

모바일 프로그래밍

04 메뉴와 대화 상자 2

2017 2학기

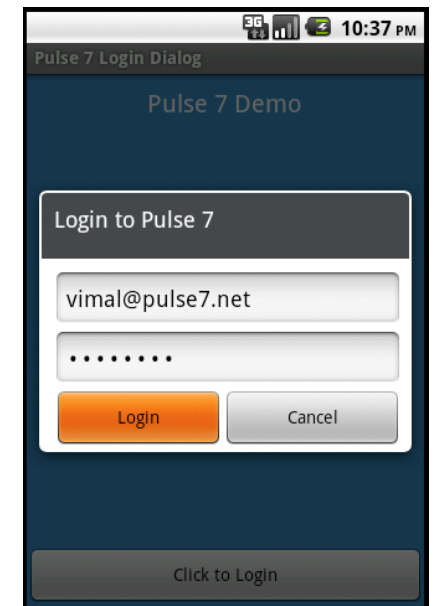
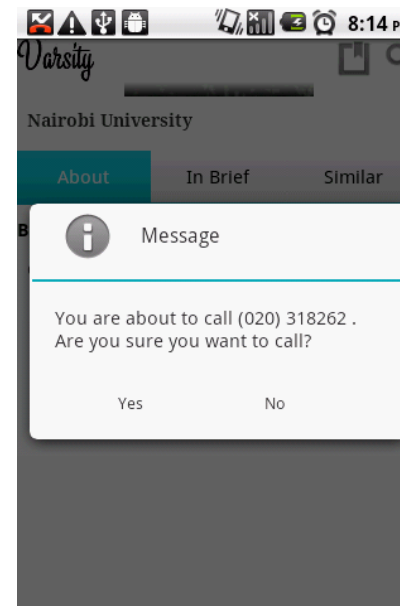
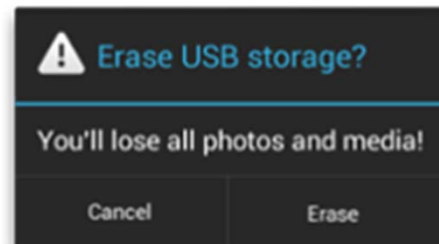
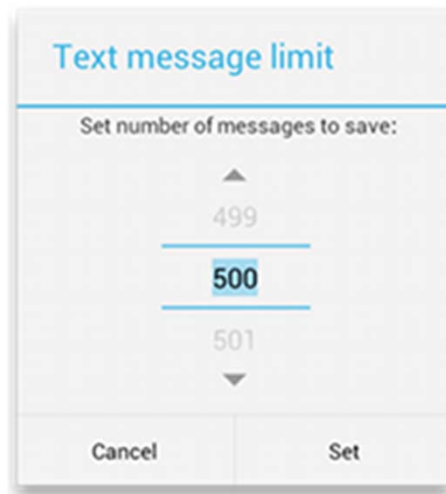
강승우

04-2 대화 상자

대화 상자 (Dialog)

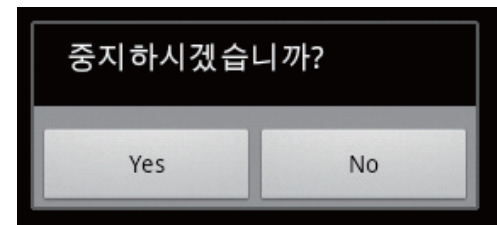
<https://developer.android.com/guide/topics/ui/dialogs.html>

- 사용자에게 메시지를 출력하고 사용자로부터 입력을 받는 인터페이스 (보통 화면 전체를 채우지 않는 작은 윈도우로 표시)
- 사용자에게 내용을 좀더 확실히 주지시켜야 할 경우, 계속 진행할 지의 여부를 선택하게 할 때 사용
 - 보통 애플리케이션 동작의 진행을 위해서는 사용자가 액션을 수행해야 함



대화 상자 종류

- Dialog 클래스가 여러 가지 dialog의 base class
- AlertDialog
 - 제목, 0-3개의 버튼, 선택 가능 항목(체크 박스, 라디오 버튼) 리스트를 표시할 수 있는 대화 상자
- DatePickerDialog / TimePickerDialog
 - 사용자가 날짜나 시간을 선택할 수 있도록 하는 대화 상자



<https://developer.android.com/reference/android/app/Dialog.html>

AlertDialog

- 1. 제목
 - 선택 항목
 - 콘텐츠 영역에 상세한 메시지, 목록 또는 사용자 지정 레이아웃이 채워져 있는 경우에만 사용
 - 단순한 메시지 경우 제목은 없어도 됨
- 2. 콘텐츠 영역
 - 메시지, 목록, 사용자 지정 레이아웃 표시 가능
- 3. 작업 버튼
 - 주로 확인(OK, positive), 취소(Cancel, negative) 2개의 버튼
 - 중립적(neutral) 버튼도 가능



제목이 없는 AlertDialog

AlertDialog 생성

- AlertDialog.Builder 클래스: AlertDialog 생성에 필요한 API 제공
- AlertDialog.Builder 객체 생성
 - AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
- 대화상자 설정
 - setTitle() - 제목
 - setMessage() - 내용
 - setPositiveButton() – 보통 OK/확인 버튼
 - setNegativeButton() – 보통 Cancel/취소 버튼
 - setSingleChoiceItems() – 라디오 버튼 목록
 - setMultiChoiceItems() – 체크박스 버튼 목록
- 대화상자 객체 생성
 - AlertDialog dialog = builder.create();

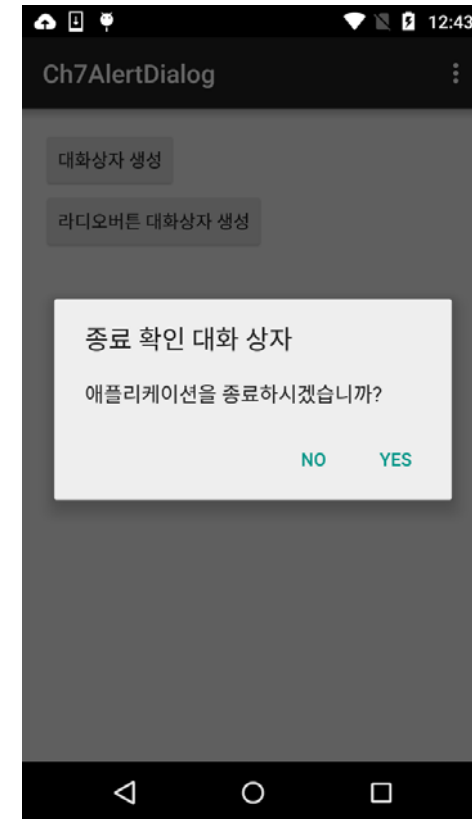
} 각 버튼 유형에는 하나의 버튼만

set 메소드는
Method chaining 방
식으로 사용 가능

https://en.wikipedia.org/wiki/Method_chaining

AlertDialog 예제

- Activity 클래스에 정의된 관련 메소드를 이용하는 방법
 - 예제 프로젝트: Ch7AlertDialog
 - 이용 메소드
 - showDialog: public method
 - onCreateDialog: protected method
 - 위 메소드들은 현재 deprecated
→ 대신 DialogFragment 이용
- DialogFragment 이용하는 방법
 - 예제 프로젝트: Ch7DialogFragment



1.

Activity 클래스 메소드 이용

AlertDialog.Builder
객체 생성

대화상자 설정

대화상자 객체 생성

```
@Override
protected Dialog onCreateDialog(int id) {
    switch (id) {
        case DIALOG_YES_NO_MESSAGE:
            AlertDialog.Builder builder = new AlertDialog.Builder(this);
            builder.setTitle("종료 확인 대화 상자")
                .setMessage("애플리케이션을 종료하시겠습니까?")
                .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        finish();
                    }
                })
                .setNegativeButton("No", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        dialog.dismiss();
                    }
                });
            AlertDialog alert = builder.create();
            return alert;
        }
    return null;
}
```



```
private static final int DIALOG_YES_NO_MESSAGE = 1;
```

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);
```

```
    Button b = (Button) findViewById(R.id.button);  
    b.setOnClickListener(new View.OnClickListener() {
```

```
        @Override
```

```
        public void onClick(View v) {
```

```
            showDialog(DIALOG_YES_NO_MESSAGE);
```

```
        }
```

```
    });
```

```
}
```

2. DialogFragment 이용

- 대화상자를 생성하는 방법은 앞의 예제와 동일
- 차이점은?

```
public static class ButtonDialogFragment extends DialogFragment {
    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        // Use the Builder class for convenient dialog construction
        AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
        builder.setTitle("종료 확인 대화 상자")
            .setMessage("애플리케이션을 종료하시겠습니까?")
            .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    getActivity().finish();
                }
            })
            .setNegativeButton("No", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    dialog.dismiss();
                }
            });
        // Create the AlertDialog object and return it
        return builder.create();
    }
}
```

@Override

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);
```

```
    Button b = (Button) findViewById(R.id.button);  
    b.setOnClickListener(new View.OnClickListener() {
```

@Override

```
    public void onClick(View v) {
```

```
        DialogFragment myFragment = new ButtonDialogFragment();  
        myFragment.show(getFragmentManager(), "FinishDialog");
```

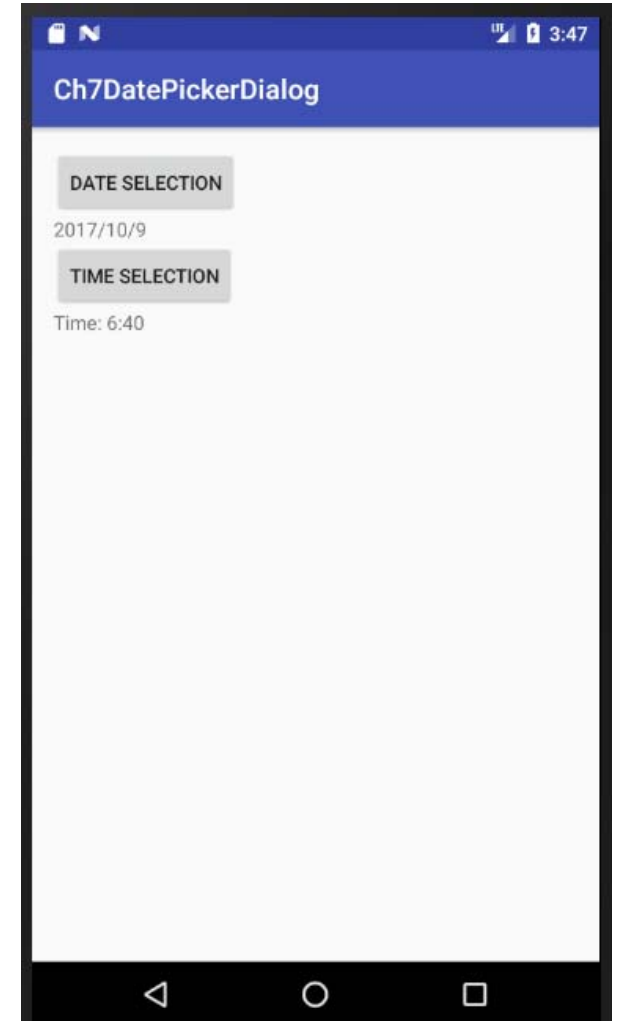
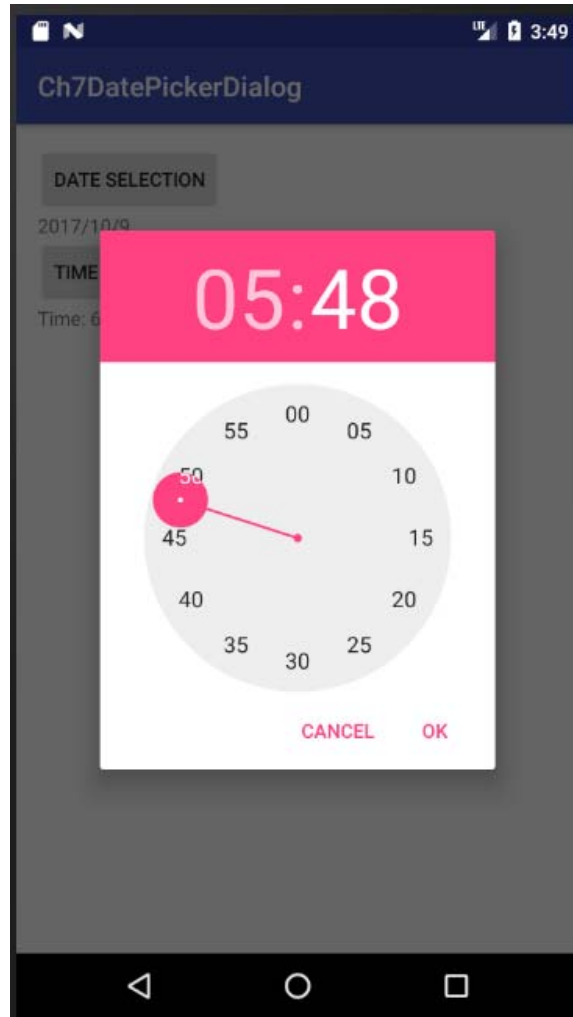
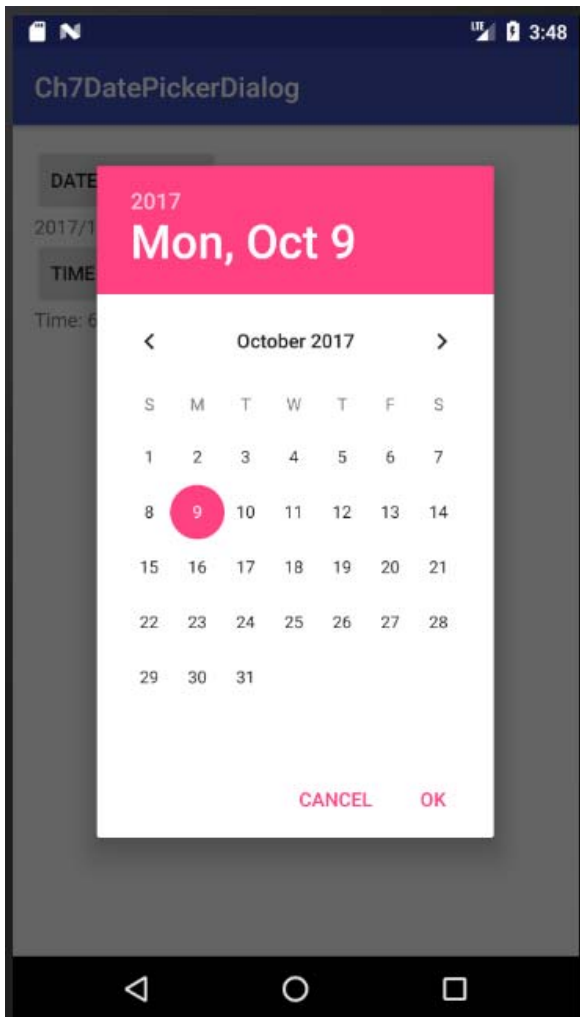
```
        // FragmentManager: public abstract class
```

```
        // android.app.FragmentManager
```

```
        // Interface for interacting with Fragment objects inside of an Activity
```

```
    }  
    };  
}
```

DatePickerDialog / TimePickerDialog



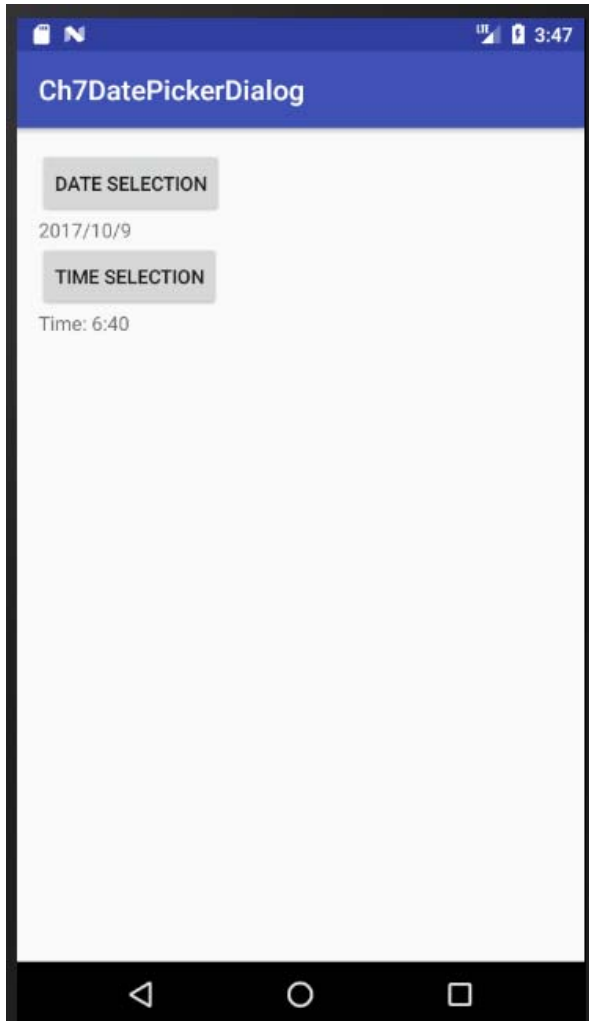
DatePickerDialog / TimePickerDialog 구현

- DialogFragment 이용
- 절차
 - AlertDialog와 비슷하게 DialogFragment를 상속받는 클래스를 정의
 - 1. onCreateDialog 메소드 구현
 - DatePickerDialog / TimePickerDialog 객체를 생성하여 반환
 - 2. 날짜 / 시간이 선택되었을 때 이를 처리하는 이벤트 리스너 구현
 - DatePickerDialog.OnDateSetListener의 onDateSet 메소드 구현
 - TimePickerDialog.OnTimeSetListener의 onTimeSet 메소드 구현

<https://developer.android.com/reference/android/app/TimePickerDialog.html>

<https://developer.android.com/reference/android/app/DatePickerDialog.html>

DatePickerDialog / TimePickerDialog 예제



- 예제 프로젝트: Ch7DatePickerDialog
 - Date selection 버튼을 누르면 DatePickerDialog로 캘린더가 화면에 표시되고 날짜를 선택하면 버튼 아래에 있는 TextView에 날짜 표시
 - Time selection 버튼을 누르면 TimePickerDialog가 화면에 표시되고 시, 분을 선택하면 버튼 아래에 있는 TextView에 시각 표시