창의공학기초설계 프로젝트 최종보고서

# 팀 구성원

|  |  |  |
| --- | --- | --- |
| **성명** | **학번** | **역할** |
| 박재선 | 2018440059 | 팀장, 초기 개발환경 구축, 기능개발 |
| 강다현 | 2021920001 | ppt, 영상 제작, 기능개발 |
| 강형원 | 2021920003 | git 관리, 기능개발 |
| 유영호 | 2021920039 | 서류 작성, 영상제작, 기능개발 |

# 과제 개요

아침에는 맑아서 우산을 안 챙겼는데 오후가 되자마자 비가 세차게 쏟아져 곤란했던 적이 누구나 한 번쯤 있을 것이다. 바쁜 아침에 오늘 비가 오는지 알려주는 앱이 있으면 참 편할 것이다. 날씨 앱들은 많이 있지만 아침에 들어가서 보려면 로딩시간이 꽤 오래 걸려 답답하고, 외출 준비를 하다가 날씨를 확인해야 한다는 사실을 잊어버리기도 한다. 날씨가 맑은지, 흐린지보다는 비가 오는 사실만 궁금하다. 사용자가 지정한 장소의 일기예보를 확인하여 사용자에게 지정된 시간에 알려준다면, 사용자가 날씨를 확인하기 위해 소중한 시간을 할애하지 않아도 강우여부를 손쉽게 알 수 있다.

프로젝트 명: Will it rain?

프로젝트 gitHub (깃헙 링크: <https://github.com/moran991231/Will-it-rain>)

회의록 gitHub (깃헙 링크: <https://github.com/moran991231/Will-it-rain/tree/main/meeting-notes>)

# 진행 상황 요약

1. 중간 보고서에 작성된 완료 사항

애플리케이션을 만들기 위해 필요한 소기능들을 먼저 구현하였다. GPS, 알림 보내기, 날씨 API사용하기와 같은 기능을 사용하는 법과 해당 기능을 이 프로젝트에 적용하기 위해서 어떠한 조작을 가해야 하는지 연구하였다. 프로젝트를 원활히 진행할 수 있도록 안드로이드 앱의 생명주기와 권한 설정 같은 시스템의 기초설정 부분을 충분히 이해하고자 하였다.

1. 변경 사항 및 진행 상황

이전에 작성한 소기능들의 코드를 참고하여 응용 기능을 구현하였다. 실질적으로 기능들을 하나의 프로젝트에 통합하기 위해 노력했다. GPS를 이용하여 위치를 가져오고, 이를 API를 이용하기 위한 (X, Y)좌표를 변환하였다. 그리고 좌표를 파일에 출력하여 앱이 재시작 했을 때, GPS 기능을 이용하지 않아도 기존 좌표를 읽을 수 있도록 하였다. 날씨 API를 이용할 수 있는 wrapper 클래스를 만들었고, 우리 프로젝트에 맞게 모듈화 하였다. 날씨 API를 이용하기 위해 설정해주어야 하는 인자들을 시스템 정보(시간 등)을 기반으로 자동으로 설정하도록 구성하였다. 프로젝트를 위한 GUI를 보완하였다.

# 수행 방법

1. GPS
   1. 파일입출력

위치정보 수신권한을 허용하고 GPS 정보를 수신해 현재 위치의 위도, 경도 정보를 받아온다. 위도, 경도 값을 기상청의 격자점(x, y)로 바꾸고, 위치에 해당하는 날씨 정보를 받아온다. 위도와 경도 값을 수신하여 (x, y)값으로 변환하였다. 그러나 이 값은 런타임 변수에 저장된 값이고, 이 값은 앱이 종료되면 사라진다. 앱이 실행될 때마다 GPS의 정보를 가져오는 것은 불필요하다. 이전에 설정한 값을 재활용하면 되기 때문이다. 더불어, 추후에 백그라운드에서 알림을 만들어야 하기 때문에 백그라운드 작업에서 GPS를 이용하는 것은 더욱 번거롭다. 따라서 파일에 이 값을 저장하여 필요할 때 읽을 수 있어야 한다. 따라서 파일 입출력을 통해 xy.txt라는 파일을 생성하고 (x, y)값을 파일에 저장해 두었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

이 좌표에 대한 정보는 MyGps라는 클래스의 정적변수 int x, y에 저장된다. MyGps에 다른 정적 함수인 isXyValid()를 호출하면 x, y가 유효한 값으로 설정되었는지 검사할 수 있다. 만약 이 값이 유효하다면 그대로 API에 (x, y)를 사용하고, 그렇지 않다면 xy.txt파일을 읽어서 (x, y)를 불러온다.

* 1. getLocation

GPS로부터 위치 정보를 새로 받지 않아도 기기에 최근에 저장된 위치 정보가 있으면 그 정보를 가져올 수 있다. 이 앱은 실시간으로 다이나믹하게 움직이는 사용자보다는 일정 지역에서 일정 시간 동안의 날씨 정보를 원하는 사용자를 대상으로 한다. 그리고 날씨라는 물리적 특성은 m보다는 km단위로 변하는 특성이므로 최신 위치 정보가 아니더라도 충분한 정밀도를 보장할 것이라 판단하여 getLocation 함수를 이용하여 위치정보를 가져왔다. 만약 getLocation 함수로도 충분하지 않다면 (getLocation함수가 위치 정보를 null로 반환했다면) 기존의 방식대로 GPS를 사용한다.

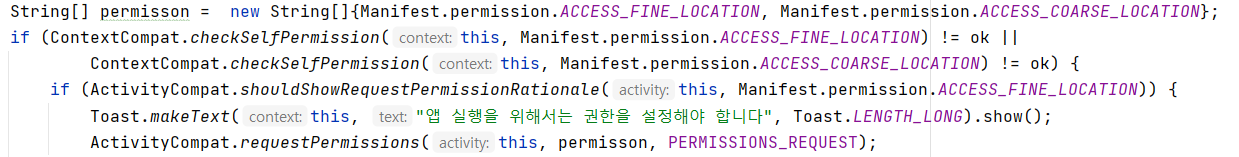
텍스트이(가) 표시된 사진

자동 생성된 설명

* 1. 권한 요청 팝업창

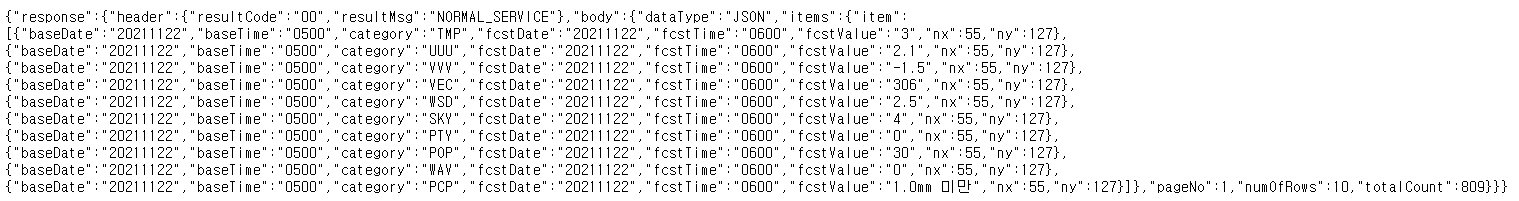
안드로이드 앱은 manifest 파일에 권한을 기입해도 사용자가 승인하지 않으면 사용하지 못하는 기능이 있다. 대표적으로 GPS가 그 예이다. 처음 GPS 소기능을 개발할 때에도 코드에는 이상이 없는데 기능이 동작하지 않아서 문제가 발생하였다. 수동으로 앱 정보-> 권한-> GPS권한 허용을 해주어야 GPS를 사용할 수 있었다. 상용 앱 같은 경우, 권한이 필요할 때 앱이 팝업창을 띄워 권한을 요청한다. 이와 관련된 기능을 이 프로젝트에도 적용했다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 날씨 API

날씨 API 로 데이터 값을 불러오는 것까진 성공했다. API를 통해 얻은 데이터가 어떤 형태인지살펴보자.



하지만 우리에게 모든 데이터가 필요한 것은 아니고 비가 오는 여부만 필요하기 때문에 강수확률에 해당하는 데이터만 필요하다. 그래서 강수확률(POP)데이터만 가져올 수 있게 하고 현재시간을 확인해서 현재시간을 기준으로 API를 호출할 수 있도록 했다.

텍스트, 오렌지이(가) 표시된 사진

자동 생성된 설명

이렇게 날씨 API 전체 기능이 아닌 일부 기능만을 요하기 때문에 API 기능을 프로젝트에서 효율적으로 사용할 수 있도록 Weather라는 클래스를 생성하였다. 중요한 요소들에 대해 아래 간략히 요약하였다.

|  |  |
| --- | --- |
| 항목 명 | 기능 요약 |
| class ApiBaseTime | ms 단위의 시간을 입력받아서 api를 이용하기 위한 시간 관련 문자열을 생성한다. |
| int isGoingToRain(int duration, int x, int y, long now) | 현재 (now)시각에 향후 (duration) 시간동안 (x, y)위치에서 비가 올 최대 확률을 반환한다. |
| boolean isPopValid(int pop) | pop이 0~100 의 유효한 강수확률인지, 에러코드인지 검사한다. |
| String makeNotificatoinText(int duration, int maxPop) | 알림에 표시할 문자열을 만든다. maxPop이 유효하면 정상적인 문자열을 만들고, 유효하지 않으면 maxPop을 에러코드로 사용하여 에러명을 출력한다. |

날씨 API의 래퍼 클래스를 만들고 잘 동작하는지 테스트했다. 간결하게 isGoingToRain 함수 호출만으로 최대 강수량을 알 수 얻을 수 있다.

테스트 결과

-1로 표시된 것은 통신 과정에서 에러가 있었던 것인데, 같은 데이터를 접근할 때에도 문제가 확률적으로 발생하였다. 앱 개발 중 오류가 발생한 줄 알았는데, pc로 접속해도 마찬가지였다. 그 시각 서버의 문제였던 것 같다. 이후에는 잘 작동하였다.

|  |
| --- |
| 동대문구 |
| 출력:    간단하게 알아보고자 duration을 10시간으로 설정하였고 테스트하였다. 강수확률 0%로 맑았다. |

|  |  |
| --- | --- |
| 제주도 | |
| 출력:    강수확률이 30%이었다. | |

1. GUI구현하기

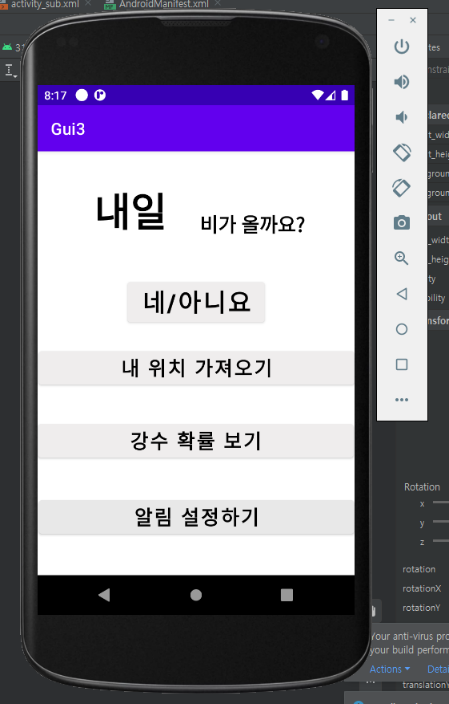
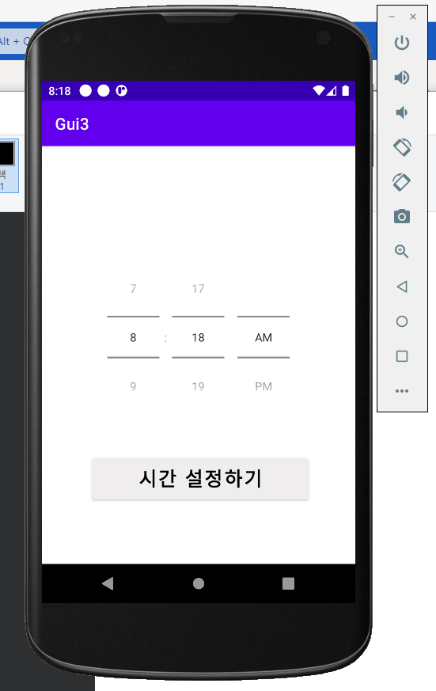
GUI (graphical user interface)는 이 앱을 사용하는 사람이 직접적으로 마주하는 화면이기 때문에 정교하게 만들어야 한다. 본 앱의 기능적인 사항은 위치정보를 받아오는 부분과 알림을 보내는 것이고 이는 GUI로 구현되어야 한다.

* 1. 앱이 제공해야 할 기능

|  |
| --- |
| 앱을 작동시켰을 때, 메인 화면에서 '내 위치 설정하기'버튼을 이용하여 내 위치를 설정할 수 있다. (이때 위치 x, y를 파일에 저장한다.)  알림 시간 설정하기 버튼을 누르면 시간 설정을 위한 화면 전환으로 전환한다. 알림을 보낼 시간을 설정한다. (시간도 파일에 저장한다.) 알림을 취소할 수 있는 기능도 제공한다.  알림 설정 화면에서 돌아가기 버튼을 누르면 다시 메인 화면으로 전환한다.  현재시간 기준으로 비가 오는지 확인하기 버튼을 제공한다. 백그라운드 작업이 아닌 포그라운드 작업으로, 설정한 알림 시간 외에도 언제든 궁금하면 앱을 실행하여 확인할 수 있다.  백그라운드 작업으로 알림이 동작한다. 알림이 울릴 시간 즈음이 되면 api를 이용해 정보를 받아오고, 푸시 알람을 보낸다. |

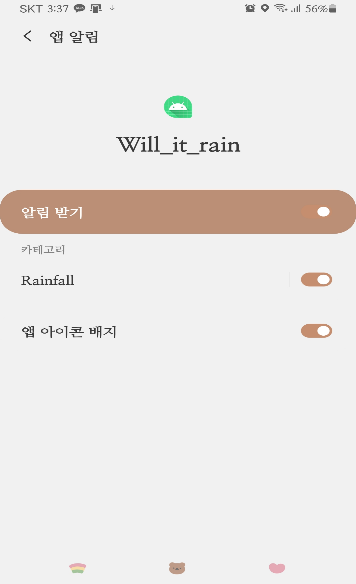
* 1. 결과

Gui로 Will it rain을 만들었다. 초기 언어 설정이 자바가 아니라 코틀린으로 되어 있어서 이를 자바로 바꾸어 저장했다. 폰트와 글의 크기를 정할 때 애뮬레이터와 안드로이드 스튜디오내에 디자인이 다르게 보이는 문제가 있었는데 이는 textsize를 지정할 때 android:textsize와 tool:textsize가 나누어져 있고 android:textsize를 설정하니 해결이 됐다. Intent를 이용하여 알림설정하기를 누르면 알림이 설정하는 칸이 뜨도록 설정했다. 알림을 받아올 때 시간을 지정하는 부분에서 timepicker를 사용했다.

1. Push Notification

프로젝트에서 다른 기능들과 알림을 보내는 기능이 독립적으로 이뤄지고, 개발을 편하게 하기 위해 알림보내기와 관련된 기능들만 모아서 MyNotification이라는 클래스를 만들었다. MyNotification의 객체를 생성하고 makeNotification(...)만 호출해주면 원하는 내용의 푸시 알림을 만들 수 있다.



안드로이드 API 레벨 31부터는 알림을 생성할 때, IMMUTABLE혹은 MUTABLE 속성을 꼭 지정해야 한다. 지정하지 않으면 에러가 뜬다.

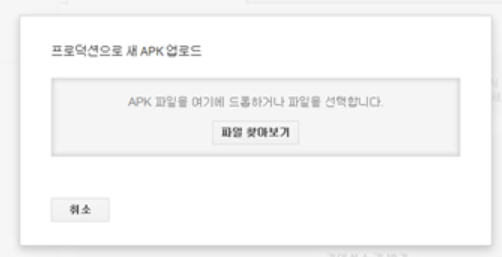
안드로이드 API 레벨 26부터는 알림을 생성할 때 채널을 생성해야 한다. Rainfall이라는 채널을 생성해 주었다. 이는 앱 정보에 들어가서 확인할 수 있다.

1. Google Play 마켓에 앱 등록하기

마켓에 앱을 등록하기 위한 절차를 조사하였다. [출시 | Google Play Console](https://play.google.com/intl/ko_ALL/console/about/releasewithconfidence/) 에 접속해서 play console로 이동하여 로그인과 인증을 한다. 개인정보를 입력하고 개발자 계정을 만든다. 25$를 결제하고 개발자 계정을 생성한 뒤, “Google Play에 Android앱을 출시”를 클릭하고 APK업로드를 한다. 최종적으로 앱의 제목, 설명, 미리보기 이미지, 앱의 가격 등을 설정하고 앱을 출시한다.

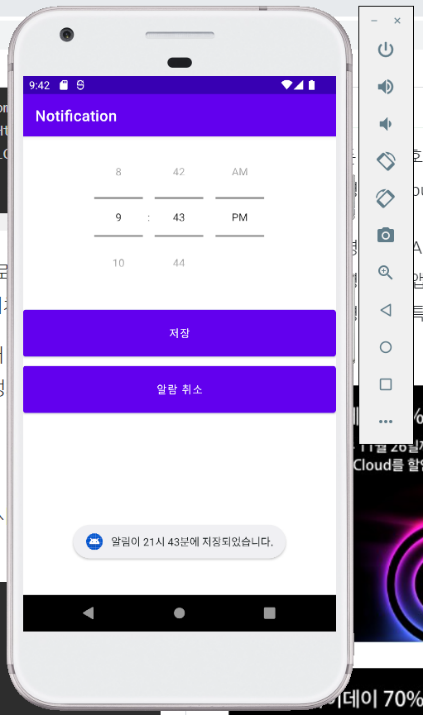
 텍스트이(가) 표시된 사진

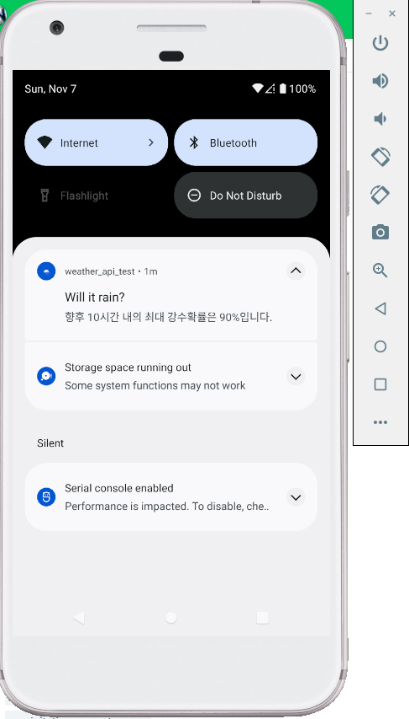
자동 생성된 설명



1. 통합 프로젝트 만들기

GPS기능, 위치정보 (x, y) 파일 입출력 기능, 날씨 API기능, API 시간 자동 설정 기능, 푸시 알림 기능, 예약 알람 기능 GUI에 통합

 텍스트이(가) 표시된 사진

자동 생성된 설명 

소기능들을 정리하고, 래퍼 클래스를 만들어 두었기 때문에, 하나의 프로젝트로 통합하는 과정은 크게 어렵지 않았다. 각각의 기능을 사용하기 위해서는 많은 함수를 호출해야 하지만, 래핑을 통해 실제 루틴에서는 1~2개의 함수만 사용하면 되도록 잘 구성하였다. 이후부터는 에뮬레이터보다는 실제 폰으로 디버깅을 진행하였다.

통합하는 과정은 9할 이상이 오류를 해결하는 과정이다. 프로그램이 효율적으로 수행되도록 고민도 많이 하고, 제대로 동작하지 않아서 새로운 방법을 적용해보았다.

* 1. 파일 입출력

안드로이드 앱은 저마다의 저장공간을 갖는다. /data/data/(패키지명)/디렉토리 내의 공간이다. 자바 언어에서 파일 입출력을 하기 위해 사용하는 클래스들과, 메서드들이 있다. 그러나 안드로이드 앱에서는 그것들을 자유롭게 사용할 수 없다. 보안상의 이유이다. 따라서 mainActivity. openFileOutput(...) 과 같이, MainActivity의 객체를 통해 저장공간에 액세스한다. 이 방식은 특별한 권한을 받지 않아도 된다. 소기능을 개발할 때에는 이와 같은 방법으로 잘 동작하였다. 그러나 백그라운드상에서 알림을 보내는 상황에서는 어떨까? 지정된 시간에 알람이 발생하여 이 이벤트를 처리하기 위해 MyAlarm이라는 클래스에 리시버를 구현하였다.

|  |
| --- |
|  |

이 문맥에서는 MainActivity의 객체를 사용할 수 없다. (백그라운드에서 호출되므로 MainAcitivy의 객체가 없기 때문이다.)

텍스트이(가) 표시된 사진

자동 생성된 설명

다만 openFileOutput/openFileInput 메서드는 Context라는 클래스에 정의되어 있고, 상속 계층을 잘 살펴보면 MainActivity는 Context를 상속받은 것을 알 수 있다. 따라서 onReceive의 매개변수로 전달된 context를 mainActivity의 객체 대신 사용하면 된다.

* 1. 스레드 내에서 API 이용

날씨 API는 원하는 데이터를 가리키는 정보를 URL로 구성하여 해당 주소의 정보를 인터넷에서 가져온다. 이러한 통신 과정은 통신장애가 발생할 수도 있고, 오래 걸리며, 비동기적으로 이루어져도 되므로 메인 스레드에서 수행할 수 없다. 그래서 이러한 통신을 할 때에는 스레드 풀에서 스레드를 할당하여 작업을 수행해야 한다. 안드로이드 예제는 ExecutorService를 사용하라고 하였다. 이를 이용하니 onReceive함수가 호출되고 난 후 2~3분이 지난 뒤 늦게 executor의 수행이 완료되었고 이 때문에 알림이 도착하기까지 너무 오래 걸렸다. 인터넷 통신이 오래 걸리고, push알림을 요청하고 안드로이드가 알림을 보내는 과정에서 오래 걸린다고 생각하였는데, 디버깅을 반복한 결과, executor의 작업이 끝날 때까지 onReceive에서 대기하는데, 대기함수가 비동기적으로 동작해서인지 대기시간이 너무 긴 것이 문제였다. 그래서 일반 Thread를 사용하였다. 문제없이 동작하며, 매우 빠르게 동작한다. onReceive함수가 호출된 이후로 10초 이내에 푸시 알림이 도착하였다.

* 1. Doze mode

처음에는 반복 알림을 안드로이드 개발자에서 제공하는 예제에 따라 AlarmManager. setRepeating(...) 을 이용하여 반복 알람을 등록하였다. 테스트할 때에는 반복주기를 약 3분간격으로 맞추고 에뮬레이터를 이용하였는데, 이때에는 잘 동작하였다. 그러나 직접 안드로이드 폰으로 디버거를 연결하지 않고 테스트해보니, 알람이 느리게 도착하고, 알람이 반복되다가 어느 시점에서는 끊기는 것이 발견되었다. 예를 들어, 알람을 1분 후에 설정하면 알람이 오지만, 1시간 후에 도착하도록 설정하면 알람이 도착하지 않았다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Android Developer에 위와 같은 경고가 있었다. 잠자기 모드와 관련해서 조금 더 자료를 찾아보니 Android 6.0(API 수준 23)부터 Android는 두 가지 절전 기능을 도입했고, 그 중 하나가 배터리를 절약하기 위한 잠자기 모드(doze mode)라는 것이다. (doze는 ‘꾸벅꾸벅 졸다’라는 뜻으로, 전원은 유지하되 앱 기능의 사용을 최소화한다는 은유로 사용된 것 같다.) 휴대전화를 n시간동안 사용하지 않거나 n시간동안 움직이지 않았을 때 등 일련의 기준에 의해 배터리 절약 모드에 들어간다.

보다 정확한 시간에 알람을 발생하고, 반복되는 알람을 보장하기 위해서 setExactAndAllowWhileIdle이라는 함수를 사용하였다. 이 함수는 작업을 1회 예약해 주므로, 리시버가 동작할 때마다 다음 작업을 예약해야 한다. 이로써 문제가 해결된 것 같았지만, 테스트해본 결과 특정 시각에 알람이 도착하는 것이 아닌 특정 시간 이후에 휴대전화를 켜면(유휴 상태를 종료하고 활성화하면) 그제서야 알람이 도착하였다. 혹은 알림이 발생해도, API를 이용하는 부분에서 네트워크 에러가 발생하였다.

자료를 찾아본 끝에 setAlarmClock 함수를 찾았다. setExactAndAllowWhildIdle은 해당 시간에 앱을 저전력 모드에서 해제하여 동작할 수 있도록 하지만, wake lock을 해제하지는 못한다. setAlarmClock은 해당 앱 외에도 다른 장치들의 저전력모드를 해제한다. 이로써 예정된 시간에 정해진 동작을 수행하는 알람 설정에 성공하였다. 배터리 최적화 옵션도 무시하는 권한을 부여하기도 했다.

아래는 앱의 동작 사진이다. (실제 기계 테스트)

|  |  |  |
| --- | --- | --- |
| 내 위치 가져오기 | 시간 설정하기 | 알람 취소하기 |
|  |  |  |

Time Picker의 모습이 (오전/오후) (시간) (분) 이다. 디자이너와 에뮬레이터 화면에서는 (시간) (분) (AM/PM)이었는데, 안드로이드에서 GUI 컴포넌트를 불러올 때 현지의 문화를 반영해서 그런 것이다.

# 문제 상황

1. GPS에서 파일 입출력 문제

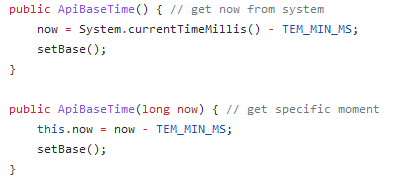
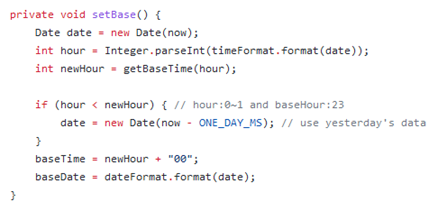
GPS에서 받아온 (x, y)값을 파일에 입출력하는 과정에서 openFileOutput과 openFileinput을 사용하였는데 원활하게 실행이 되지 않았다. 문제가 발생한 원인은 객체인 mainactivity를 앞에 써주지 않은 것이었다. 해당 내용을 mainactivity.openFileOutput과 mainactivity.openFileInput으로 바꿔주었다.

1. API에서 기준시간 문제

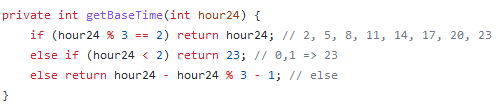
강수확률(POP)에 대한 데이터를 가져오는 것은 문제없이 해결했지만 현재시간을 가져오는 것에서 문제가 발생했다. 현재시간을 가져올 때 3시간 단위로만 가져올 수 있고 ex) 0200, 0500, 0800 등 그리고 기상청에서 정보를 업데이트 하는데 10분 정도 소요된다. ex) 0200~0210 사이에는 02시의 정보를 사용할 수 없다. 그러다 보니까 현재 시각이 00:00~02:10 사이에는 전날의 23시의 정보를 사용해야 했다. 그래서 현재시간을 무작정 가져오는 것이 아니라 요구에 맞춰서 코드를 수정한 결과, 마침내 원하는 결과를 얻을 수 있었다. System.currentTimeMillis() 함수를 통해서 현재 시간을 밀리 초 단위로 가져오고 그것을 이용해서 각각 10분과 24시간을 표현했다.



그리고 업데이트 하는데 걸린 시간 10분을 해결하기 위해서 현재 시간에서 10분을 뺐다. 또한, 0시나 1시 일 때는 전날 23시의 정보를 사용하게끔 함수를 만들었다.

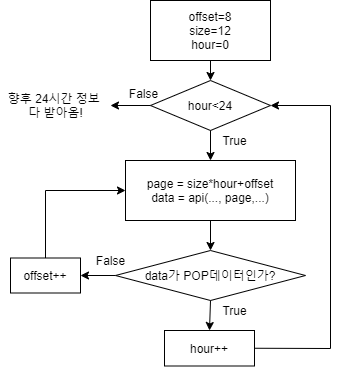
 

마지막으로 3시간 단위마다 시간을 불러올 수 있기 때문에 이것은 나머지 연산자(%)를 통해 해결했다.



1. API에서 원하는 데이터를 가져오는 데 발생한 문제

테이블이(가) 표시된 사진

자동 생성된 설명

기준시각 (2, 5, 8, 11, 14, 17, 20 23시)에서 매 1시간 마다 향후 72시간에 대한 정보를 API로 얻을 수 있다. 1시간에는 위 표에 나타난 항목들의 정보가 포함된다. 시간별로 제공하는 항목이 다를 수 있다.

한 기준시각에는 (예측 72시간)x(항목 약 12개) = 약 700~900개의 정보가 제공된다. 제공되는 항목은 API호출 기준시각과 우리가 원하는 강수확률 정보를 얻기 위해서는 기준시각-예측시간 간의 관계를 명확히 파악하여 규칙을 작성하거나, 전체 데이터를 받아와서 원하는 데이터를 파싱해야 했다. 고민 끝에 데이터를 1개씩 24번(24시간 치) 가져오기로 하였다. 그러나 덜 엄격한 방식을 적용하여 가져온 데이터가 POP 이 아니면 그 다음 데이터를 가져오고, POP이면 1시간 후의 데이터를 가져오는 방식으로 했다. 전체적으로 보면 데이터를 24+a 번 가져오게 되지만 복잡한 관계를 구현하지 않은 쉬운 방법과 적은 자원으로 문제를 해결하였다.

# 추가해볼 수 있는 기능

1. 예정된 시간에 날씨 정보를 받아보는 것뿐만 아니라, 원할 때는 앱에 들어가서 바로 강수확률 정보를 확인한다.
2. 앱을 켰을 때, 위치정보가 유효하게 설정되었는지 확인한다.
3. Time picker 화면으로 이동했을 때, 알람이 설정되어 있다면 Time Picker의 시간을 설정한 알람의 시간으로 보여준다.
4. 알람을 여러 개, 여러 장소에 대해 설정할 수 있다.
5. 앱의 아이콘과 GUI를 다듬어 사용자에게 심미성을 어필할 수 있다.
6. 푸시 알림음을 소리/진동 등으로 선택할 수 있다.
7. 체감온도를 알려준다. 실제로 외부활동을 진행할 때, 기온과 더불어 습도와 바람 세기가 복합적으로 계산된 체감온도가 날씨에 맞는 옷차림을 정하는데 큰 도움이 된다.

# 수행 효과

오늘 비가 오는지 빠르게 확인할 수 있어서 바쁜 아침에 시간을 절약하고, 비가 온다면 애플리케이션이 알림을 띄워 우산을 챙길 수 있다.  Java와 안드로이드 환경을 이해할 수 있다. 여러 패키지와 API를 사용해 보며 응용능력을 키울 수 있다. 나아가, 애플리케이션을 앱 마켓에 출시하기 위해 어떤 절차가 필요한지 경험을 통해 알 수 있고, 그 과정에서 겪는 문제를 해결해보며 상황 대처능력도 키울 수 있다.

테스트를 진행하며 아침에 알람이 도착하도록 오전 6시에 알람을 설정해 두었다. 푸시 알림으로 강수확률 정보가 도착하기 때문에 휴대전화의 잠금을 해제하지 않고도 눈길 한 번으로 그날의 강수확률을 확인할 수 있어서 편리했다.

# 수행 환경

Win10 안드로이드 스튜디오로 앱 개발 전반전인 부분을 진행하고 있다. 에뮬레이터내에서 안드로이드 버전 API 레벨 21 이상에서 동작하고, API 레벨 31도 타겟하여 코드를 작성하고 있다. 가지고 있는 안드로이드 폰으로 테스트를 진행하고, 에뮬레이터로 최신 버전을 테스트하고 있다.