프론트엔드 개발자, 김재환입니다.

💁 About Me

Introduction

안녕하세요! 2년 차 클라이언트 개발자 김재환입니다. 그동안 **Github**, 회의를 ***Spirnt로 변경하기*** 통해 협업 능력을 길렀습니다. **개발 경험**으로 Slack app(C#), WiFi print app(MAUI), .NET Core REST API, Azure Devops CI/CD 가 있습니다. **Frontend 개발자**이며, Backend 개발 경험이 있습니다.TDD와 Clean Architecture,MVVM 패턴을 공부 중이며 다양한 프로젝트의 구성요소와 동작 방식을 이해하여 풀스택 개발자가 되고싶습니다.

시퀀스 다이어그램과 TDD, Clean Code, Clean architecture로 개발을 진행했습니다. 시퀀스 다이어그램을 통해 아키텍처를 설계하고 비즈니스 로직을 분리하였으며, TDD를 통해 버그를 진단 할 수 있었습니다.

Contact & Channel

- **Github** | https://github.com/SSstupid
- **Email** |rlawoghks3@gmail.com

⛏️ Skills

Backend

- C# : .NET Core 5.0 ~ 8.0

DevOps

- Azure DveOps - Pipeline
- YAML

Frontend

- XAML

Tools & Collaboration

- Visaul Studio
- Visaul Studio Code
- Github

💻 Work Experience & Projects

Mangoslab

2022.06 ~ 현재 (1년 6개월)

메모지 프린터 기업

Android Wifi printer app

- 인터넷 프린터를 원격으로 명령할 수 있는 앱입니다. 블루투스로 인터넷 연결 셋팅하고 나면 원격으로 명령을 내릴 수 있습니다. 프로젝트의 파트는 임베디드, 메시지버스(MQTT), App으로 되어있으며 저는 App을 담당하여 UI 구성했습니다. 개발에 필요한 지식을 문서, 유튜브를 통해 학습 및 질문하여 채웠습니다.

Finecube Xamarin app 마이그레이션

- MAUI로 마이그레이션하는 작업입니다. 마이그레이션을 진행하기위해 문서, Git issue 참조 Stack overflow를 활용해 진행하고 있습니다. 특정 기능을 오픈 소스를 참조하여 구현한 경험이 있습니다.

  Finecube는 타투 프린터처럼 드래그 형식으로 종이, 박스 등 여러곳에 프린트 할 수 있는 제품입니다. 이 제품을 제어하는 앱으로, 서버, UI, 블루투스를 활용합니다. App을 마이그레이션 하게 된 이유는 Xamarin 지원이 2024.06 이후로 종료되며 앱을 게시하거나 업데이트에 문제가 발생합니다.

  그래서 Xamarin을 인수한 MS가 제공하는 MAUI로 마이그레이션을 진행했습

  니다.

  진행하면서 어려웠던 점은 변경된 UI, 아키텍처, 버그, MAUI를 지원하지 않는 Xamarin framework 등이 있습니다. 기억에 남는 것은 사용할 수 없는 framework

  를 해결하는 작업입니다. 대체할 framework가 없어 직접 구현해야 했습니다.

  그래서 OpenSource를 활용해 구현했습니다. 구현한 것은 Mvvm Cross

  기능으로 Mvx.IoCProvider.IoCConstruct, Task<TResult> NavigateAsync 등이

  있습니다.

  IoCConstruct는 컨테이너를 통해 컨테이너가 관리하지 않는 객체를

  생성하는 기능입니다. 서비스가 Scope, Singleton으로 등록되면 생성자로

  사용하기에는 에러 사항이 있기 때문에 사용했습니다. 장점으로 **IoCConstruct**

  객체의 Constructor에 주입된 서비스는 컨테이너가 관리 할 수 있습니다.

이미지 API

- 메모지 프린터인 Nemonic에 전달 할 **이미지 생성 및 명령어**로 변환하는 기능입니다.

  해당 라이브러리는 Server에 추가되어 윈도우, 모바일 등 다양한 기기와 앱에서 사용 할 수 있습니다. 이미지 생성 라이브러리를 사용해 이미지 사이즈를 **용지에 맞게** 크기를 설정하고, **분절, 회전** 등의 기능을 제공합니다. Given-When-Then Pattern을 활용한

  Unit Test 코드를 작성했습니다.

  API 테스트를 **Unit Test 코드** 변경하여 시간, 용지 절약하기

  이미지가 알맞은 각도로 회전 되었냐의 문제는 사람이 판단 가능한 문제라 생각하였고, API 변경 사항이 있을 때마다 디버그하여 Docker, Swager를 활용해 실제로 인쇄하는

방식으로 진행했습니다. 이 과정이 시간이 오래 걸리며, 용지가 낭비가 되고 변경사항이 있을 때 기존 API가 작동할까?라는 문제가 있었습니다. 이미지 크기는 실제 크기를 측정하거나 이미지 객체에 포함된 정보를 조회하는 방법으로 측정 할 수 있었고, 회전 유무는 이미지에 점 하나를 찍고 회전하여 점의 위치를 특정하는 방식으로 Test 코드를 작성했습니다.

Slack App

- Slack에서 주고 받는 메시지를 프린터 할 수 있는 앱입니다. 프린터 식별 번호, Server를 통해 원격으로 프린터가 가능합니다. 즉 모바일, 데스크탑 등 기기에서 물리적인 연결 없이 프린터가 가능하며, 프린터를 공유하여 사용 할 수 있습니다. Slack은 Block kit 객체를 제공하여 UI 구성 및 데이터를 전달합니다. 이 Json 형식의 데이터를 파싱하여 C# 객체로 전환하여 사용하였습니다.

```
1  private bool IsReprintMessage(JsonArray blocks)
2  {
3      dynamic dynamicBlocks = JsonConvert.DeserializeObject(blocks.ToString());
4
5      foreach (var block in dynamicBlocks)
6      {
7          if (block.type ≠ "actions")
8              continue;
9
10         if (block.elements is null)
11             continue;
12
13         foreach (var element in block.elements)
14         {
15             if (element.action_id == SlackActionsIdDefine.PrintResultReprint)
16                 return true;
17         }
18     }
19
20     return false;
21 }
```

코드 줄 개수를 61 => 21개로 줄여 **가독성을 65% 향상**시켰습니다. if, foreach문을 중첩으로 사용하여 특정 Item을 가져오는 코드를 **dynamic을 활용**하여 개선 했습니다.

Azure DevOpsCI / CD

- Android, Window, MAUI Pipeline (Azure DevOps)
- Jenkins To Azure Devops 마이그레이션

  여러 시나리오에서 CI/CD 를 구축하고, 재사용 가능하도록 템플릿을 생성했습니다.

  초기 구입 비용이 큰 서버 컴퓨터를 제거하고 Azure로 이주하여 모니터링이 쉽고 다양한 Task를 제공하며 템플릿을 생성하여 공유 및 재사용이 가능합니다. 특히 Jenkins에 비해 Azure는 빌드 실행 시간 및 기간, Task 내역을 확인하는 것이 편했습니다.