종합설계1

AOSP 보고서(3)

201503069
김재현

설문조사 결과 및 디자인 스프린트 최종 정리

디자인 스프린트 프로토타입 까지의 과정을 설문조사를 진행되었을 때 다음과 같은 결과를 확인을 할 수 있다.

개선된 안드로이드에서 마음에 들었던 점이 있다면 어느 점에서 장점을 느끼셨나여?

응답 5개

- 좀더 빠른로딩속도
- 속도가 향상됨
- 코드 가독성 향상
- 좀 더 빠른 속도를 경험할 수 있을 것 같음
- 빠른 속도입니다.

안드로이드에서 신뢰할 수 없는 기능이나 단점이 있다면 어떤 것이며 그 이유는 무엇인가요? *

응답 5개

- 몇몇 안쓰는코드 레거시코드들의 존재나 요즘 하드웨어들의 성능을믿고 통으로짠 코드들이단점으로보임
- 안드로이드 운영체제 자체가 오픈되어있다는 느낌;
- 성능 향상 과제라고 설명 들어 단점을 없을 것 같으나 옳은 방향인지 확인이 필요
- 저사양 기기의 경우 어플리케이션을 몇개만 실행시켜도 버벅 거리는 현상
- 어플 몇개만 실행하면 가용램이 너무 적다. / 램이 적으면 실행속도가 느려지기 때문이다.

개선된 안드로이드에서 추가된 기능이 필요없다 생각하는 부분이 있나요?

응답 5개

- 없다.
- 속도관련 개선은 다다익선이라고 생각함 다필요
- try-catch 문은 확인을 할 필요
- 오류 확인 부분에 있어 관습적인 코드 일 수 도 있을 거라는 것이 생각

안드로이드에서 개선의 여지가 있는 부분이 있다면 어떤 것이며 그 이유는 무엇인가요? *

응답 5개

- 자바가 코드짜기 편하긴하지만 속도가 느린거같으면 c를 써보는것도 좋아보임
- c 프레임워크 단에서 레거시 코드들이 있을 것으로 생각
- 텔레스코핑 패턴 말고도 다른 레거시 패턴들이 있을 것 같다.
- 안드로이드 JNI 를 사용하기 때문에 관습적인 코드가 있을 수 밖에 없다고 생각
- 배터리를 줄일 수 있는 방법이 있을까? / 배터리 문제 또한 중요하기 때문

어느 방면에서 성능이 향상되었다고 생각하시나요?

응답 5개

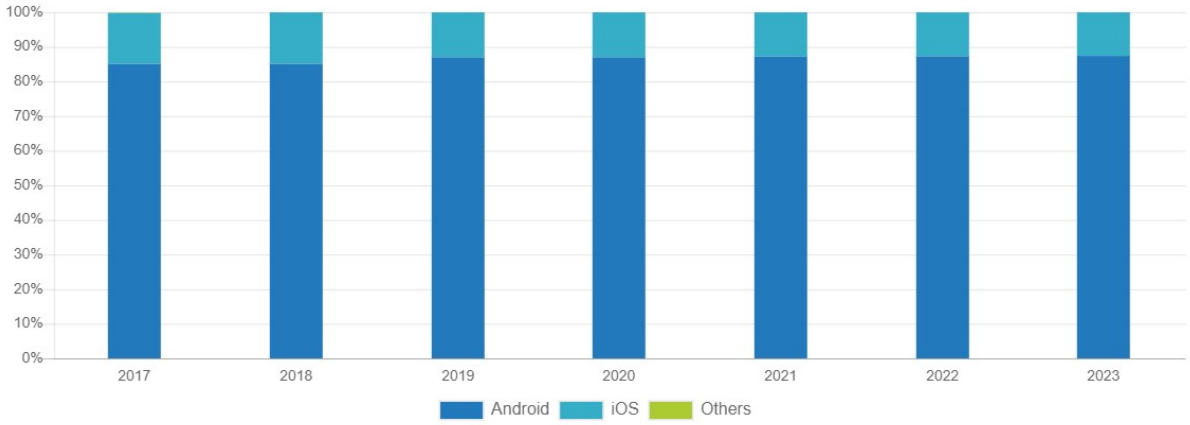
- 압도적으로 느껴지는 정도는 아니지만 로딩속도향상
- 속도 및 안정화라고 생각한다.
- 코드 가독성, 앞으로 코드 개발에 있어서 유의미한 점
- 속도면에서 향상 시켰다 생각
- 실행의 속도

위에 설문 결과에서 살펴볼 수 있듯이 "성능"에 초점을 맞춘 프로젝트이기에 성능에 대한 의견이 많이 발견이 됨을 알 수 있다. 또한 이 의견들 중에서 속도와 코드의 가독성으로 의견 나뉜 것을 알 수 있다. 속도는 성능과 직접적인 연관성이 있음을 직관적으로도 확인을 할 수 있으나 코드의 가독성 앞으로의 유지 보수 성을 고려한 의견이라고 생각을 할 수 있다.

이 프로젝트는 성능을 개선을 하는 프로젝트이지만 개선의 의미를 좀 더 넓게 확장을 하면 코드 가독성 또한 AOSP 기여가 될 수 있는 요소 이기에 반영을 할 수 있다.

특히 유의미한 의견으로 try – catch 의견이 있는데 이를 확인 한 결과 DeadObjectException을 처리할 수 있는 양식임으로 이는 개선이 필요한 부분이 아님을 확인하였다. 이에 설문조사 결과에서 유의미한 결과를 얻었다 할 수 있다.

디자인 스프린트 정리



[그림1]을 보면 IDC의 보고서에서 애플의 시장 점유율이 점점 하락하고 있다. 이는 안드로이드의 시장 점유율이 높아진다는 것을 의미한다. 또한 소비자는 스펙 상향평준화가 되어버린

스마트폰 시장에서, 더 이상, 다른 제품보다 비싼, 고가의 아이폰을 살필요가 없다고 여겨지고 있다. AOSP 기반 안드로이드 OS는 구글의 인증을 받지 않았기 때문에 구글 모바일 서비스를 공식적으로 이용할 수 없지만 안드로이드 스마트폰에서는 정상적으로 작동한다.

AOSP를 통해 OS를 개발한 업체가 구글 서비스를 대체할 자체 서비스와 앱 마켓을 운용할 수 있다면 독자적 생존이 가능한 것이다. AOSP를 가장 적극적으로 이용한 업체가 바로 중국의 샤오미다. 샤오미 또한 빠른 속도로 스마트폰 시장 점유율을 차지하고 있는 것을 [그림2]를 통해 확인이 가능하다. 이를 통해 만약 우리 조가 눈에 띄는 안드로이드 성능 개선 프로젝트를 성공하면, 미래 시장 및 취업에 많이 유리할 것이다.

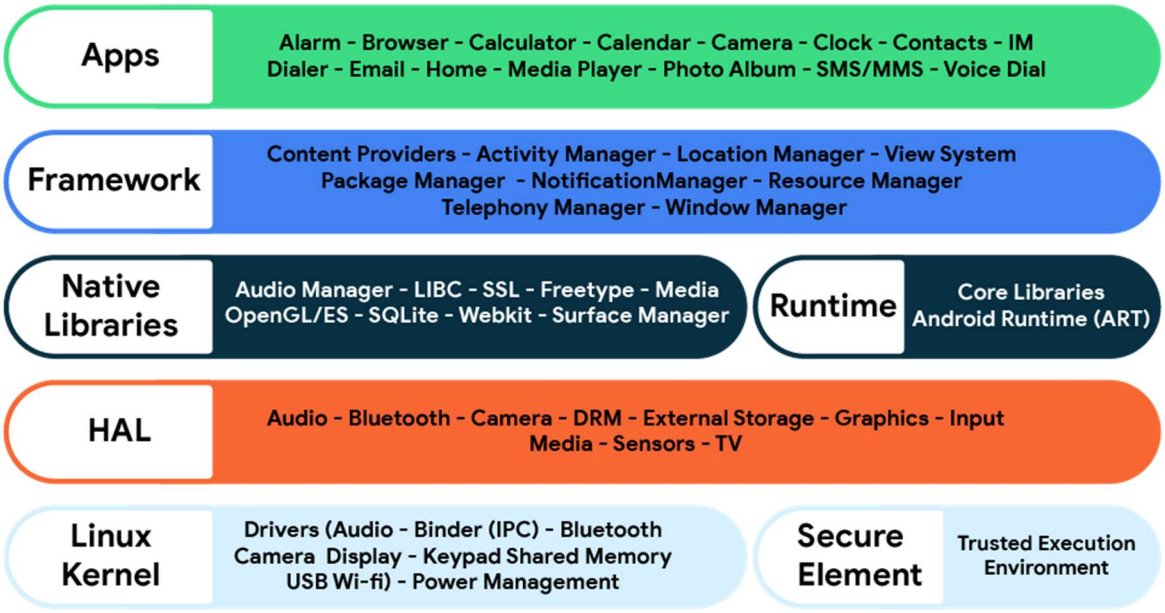


그림 0-1.안드로이드 계층도

우리 조는 AOSP(Android Open Source Project) 를 개선을 하기 위하여 위 그림과 같은 안드로이드 OS 를 분석을 하고 분석을 바탕으로 프레임워크 단에서 개선할 수 있는 사항을 파악을 하여 이를 통해 성능을 향상을 하는 것을 목표로 한다.

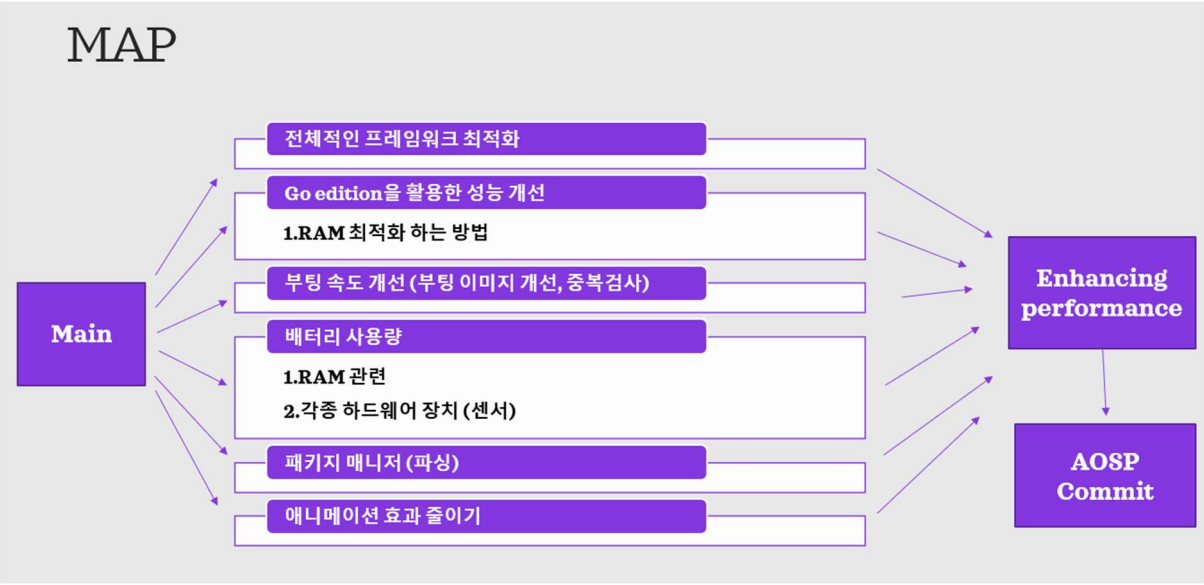


그림 0-2 디자인스프린트 MAP

디자인 스프린트 과정을 통하여 6가지의 방향성을 가지고 "성능 향상"이란 주제를 가지고 디자인 스프린트를 진행하였다.

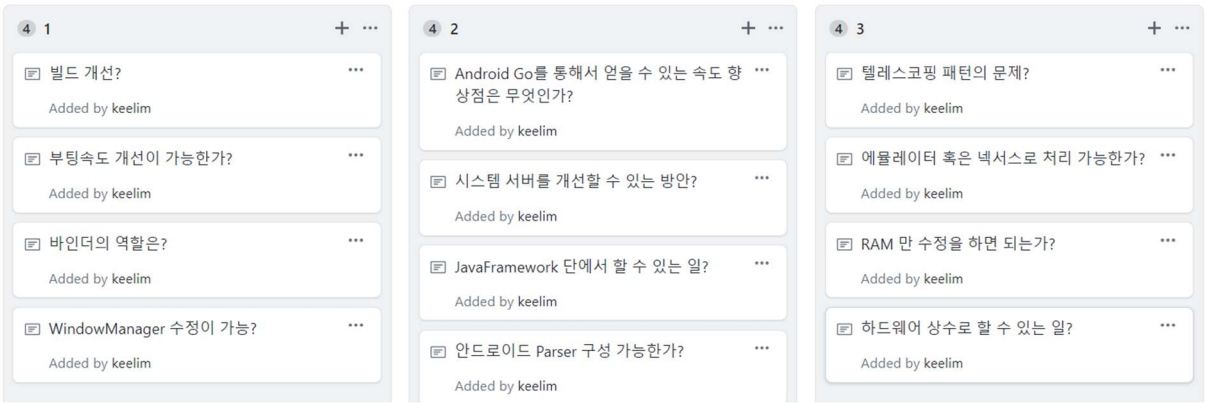
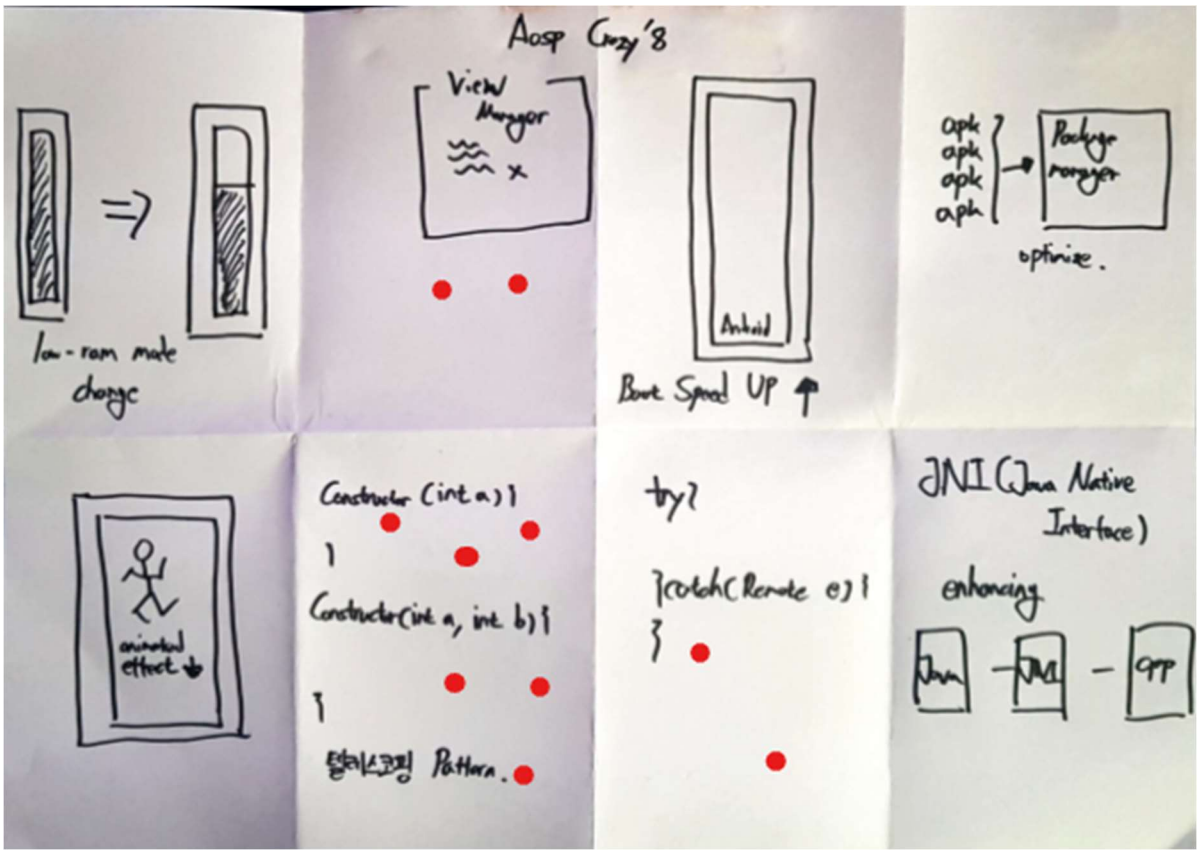


그림 0-3 HMW

하지만 디자인 스프린트 과정을 성능 향상의 측면에서 사용하기에는 어려움이 있었으나 팀원들과의 관점의 대한 논의 후 개선사항 및 방향성을 중심으로 디자인 스프린트를 진행을 하였다.



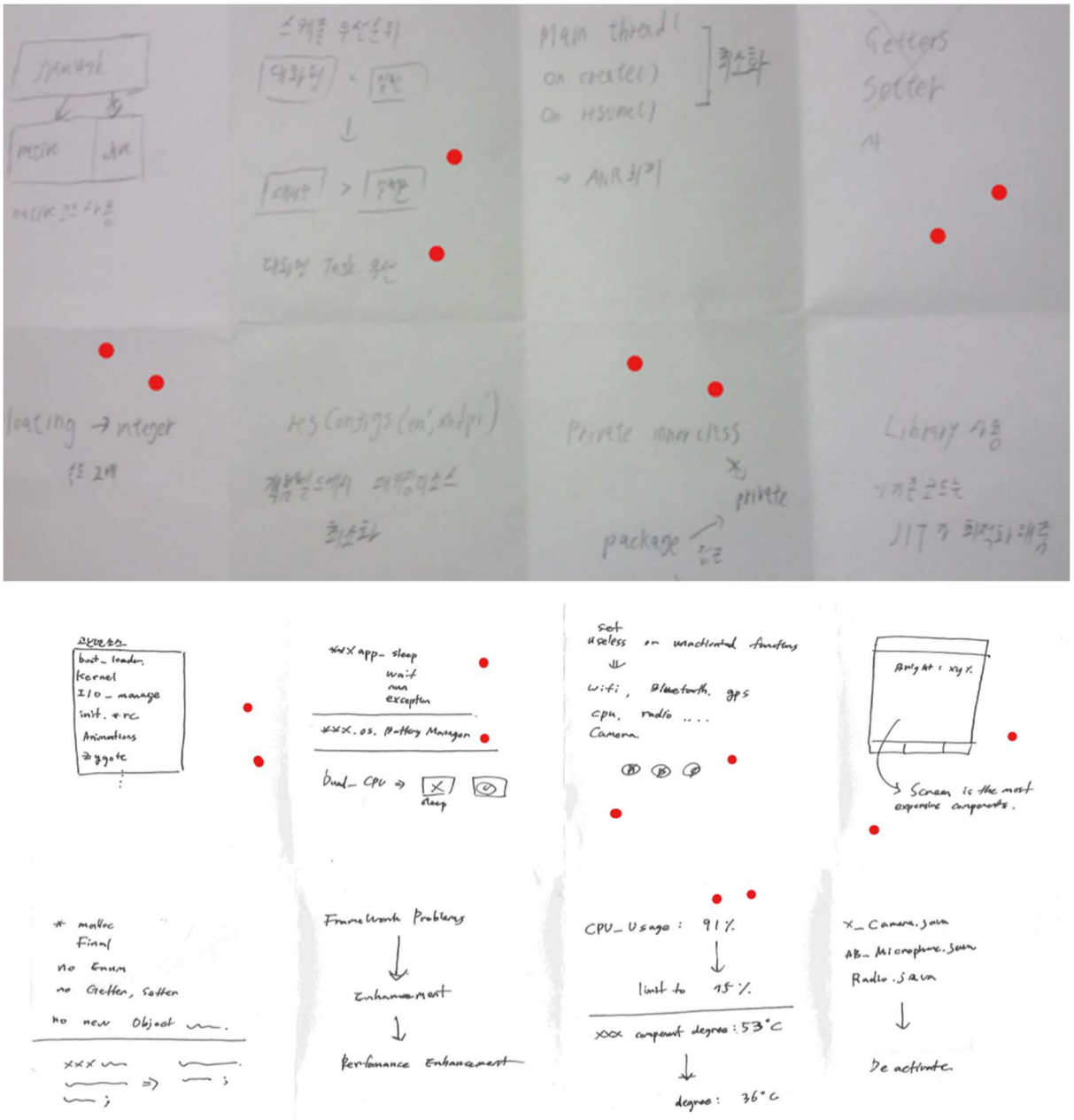


그림 0-4 Crazy`8 및 결과

각각의 팀원은 서로의 Solution을 확인을 하고 AOSP 개선의 알맞은 solution 이 나올 수 있도록 진행하였다. 이에 대한 결과로 오류 개선 사항과 패턴 이상을 확인을 하고 이를 개선하기 위한 방향으로 삼았으면 이에 스토리보드와 프로토 타이핑을 만들었다.

key board & bottom navigation

key board & bottom navigation

key board & bottom navigation

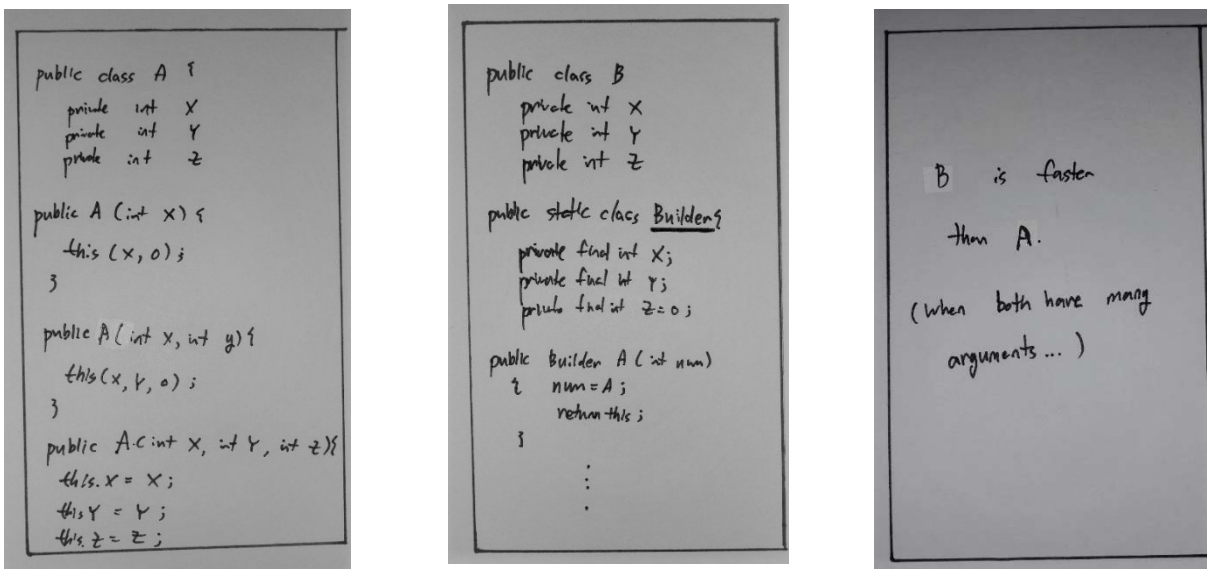
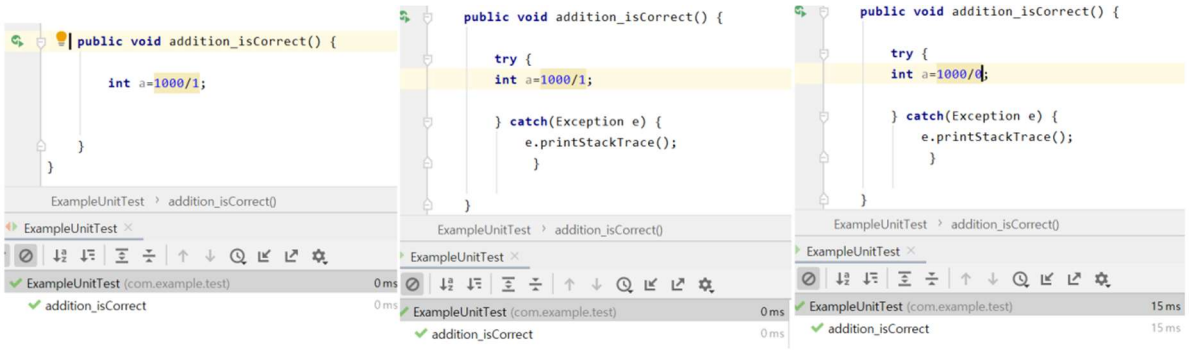


그림 0-5 스토리 보드 제작



try문에서 예외 발생시 추가연산발생

```java
package new1;

public class LayoutParams {

    private final int type;
    private final int flags;
    private final int format;
    private final int w;
    private final int h;
    private final int x;
    private final int y;
    private final int temp;

    public LayoutParams(int type, int flags, int format) {
        this(type, flags, format, 0);
    }

    public LayoutParams(int type, int flags, int format, int w) {
        this(type, flags, format, w, 0);
    }

    public LayoutParams(int type, int flags, int format, int w, int h) {
        this(type, flags, format, w, h, 0);
    }

    public LayoutParams(int type, int flags, int format, int w, int h,int x) {
        this(type, flags, format, w, h, x, 0);
    }

    public LayoutParams(int type, int flags, int format, int w, int h,int x, int y) {
        this(type, flags, format, w, h, x, y, 0);
    }
```

```java
    public Builder(int type, int flags, int format) { // 필수인자 생성자
        this.type = type;
        this.flags = flags;
        this.format = format;
    }

    public Builder w(int num) {
        w = num;
        return this;
    }

    public Builder h(int num) {
        h = num;
        return this;
    }

    public Builder x(int num) {
        x = num;
        return this;
    }
    public Builder y(int num) {
        y = num;
        return this;
    }
    public Builder temp(int num) {
        temp = num;
        return this;
    }

    public LayoutParams_2 build() {
        return new LayoutParams_2(this);
    }
```

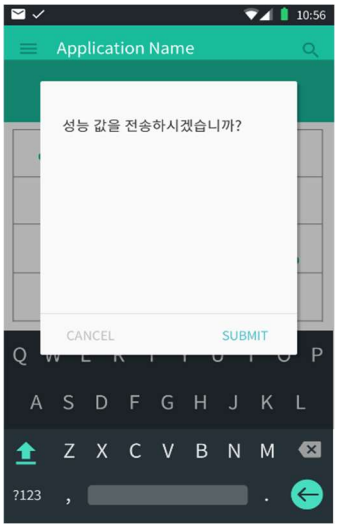


그림 0-6 프로토타이핑 제작

위 과정들을 통하여 제작된 스토리 보드와 프로토 타이핑은 프로젝트 방향성을 정하고 어느 방향에서 개선을 할지를 명확하게 해주었다.

후기 및 진행

이번 프로젝트 디자인 스프린트를 통하여 앞으로 프로젝트의 방향을 어느 정도 확인을 할 수 있다. AOSP 팀은 안드로이드 오픈 소스 프로젝트 (AOSP)에서의 성능을 개선을 하기 위하여 코드를 살펴보고 안드로이드 어플리케이션 level에서 분석이 아닌 framework에서 분석을 하고 있다.

이에 이번 디자인 스프린트에서 확인을 하는 것은 framework level 에서 잘 못 사용되거나 오래된 레거시 패턴들로 인한 성능 약화를 개선을 하고자 하면 몇몇 자바 프레임워크 서비스에서 이를 확인을 할 수 있었다. AOSP 가 약식으로 진행한 코드 분석은 Java 프레임워크단에서 WindowManger이다. WindowManger는 안드로이드 OS에서 자바 프레임워크단에 속하는 시스템 서비스이다. WindowManger는 여러가지 Class 파트가 있으면 특히나 주목을 한 것은 생성자를 작성을 하는 부분이었다. 우리 조는 디자인 스프린트 진행과정, 프로젝트 진행과정에서 이 생성자 패턴이 텔레스코핑 패턴으로 작성이 된 것을 확인하였다. 이에 이를 개선할 수 있는 사항으로 간주하여 Builder 패턴으로 전환이 가능한지를 고려하였다. 성능에 관점에서는 프로젝트를 지속적으로 진행을 하지만 디자인 스프린트 과정에서 프로젝트에 진행 방향성을 확인 할 수 있었다.

또한 디자인 스프린트, 프로젝트를 진행을 함에 있어서 초기 설정안 개선 사항에서의 수정점을 확인을 할 수 있었다. Try-catch에서의 개선 사항이었는데 위 설문조사 결과에서 확인 할 수 있듯이 설문조사 결과를 얻는 과정에서 이 부분을 확인을 하였고 디자인 스프린트를 진행을 하면서 빠른 결과를 확인을 할 수 있었다. 이에 디자인 스프린트의 긍정적인 결과였음을 확인을 할 수 있다.

앞으로의 프로젝트는 크게 3가지로 진행이 될 것이다.

1. 프레임워크 개선사항 추가 도출
2. 개선사항 수정 및 AOSP 커스텀 빌드
3. 수정 커스텀 빌드 테스트 및 AOSP Commit

우선 기존 계획을 하고 있는 WindowManger를 포함을 하여 C로 작성된 하드웨어 프레임워크단 또한 추가 조사할 계획이다.

이러한 AOSP를 빌드 사항에 맞게 커스텀 과정을 진행을 하여 위 개선 사항을 완료하였을 때 테스팅앱과 비교를 통하여 프로젝트의 성능을 확인할 것이다.

마지막으로 성능 평가를 완료한 커스텀 빌드의 반영 사항을 AOSP(Android Open Source Project)을 관리하는 gerrit 에서의 commit 을 통해 위 프로젝트를 마무리 한다.