

진도보고서

팀원 20160402 최성원
20160369 박준홍
20212897 도호준

작성일

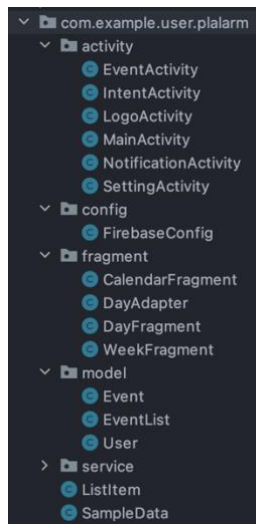
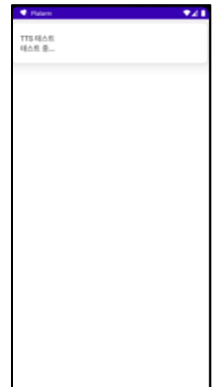
2022.11.29

성원



달력의 월간, 주간, 일일을 표현하는 액티비티를 MainActivity에서 표현되는 Fragment로 변환하여 버튼을 통해 전환 할 수 있도록 함. 또한 알림 기능을 테스트 할 수 있는 NotificationActivity도 버튼을 추가해 MainActivity에서 접근 할 수 있도록 구현. 앱 설계 시 좌우 슬라이드로 Fragment 전환이 일어날 수 있도록 구상을 하여 현재는 편의를 위해 버튼방식으로 테스트하고 배포 전 단계에 슬라이드 방식으로 변경하려고 함.

푸쉬 알림을 보내는 NotificationActivity를 생성하고 액티비티 내에서 버튼을 누르면 알림이 오도록 함. 달력에 일정이 표시되는 기능이 아직 구현되지 않아 정해진 알림이 출력되도록 하고 알림의 이름이 tts객체로 음성 출력 되도록 구현. 따라서 액티비티 내에서 버튼을 클릭하면 title과 content를 담은 푸쉬알림이 음성알림과 함께 출력됨. 액티비티 내에서 구현 된 것으로 이를 백그라운드 서비스에서 동작하도록 수정해야함.



유지보수가 용이하도록 클래스를 각각의 기능에 대응되는 이름의 패키지로 정리하고 전체적으로 리팩키징 함.

1주차에 파이어베이스 데이터베이스 접근방법을 정의하기 위해 FirebaseConfig 클래스에 기본적인 CRUD 작업들을 메소드로 정리하였는데 데이터베이스를 가져오는 과정이 log로는 출력되나 엔티티 객체에 담아 반환하려고 하니 null만 반환이 됨. 그 이유가 데이터베이스를 가져오는 방식이 단순히 getter형식의 메소드로 받아오는 것이 아니라 비동기적으로 처리하는데 데이터베이스에 접근 하기 전에 메소드에서 return을 해버려서 빈 데이터를 얻게 되는거라 생각이 됨.

따라서 데이터를 사용할 MainActivity 내에서 직접 데이터 베이스에 접근하고 값을 사용하는 로직으로 구현방법을 변경해야함.

준홍 Day_Activity의 Data를 관리하는 DayAdapter와 SampleData를 Event의 객체로 변동함. 즉, Firebase의 데이터와 연동하기 쉽게 구조를 변경. 목업 데이터를 직접 메모리에 추가해 컴파일 후 바로 확인 할 수 있도록 하였음.

```
import ...

public class SampleData {

    String collectionPath = "user";
    EventList eventList = new EventList();
    Schedule schedule = new Schedule();

    public EventList getEventItems(){


        Event event1 = new Event("1", "content: 2", "startTime: 3", "endTime: 4", "intentApp: %");
        Event event2 = new Event();
        Event event3 = new Event();

        eventList.add(event1);
        eventList.add(event2);
        eventList.add(event3);
        return eventList;
    }

    public Schedule getSchedule(){

        return schedule;
    }
}
```

기존 Week_Activity의 구현되었던 Material Calendar(Week버전)대신 TimeTableLayout을 이용하여 새로 구현.



The screenshot shows a mobile app interface with a top navigation bar containing icons for notifications, a calendar, and settings. Below the bar are tabs for 'CALENDAR', 'WEEK', 'DAY', and 'NOTI'. The 'WEEK' tab is selected, displaying a weekly calendar grid. The grid shows days of the week (일, 월, 화, 수, 목, 금, 토) and hours (9 to 4). Three events are visible: '사인페 정보성' on Monday (red), '사인페 정보성' on Wednesday (green), and '사인페 정보성' on Friday (orange). A '+' button is at the bottom right.

```
public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
    View view = inflater.inflate(R.layout.fragment_week, container, attachToRoot false);
    timetableView = view.findViewById(R.id.timetable);
    LocalDate now = LocalDate.now();
    ArrayList<String> week = new ArrayList<>(Arrays.asList("일", "월", "화", "수", "목", "금", "토"));
    ArrayList<Integer> Year = new ArrayList<>(Arrays.asList(31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31));
    int y = now.getYear();
    if(y % 4 == 0) Year.set(1, 29);

    MakeData("사인페", "p", "정보성", SD: 0, S_h: 9, S_m: 0, E_h: 11, E_m: 30);
    MakeData("사인페", "p", "정보성", SD: 4, S_h: 11, S_m: 0, E_h: 13, E_m: 30);
    MakeData("사인페", "p", "정보성", SD: 2, S_h: 10, S_m: 0, E_h: 14, E_m: 30);

    public void MakeData(String t, String p, int SD, int S_h, int S_m, int E_h, int E_m)
    {
        schedule.setClassTitle(t);
        schedule.setClassPlace(p);
        if(SD >= 0 && SD <= 6) schedule.setDay(SD);
        schedule.setStartTime(new Time(S_h, S_m));
        schedule.setEndTime(new Time(E_h, E_m));
        schedules.add(schedule);
        timetableView.add(schedules);
    }
}
```

추가 요일 strings.xml을 이용하여 제작, 정적인 데이터를 이용하여 화면 출력

현재 Calendar_Activity의 AlertDialog를 제작하였으나 Fragment로 변경이 되어, AVD에 확인이 되지 않은 점을 인지하고 수정 중(Fragment변경 후 코드를 변경하였으나 동작이 되지않음.)

TimeTableLayout의 요일 표현은 구현하였으나, 세로(시간 표현)에 대하여 추가적으로 작업을 찾아보고 수정 단계 중.

```
// sharedPref 로 지정된 Fragment 먼저 가져오는 부분
// TODO : sharedPref 로 이 부분을 가져올 수 있어야 함 (DONE)
SharedPreferences sharedPref = PreferenceManager.getDefaultSharedPreferences(context.this);
int DEFAULT = sharedPref.getInt(key, "calendarSpinner", defValue: 0) + 1;
fragmentView(DEFAULT);

// sharedPref 로 notification button 의 상태 저장
boolean NOTIFICATION_MUTE = sharedPref.getBoolean(key, "NOTIFICATION_MUTE", defValue: false);
if (NOTIFICATION_MUTE) {
    notificationOnIcon.setVisibility(View.INVISIBLE);
    notificationOffIcon.setVisibility(View.VISIBLE);
}
}
```

SharedPreferences를 활용하여 Calendar Spinner의 저장값을 MainActivity에서 적용할 수 있게 코드를 수정하였고, 안드로이드 가상 장치의 SharedPreference 파일을

AVD 파일 탐색기를 통해 확인하여 정상적으로 저장되는지, 또 테스트를 하여 작동 여부를 확인하였다.

또한 SharedPreferences가 Notification Mute 버튼에도 정상적으로 작동하도록 구현 및 테스트까지 완료하였다. 다만, 위의 기능이 실제로 작동하려면 별도의 구현 작업이 필요하다. 따라서 Notification 기능의 구현이 완전히 끝난 이후에, 이 버튼을 연결하여 실제로 Notification Mute가 켜진 여부에 따라 버튼의 기능의 실질적 구현을 할 예정이다. 오른쪽 위 사진은 해당 부분의 코드이며, 아래는

```
//soundButton 을 클릭한 경우, visibility 값을 바꾸는 부분
// TODO : sharedPref 로 MUTE 의 ON/OFF 상태 저장 가능해야 함 (DONE)
SharedPreferences sharedPref = PreferenceManager.getDefaultSharedPreferences(context.this);
if (view == soundButton) {
    if(notificationOnIcon.getVisibility() == View.VISIBLE) {
        notificationOnIcon.setVisibility(View.INVISIBLE);
        notificationOffIcon.setVisibility(View.VISIBLE);
        SharedPreferences.Editor editor = sharedPref.edit();
        editor.putBoolean("NOTIFICATION_MUTE", true);
        editor.apply();
    }
    else {
        notificationOnIcon.setVisibility(View.VISIBLE);
        notificationOffIcon.setVisibility(View.INVISIBLE);
        SharedPreferences.Editor editor = sharedPref.edit();
        editor.putBoolean("NOTIFICATION_MUTE", false);
        editor.apply();
    }
}
}
```

SharedPreferences 파일에 저장이 정상적으로 됨을 보이는 파일의 스크린샷이다.

```
17 // public void createNotificationChannel(String channelId, String channelName, int importance){
18 //     if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.O){
19 //         NotificationManager notificationManager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
20 //         notificationManager.createNotificationChannel(new NotificationChannel(channelId, channelName, importance));
21 //     }
22 // }
23 //
24 //
25 // public void createNotification(String channelId, int id, String title, String content){
26 //     NotificationCompat.Builder builder = new NotificationCompat.Builder(this, channelId)
27 //         .setPriority(NotificationCompat.PRIORITY_HIGH)
28 //         .setSmallIcon(R.drawable.logo)
29 //         .setContentTitle(title)
30 //         .setContentText(content)
31 //         .setDefaults(Notification.DEFAULT_SOUND | Notification.DEFAULT_VIBRATE);
32 //
33 //     NotificationManager notificationManager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
34 //     notificationManager.notify(id, builder.build());
35 // }
36 // }
```

왼쪽은 Notification을 구현하기 위한 파일의 일부분이다.

Notification.java라는 새로운 파일을 만들어 해당 부분에 Notification 구현을 하려고 하였으나, 많은

Error 및 많은 결함을 보여 해당 파일을 전부 주석 처리하였다. Notification 기능은 이미 틀이 잡혀있으므로 해당 파일은 참고로만 사용할 예정이며, 구체적 구현은 추후 진행할 예정이다. 대신, Calendar Fragment 상에 사용자가 생성하고, Firebase에 전송 및 다운로드 받은 Event List로부터 Event 정보를 얻어 Fragment의 해당 날짜에 Event가 있음을 알리는 기능의 구현을 진행하고 있다.

No.	Tasks	Status	Description	11월																												12월		
				2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2
1	Logo Activity	X																																
2	Main Activity	X	Fragment로 설계해 전달																															
3	Setting Activity	X	SharedPreferences 전역상태 저장																															
4	Alert	X	Intent 작업 완료 후 앱 이동까지																															
5	Week Activity	X																																
6	Day Activity	X	일정 로직 완성 후 Merge																															
7	Calendar Activity	X																																
8	TTS	X	일정 객체 모델링 후 바로 시작																															
9	Firebase Establishment and Connection	X																																
10	Intent Object	X																																
11	Debug & Test	X	각자 개발한 내용 조율 및 테스트																															

완료작업 중작업 실패호준준홍성원

현재 개발 일정

No.	Tasks	Status	Description	11월																														12월	
				2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2	
1	Logo Activity	X	200ms후 Main으로 이동되도록 함																																
2	Main Activity	X	fragment 구현중																																
3	Setting Activity	X																																	
4	Alert	X																																	
5	Week Activity	X	내 작업 완료																																
6	Day Activity	X																																	
7	Calendar Activity	X	내 작업 완료																																
8	TTS	X	일정 삽입 기능 완료 후 재개																																
9	Firebase Establishment and Connection	O	엔티티 정의 및 적재방법 정의																																
10	Intent Object	X	모든앱의 아이콘 및 패키지명 가져옴																																
11	Debug & Test	X																																	

완료

작업 중

작업 실패

호출

준중

성원

Github Project Kanban Board