹 서버의 효율적인 관리와 최적화를 위해 많은 기술을 을 배울 수 있었습니다.

단순히 위와 같은 최적화뿐만 아니라, 보안을 위한 기술도 많이 접할 수 있었습니다.
XSS 스크립팅 방지, SQL 인젝션 방지, CSRF 방지, 암호화를 통한 보안 로그인, API 사용자를 위한 토큰 관리 등 이러한 웹 서버를 지탱하는 기술을 공부하면서 많은 재미를 느낄 수 있었고 많은 것을 배울 수 있었습니다.
이렇게 배운 기술을 제작하는 프로젝트에 적용하여 좀 더 견고한 웹 서버를 만들 수 있었고 완성도 높은 프로젝트를 할 수 있었습니다.

지금도 여러가지 서버를 지탱하는 기술을 배우면서 웹 프레임워크를 공부하면서 프로젝트를 하고 있습니다. 프레임워크의 내부 구조를 계속해서 공부하면서 이러한 프레임워크와 함께 효율적인 서버의 아키텍쳐 구성은 어떻게 하는 지 등과 최근에는 컨테이너 관리 기술인 쿠버네티스에 관심이 많아 재미있게 공부하고 있습니다.

여기까지 제 이력서를 봐주셔서 감사합니다.
더 궁금하신 내용이 있으시면
8byte@naver.com / 010-7210-2992 으로 연락주시면 감사하겠습니다.