● 다이얼로그(dialog): 다양한 상황을 알리기 위해 다이얼로그를 이용한다. 모달(modal)과 모달리스(modeless)로 구분한다. 모달은 다이얼로그를 닫기 전까지 원래의 창(해당 다이얼로그를 호출한 창)을 사용자가 이용할 수 없다. 단면에, 모달리스는 다이얼로그가 화면에 떠 있더라도 사용자가 원래의 창을 계속 이용할 수 있다.

● 토스트 메시지(Toast): 모달리스 형식으로 실행되는 다이얼로그로, 사용자에게 메시지를 알리면서 사용자 행동을 방해하지 않는다. 화면 하단에 검정 바탕의 흰색 글이 잠깐 보이다가 사라지는 다이얼로그다. 시간이 지나면 자동으로 사라지고, 스쳐 지나가는 메시지이므로 사용자 확인이 꼭 있어야 하는 메시지는 일반 다이얼로그로 띄어야 한다. 다이얼로그와 같은 UI는 제공되지 않지만, 액티비티 화면 위에 뛰울 수 있다는 측면에서 다이얼로그의 일종으로 취급한다.

▶ 토스트 생성: 토스트 메시지는 makeText() 함수를 이용해서 생성한다.

```
makeText(Context context, int resId, int duration)
makeText(Context context, CharSequence text, int duration)
```

```
Toast t = Toast.makeText(this, "종료하려면 한 번 더 누르시에ㅛ.", Toast.LENGTH_SHORT):
t.show();
```

두 번째 매개변수가 토스트로 띄울 메시지로, 문자열이나 문자열 리소스를 지정한다. 세 번째 매개변수는 토스트로 보이는 메시지의 유지 시간을 LENGTH_SHORT이나 LENGTH_LONG과 같은 상수로 지정한다. LENGTH_SHORT의 실제 시간은 3초이며, LENGTH_LONG은 5초이다. 이 값을 임의의 숫자 값으로 지정하지는 못한다. 대부분 위처럼 간단히 구현하지만, 원한다면 다음의 함수를 이용하여 추가로 설정할 수도 있다.

setDuration(int duration)	화면에 보이는 시간을 설정한다
setText(int resID)	문자열을 설정한다
setView(View view)	임의의 뷰를 토스트로 띄운다
setGravity(int gravity, int xOffset, int yOffset)	토스트가 뜨는 위치를 조정한다
setMargin(float horizontalMargin, float verticalMargin)	

▶ 콜백: 만약 토스트가 뜨는 순간 혹은 사라지는 순간에 특정 로직을 실행시켜야 한다면, 콜백을 등록해야 한다. 콜백을 등록하기 위해서는 Toast.Callback 객체를 addCallback() 함수로 생성해야 한다. 그러면 토스트가 화면에 보이는 순간에 onToastShown() 함수가, 사라지는 순간에는 onToastHidden() 함수가 자동으로 호출된다.

```
Toast toast = Toast.makeText(this, "toast....", Toast.LENGTH_SHORT);
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.R) {
    toast.addCallback(new Toast.Callback() {
        @Override
        public void onToastShown() {
            super.onToastShown();
            Log.d("kkang", "toast show...");
        }

        @Override
        public void onToastHidden() {
            super.onToastHidden();
            Log.d("kkang", "toast hide...");
        }
    });
}

toast.show();
```

● 알림 창: 시간이 지나면 자동으로 사라져버리는 토스트와는 달리, 알림 창을 띄우면 사용자에게 메시지 확인을 강제할 수 있다. 안드로이드 다이얼로그의 가장 기본은 AlertDialog로, 간단한 메시지 출력뿐 아니라 다양한 화면 구성이 가능해서 복잡한 화면도 띄울 수 있다. 안드로이드 라이브러리에서도 몇몇 다이얼로그를 제공하는데, 이들 모두가 AlertDialog의 서브 클래스이다. 알림 창에는 세 가지 영역이 있다.
ㄴ 타이틀(Title) 영역: 아이콘 이미지와 문자열이 위치한다.
ㄴ 메시지 영역: 다이얼로그의 본문. 문자열을 비롯하여 다양하게 구성할 수 있다.
ㄴ 버튼 영역: 버튼이 위치한다.
▶ 알림 창의 생성: Builder 클래스로 생성한다. 알림 창을 만들어 주는 Builder 클래스의 객체를 생성한 것으로, Builder 객체의 여러 setter 메서드를 이용하여 다이얼로그 구성을 설정하면 Builder 객체가 알림 창을 생성해주는 구조이다.

> 안드로이드에서 몇몇 클래스는 꼭 Builder로 생성하게 강제하고 잇다. 항상 똑같이 생성해야 한다면 new로 하지만, 여러 가지 설정을 어떻게 하느냐에 따라 결과가 달라지는 객체라면 Builder 클래스를 이용해서 생성한다. AlertDialog도 개발자가 어떻게 설정 하느냐에 따라 버튼이나 타이틀이 보일 수 있고, 내용이 다양하게 나올 수 있다. 이처럼 다양한 설정에 따라 객체 생성을 대행하기 위해 Builder을 이용한다. 이런 방식을 디자인 패턴에서는 빌더 패턴 Builder Pattern이라고 한다.

```
AlertDialog.Builder builder = new AlertDialog.Builder(this);
```

setIcon(int iconId)	타이틀 영역의 아이콘 지정
setTitle(CharSequence title)	타이틀 문자열 지정
setMessage(CharSequence message)	본문을 단순 문자열로 구성
setPositiveButton(CharSeuqence text, DialogInterface.OnClickListener listener)	알림 창의 버튼을 추가하는 함수. 최대 3개까지 지정할 수 있고, 같은 성격의 버튼을 두 개 추가할 수는 없다. 두 번째 매개변수를 null로 하면 닫히는데, 닫히는 순간 추가 이벤트 처리가 필요하다면 두 번째 매개변수에 이벤트 핸들러 클래스를 등록해주면 된다. 이 객체는 개발자가 이벤트 처리를 위해 DialogInterface.OnclickListener를 구현한 클래스로, 객체의 onClick() 함수가 이벤트 발생 시 자동으로 호출된다. onClick() 함수의 첫 번째 매개변수가 이벤
setNegativeButton(CharSeuqence text, DialogInterface.OnClickListener listener)	
setNeutralButton(CharSeuqence text, DialogInterface.OnClickListener listener)	

	트가 발생한 다이얼로그 객체이고, 두 번째 매개변수는 버든의 종듀이다.
setCancelable(boolean cancelable)	false로 지정하면 스마트폰의 뒤로가기 버든을 누르거나 알림 창 밖의 화면을 터치했을 때 알림 창이 닫히는 것을 막을 수 있다.
setCanceledOnTouchOutside(boolean cancel)	다이얼로그 창밖을 터치했을 때 다이얼로그가 닫히는지 대한 설정이다. false로 설정하면, 다이얼로그 창밖을 터치했을 때 다이얼로그가 닫히는 것을 막을 수 있다.

Builder에 설정한 내용대로 알림 창을 생성하려면 create() 함수를 이용하고, 생성된 다이얼로그를 show() 함수를 이용하여 화면에 출력한다.

```
DialogInterface.OnClickListener dialogListener = new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        //...
    }
};

AlertDialog.Builder builder = new AlertDialog.Builder(this);

builder.setIcon(android.R.drawable.ic_dialog_alert);
builder.setTitle("알림");
builder.setMessage("정말 종료하시겠습니까?");
builder.setPositiveButton("OK", dialogListener);
builder.setNegativeButton("NO", null);
builder.setCancelable(false);

alertDialog = builder.create();
alertDialog.show();
```

● 목록 다이얼로그: 목록 형식의 다이얼로그도 AlertDialog에서 기본으로 제공해준다. AlertDialog의 본문 영역을 문자열로 지정하지 않고, 목록을 구성하기 위한 문자열 배열을 주면 알아서 목록 다이얼로그 형태로 만들어 준다.

setItems(int itemsId, DialogInterface.OnClickListener listener)	목록을 구성하기 위한 문자열을 배열 리소스로 만들어 사용한다.
setItems(CharSequence[] items, DialogInterface.OnClickListener listener)	코드에서 직접 만든 배열을 매개변수로 지정하여 사용한다.
목록 다이얼로그 내에서는 사용자 항목 선택 이벤트는 두 번째 매개변수로 항목 선택 이벤트 핸들러 객체를 등록하여 구현할 수 있다.	

```
builder.setItems(R.array.dialog_array, dialogListner);

DialogInterface.OnClickListener dialogListener = new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
    // 항목 선택 이벤트가 발생했을 때 호출되는 onClick() 함수의 두 번째 매개변수는
    // 선택한 항목의 인덱스 값이다.
    }
};
```

▶ 각 항목 옆에 체크 박스나 라디오 버튼이 나와는 하는 경우에 사용하는 함수: setMultiChoiceItems

setMultiChoiceItems(CharSequence[] items, boolean[] checkedItems, DialogInterface.OnMultiChoiceClickListener listener)	첫 번째 매개 변수로 항목 구성을 위한 배열 값을 지정한다.
setMultiChoiceItems(int itemsId, boolean[] checkedItems, DialogInterface.OnMultiChoiceClickListener listener)	첫 번째 매개 변수로 항목 구성을 위한 배열 리소스를 지정한다.
두 번째 매개변수는 선택되어 있어야 할 체크박스의 체크 상태 값이다. 이 값을 주어서 체크 박스가 선택된 상태로 다이얼로그를 띄울 수 있다. 세 번째 매개변수는 항목 선택 이벤트 핸들러이다.	

▶ 항목 옆에 라디오 버튼이 나오도록 하는 함수: setSingleChoiceItems()

setSingleChoiceItems(CharSequence[] items, int checkedItem, DialogInterface.OnclickListener listener)	배열 정보/배열 리소스로 라디오 버튼이 나오는 목록 다이얼로그를 띄우는 코드이다. 두 번째 매개변수는 초기 선택 항목의 인덱스 값이다. 예를 들어 0으로 지정할 경우, 처음 다이얼로그가 열릴 때 첫 번째 라디오 버튼이 체크된 상태로 열린다.
setSingleChoiceItems(CharSequence[] items, int checkedItem, DialogInterface.OnclickListener listener)	

● 날짜 선택 다이얼로그: 사용자가 다이얼로그 창에서 날짜를 선택할 수 있도록 하는 DatePickerDialog.

```
DatePickerDialog dateDialog = new DatePickerDialog(this, null, year, month, day);
```

두 번째 매개변수는 사용자가 날짜를 선택할 때 발생하는 이벤트 처리 핸들러이며, 이후 숫자 3개가 다이얼로그에서 기본으로 보여야 할 날짜의 연, 월, 일이다.

```
DatePickerDialog dateDialog = new DatePickerDialog(this, new DatePickerDialog.OnDateSetL
    @Override
    public void onDateSet(DatePicker view, int year, int monthOfYear, int dayOfMonth) {
    }
}, year, month, day);
```

이벤트가 발생하면 onDateSet() 함수가 호출되어 매개변수로 사용자가 선택한 연, 월, 일을 받을 수 있다.

● 시간 선택 다이얼로그: 사용자가 시간을 선택할 수 있도록 하는 TimePickerDialog.

```
TimePickerDialog timeDialog = new TimePickerDialog(this, null, hour, minute, true);
```

매개변수로 시, 분을 입력하면 초기 시간이 화면에 나온다. 세 번째 매개 변수는 24시간 체계로 나타낼지 설정하는 논리값으로 true로 지정하면 24시간 체계로 나오고, false로 지정하면 12시간 체계로 오전, 오후를 선택할 수 있는 버든이 나온다. 시간 조정 이벤트는 생성자의 매개변수에 이벤트 핸들러를 등록하여 처리한다.

```
TimePickerDialog timeDialog = new TimePickerDialog(this, new TimePickerDialog.onTimeSetL
    @Override
    public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
    }
}, hour, minute, false);
```

● 커스텀 다이얼로그: 개발자가 원하는 임의의 화면을, 개발자가 직접 구성하는 다이얼로그를 AlertDialog를 이용하여 만들 수 있다. 우선 다이얼로그 화면을 구성하는 레이아웃 XML파일을 만든다. 커스텀 다이얼로그는 AlaertDialog과 구현 방법은 같지만, 개발자가 직접 구성한 뷰를 setView()함수를 통해 다이얼로그 본문으로 지정하는 과정이 필요하다.

```
// LayoutInflator 클래스를 이용하여 레이아웃 xml 파일을 초기화한다.
LayoutInflator inflator = (LayoutInflator)getSystem.Service(LAYOUT_INFLATOR_SERVICE);
View view = inflator.inflate(R.layout.dialog_custom, null);
// 초기화된 뷰를 뷰를 Builder의 setView() 함수를 이용해 AlertDialog의 본문에 지정하면 된다.
builder.setView(view);
```

```
// dialog_layout.xml

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/text1"
        android:text="USB 디버깅을 허용하시겠습니까?"
        android:textStyle="bold"
        android:textSize="15dp"
        android:layout_marginLeft="32dp"
        android:layout_marginTop="32dp"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/text2"
        android:text="컴퓨터 RSA키 지문 : "
        android:layout_below="@id/text1"
        android:layout_alignLeft="@id/text1"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/text3"
        android:text="17:AA:BB:77:88:DD:98:7E"
        android:layout_below="@id/text2"
        android:layout_alignLeft="@id/text2"/>
    <CheckBox
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="이 컴퓨터에서 항상 허용"
        android:layout_below="@id/text3"
        android:layout_alignLeft="@id/text3"/>
</RelativeLayout>
```

```
// arrays.xml (values - strings)
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="dialog_array">
        <item>기본 알람 소리</item>
        <item>Argon</item>
        <item>Awaken</item>
        <item>Bounce</item>
        <item>Carbon</item>
    </string-array>
</resources>
```

```xml
// activity_main.xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/alertButton"
        android:text="alert dialog"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/listButton"
        android:text="list dialog"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/dateButton"
        android:text="date dialog"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/timeButton"
        android:text="time dialog"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/customButton"
        android:text="custom dialog"/>
</LinearLayout>
```

```java
// MainActivity.java

package com.android.practicedialog;

import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

import android.app.DatePickerDialog;
import android.app.TimePickerDialog;
import android.content.DialogInterface;
import android.os.Build;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.DatePicker;
import android.widget.TimePicker;
import android.widget.Toast;

import com.android.practicedialog.databinding.ActivityMainBinding;

import java.util.Calendar;

public class MainActivity extends AppCompatActivity {
```

```
ActivityMainBinding binding;

// 이벤트 처리를 위해 다이얼로그 객체를 멤버변수로 선언
AlertDialog customDialog;
AlertDialog listDialog;
AlertDialog alertDialog;

// 매개 변수의 문자열을 Toast로 띄우는 개발자 함수
private void showToast(String message) {
    Toast toast = Toast.makeText(this, message, Toast.LENGTH_SHORT);
    toast.show();
}

// Dialog 버튼 이벤트 처리
DialogInterface.OnClickListener dialogListener = new DialogInterface.OnClickListener
    @Override
    public void onClick(DialogInterface dialog, int which) {
        if (dialog == customDialog && which == DialogInterface.BUTTON_POSITIVE) {
            showToast("custom dialog 확인 click");
        } else if (dialog == listDialog) {
            // 목록 다이얼로그의 항목이 선택되었을 때, 항목 문자열 획득
            String[] datas = getResources().getStringArray(R.array.dialog_array);
            showToast(datas[which] + " 선택 하셨습니다.");
        } else if (dialog == alertDialog && which == DialogInterface.BUTTON_POSITIVE
            showToast("alert dialog ok click");
        }
    }
};

@RequiresApi(api = Build.VERSION_CODES.N)
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    binding = ActivityMainBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());

    // alartButton
    binding.alertButton.setOnClickListener(view -> {
        //AlertDialog 띄우기
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setIcon(android.R.drawable.ic_dialog_alert);
        builder.setTitle("알림");
        builder.setMessage("정말 종료하시겠습니까?");
        builder.setPositiveButton("OK", dialogListener);
        builder.setNegativeButton("NO", null);
        builder.setCancelable(false);
        alertDialog = builder.create();
        alertDialog.show();
    });

    // listButton
    binding.listButton.setOnClickListener(view -> {
        // 목록 다이얼로그 띄우기
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("알람 벨소리");
        builder.setSingleChoiceItems(R.array.dialog_array, 0, dialogListener);
        builder.setPositiveButton("확인", null);
        builder.setNegativeButton("취소", null);
        builder.setCancelable(false);
        listDialog = builder.create();
```

```
            listDialog.show();
        });

        // dateButton
        binding.dateButton.setOnClickListener(view -> {
            // 현재 날짜로 다이얼로그를 띄우기 위해 날짜를 구함
            Calendar c = Calendar.getInstance();
            int year = c.get(Calendar.YEAR);
            int month = c.get(Calendar.MONTH);
            int day = c.get(Calendar.DAY_OF_MONTH);

            // 두 번째 매개변수는 날짜 선택 시 발생하는 이벤트, 세 번째 매개변수는 다이얼로그에
            DatePickerDialog dateDialog = new DatePickerDialog(this,
//                    new DatePickerDialog.OnDateSetListener() {
//                        @Override
//                        public void onDateSet(DatePicker view, int year, int month, ir
//                            showToast(year + ":" + (month+1) + ":" + dayOfMonth);
//                        }
//                    }, year, month, day);
                    (view1, year1, month1, dayOfMonth) -> showToast(year1 + "년 " + (mon

            dateDialog.show();
        });

        // timeButton
        binding.timeButton.setOnClickListener(view -> {
            // 현재 시간으로 다이얼로그를 띄우기 위해 시간을 구함
            Calendar c = Calendar.getInstance();
            int hour = c.get(Calendar.HOUR_OF_DAY);
            int minute = c.get(Calendar.MINUTE);
            TimePickerDialog timeDialog = new TimePickerDialog(this,
//                    new TimePickerDialog.OnTimeSetListener() {
//                        @Override
//                        public void onTimeSet(TimePicker view, int hourOfDay, int minu
//                            showToast(hourOfDay + ":" + minute);
//                        }
//                    }, hour, minute, false);
                    (view12, hourOfDay, minute1) -> showToast(hourOfDay + " 시 " + minut

            timeDialog.show();
```

Resources: 깡쌤의 안드로이드 프로그래밍