Anda sudah lulus kelas ini

DISKUSIKAN MATERI INI

Daftar Modul

Search

WebView : Latihan - WebView Sederhana

WebView : Quiz

SoundPool dan MediaPlayer : Teori

SoundPool dan

SoundPool dan Mediaplayer : Latihan - Aplikasi MediaPlayer sederhana

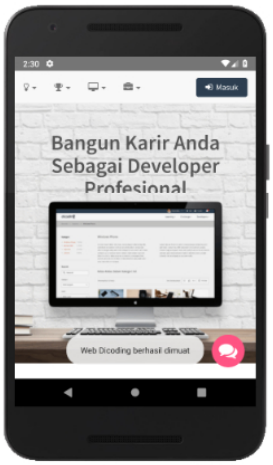
SoundPool dan Mediaplayer : Latihan -

Pembaharuan! Modul ini dibuat pada tanggal 6 May 2019. Pembaharuan terakhir adalah: **Migrasi ke Kotlin**. *Lihat riwayat »*

Tujuan

Pada Codelab kali ini Anda akan mempelajari bagaimana mengimplementasikan WebView dalam membuat aplikasi Android dengan menampilkan sebuah web.

Hasil dari codelab kali ini akan menjadi seperti ini:



Logika Dasar

Menjalankan Aplikasi → Memuat sebuah alamat web → Menampilkan halaman web di WebView.

Codelab WebView

1. Buat Project baru di Android Studio dengan kriteria sebagai berikut:

Nama Project	**MyWebView**
Target & Minimum Target SDK	**Phone and Tablet, Api level 21**
Tipe Activity	**Empty Activity**
Activity Name	**MainActivity**
Use AndroidX Artifacts	**True**
Language	**Kotlin/Java**

2. Setelah itu, bukalah ***activity_main.xml*** dan tambahkan sebuah WebView di dalamnya. Kode pada layout tersebut:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/a
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <WebView
        android:id="@+id/webView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
```

3. Setelah itu bukalah ***MainActivity***, tambahkan kode berikut ini:

Kotlin **Java**

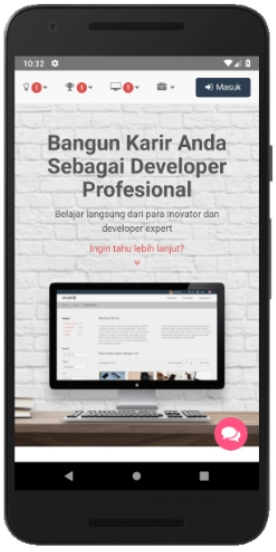
```java
public class MainActivity extends AppCompatActivity {

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

        WebView myWebView = findViewById(R.id.webView);
        myWebView.loadUrl("https://www.dicoding.com");
    }
}
```

4. Kemudian jalankan aplikasi Anda, maka tampilannya akan menjadi seperti ini:

5. Jika dilihat pada gambar di atas, terjadi eror berupa tidak bisa dimuat. Ini terjadi karena Anda belum memberikan *permission* di bagian manifest. Bukalah ***AndroidManifest.xml***, tambahkan *permission* berikut ini dan letakkan di atas `<application ...>`:

```xml
<uses-permission android:name="android.permission.INTERNET" />
```

6. Setelah itu jalankan kembali aplikasi yang Anda buat.



7. Jika Anda perhatikan, ada beberapa kode pada halaman WebView yang tidak bekerja seperti JavaScript-nya. Tambahkan kode berikut untuk mengaktifkan JavaScript:
Kemudian jalankan kembali dan klik tombol pesan yang ada pada kanan bawah, maka hasilnya akan menjadi seperti ini:

Kotlin **Java**

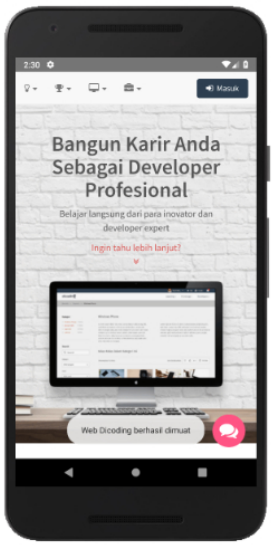
```
myWebView.getSettings().setJavaScriptEnabled(true);
```

8. Kemudian tambahkan kode berikut untuk melakukan konfigurasi pada `WebViewClient`.

Kotlin **Java**

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

    WebView myWebView = findViewById(R.id.webView);
    myWebView.getSettings().setJavaScriptEnabled(true);

    myWebView.setWebViewClient(new WebViewClient() {
        @Override
        public void onPageFinished(WebView view, String url) {
            Toast.makeText(MainActivity.this, "Web Dicoding berhasil dimuat", Toast.LENGTH_LO
        }
    });

    myWebView.loadUrl("https://www.dicoding.com");
}
```

9. Jalankan kembali aplikasi Anda, maka akan muncul pesan setelah web berhasil dimuat.



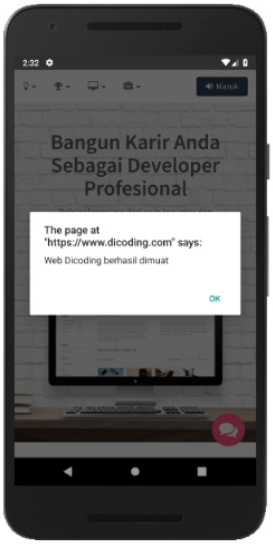
10. Selanjutnya, tambahkan kode berikut untuk melakukan konfigurasi pada `WebChromeClient`.

Kotlin **Java**

```
public class MainActivity extends AppCompatActivity {

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

        WebView myWebView = findViewById(R.id.webView);
        myWebView.getSettings().setJavaScriptEnabled(true);

        myWebView.setWebViewClient(new WebViewClient() {
            @Override
            public void onPageFinished(WebView view, String url) {
                view.loadUrl("javascript:alert('Web Dicoding berhasil dimuat')");
            }
        });
```

```
17.
18.         myWebView.setWebChromeClient(new WebChromeClient());
19.
20.         myWebView.loadUrl("https://www.dicoding.com");
21.     }
22. }
```

11. Jalankan kembali, maka akan menjadi seperti ini:

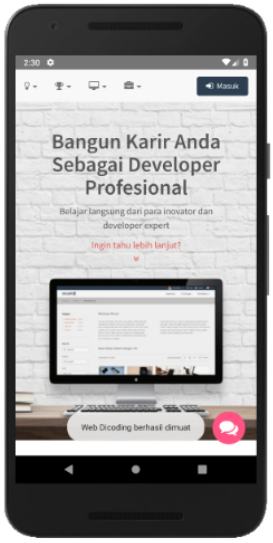


12. Ubahlah kode pada `WebChomeClient` berikut ini:

Kotlin **Java**

```
6.          setContentView(R.layout.activity_main);
7.
8.          WebView myWebView = findViewById(R.id.webView);
9.
10.         myWebView.getSettings().setJavaScriptEnabled(true);
11.
12.         myWebView.setWebViewClient(new WebViewClient() {
13.             @Override
14.             public void onPageFinished(WebView view, String url) {
15.                 view.loadUrl("javascript:alert('Web Dicoding berhasil dimuat')");
16.             }
17.         });
18.
19.         myWebView.setWebChromeClient(new WebChromeClient() {
20.             @Override
21.             public boolean onJsAlert(WebView view, String url, String message, final androi
22.                 Toast.makeText(MainActivity.this, message, Toast.LENGTH_LONG).show();
23.                 result.confirm();
24.                 return true;
25.             }
26.         });
27.
28.         myWebView.loadUrl("https://www.dicoding.com");
29.     }
30. }
```

13. Jalankan kembali aplikasi Anda, maka akan muncul pesan setelah web berhasil dimuat.



Bedah Kode

WebView

Perhatikan kode berikut ini:

```
<WebView
      android:id="@+id/webView"
      android:layout_width="0dp"
      android:layout_height="0dp" />
```

Kode di atas digunakan untuk menampilkan widget WebView di dalam layout. Attribut yang paling penting dalam pemanggilan WebView adalah `id`, `layout_width` dan `layout_height`.

Kotlin **Java**

```
myWebView.loadUrl("https://www.dicoding.com");
```

Kode di atas digunakan untuk memuat sebuah url dari website. Perlu Anda ketahui bahwa untuk OS Nougat ke atas, sebuah url harus menggunakan *https*. Jika Anda tidak menggunakan *https* atau hanya *http* saja, maka alamat tersebut tidak akan dimuat.

```
<uses-permission android:name="android.permission.INTERNET" />
```

Tentu, *permission* pasti dibutuhkan untuk mengakses internet. Jika tidak ada permission di atas, maka aplikasi tidak akan mempunyai akses internet.

Kotlin **Java**

```
myWebView.getSettings().setJavaScriptEnabled(true);
```

Kode di atas digunakan untuk *mengaktifkan JavaScript* pada sebuah website. *Default* pengaturan JavaScript dari sebuah WebView adalah *false*, jadi Anda perlu mengubahnya menjadi *true* jika ingin menampilkan JavaScript pada website yang dimuat.

Kotlin **Java**

```
myWebView.setWebViewClient(new WebViewClient() {
    @Override
    public void onPageFinished(WebView view, String url) {
        view.loadUrl("javascript:alert('Web Dicoding berhasil dimuat')");
    }
});
```

`WebViewClient` merupakan *client* yang ada pada WebView. Anda bisa menerapkan tindakan seperti persiapan ketika webview dimuat, atau ketika webview tersebut selesai memuat sebuah halaman. Contoh pada kode di atas menampilkan sebuah pesan "***Web Dicoding berhasil dimuat***" melalui JavaScript. Namun perlu Anda ketahui, untuk menampilkan sebuah *alert* dibutuhkan `WebChromeClient`.

Kotlin **Java**

```
myWebView.setWebChromeClient(new WebChromeClient() {
   @Override
   public boolean onJsAlert(WebView view, String url, String message, final android.webkit.JsResult
       Toast.makeText(MainActivity.this, message, Toast.LENGTH_LONG).show();
       result.confirm();
       return true;
   }
});
```

`WebChromeClient` merupakan *client* tambahan yang ada pada WebView dan mempunyai fungsi untuk menampilkan *loading*, menampilkan alert atau perintah-perintah JavaScript lainnya. Pada contoh di atas, kode tersebut akan menampilkan sebuah Toast dengan pesan sesuai dengan yang ada pada alert di JavaScript.

Menarik bukan? Sebenarnya, Anda bisa membuat WebView sekreatif mungkin sesuai kebutuhan Anda. Untuk memahami materi WebView lebih dalam, Anda bisa melihat tautan berikut sebagai

referensi:

- WebView Started

Anda bisa unduh proyek di atas pada tautan berikut:

- MyWebView

← KEMBALI KE MATERI SEBELUMNYA

LANJUTKAN KE MATERI BERIKUTNYA →