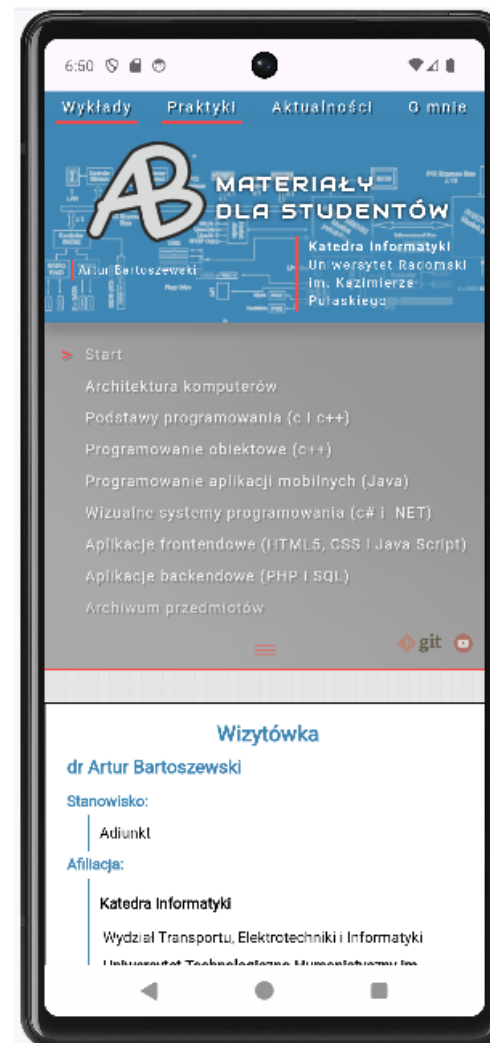


PROGRAMOWANIE APLIKACJI MOBILNYCH

**Aplikacje zdalne - Wyświetlanie stron www
bezpośrednio w aplikacji**

dr Artur Bartoszewski

Aplikacje internetowe - Wyświetlanie stron www bezpośrednio w aplikacji



Aplikacje działające na systemie Android podzielić można na 3 grupy:

1. **Aplikacje natywne** - tworzone z wykorzystaniem SDK systemu oraz języków Java lub Kotlin. Przechowywane są na urządzeniu (plik .apk)
2. **Aplikacje internetowe** - realizowane jako aplikacje zdalne, działające na serwerze. Często są to po prostu odpowiednio przygotowane (responsywne) strony WWW.
3. **Aplikacje hybrydowe** - aplikacje internetowe, jednak oprócz wykorzystania zdalnych zasobów możliwa jest interakcja z lokalnymi zasobami smartfona (np. dostęp do czujników)

Komponent WebView - wstępnie zainstalowany na urządzeniu składnik systemu bazujący na Chrome. Umożliwia on aplikacjom na Androida wyświetlanie treści internetowych

Komponent WebView nie posiada interfejsu użytkownika przeglądarki – tylko renderuje stronę, bez standardowych menu. przycisków itp..

Klasa WebView zawiera metody:

- Ładowanie strony,
- nawigacja wprzód/wstecz przez historię przeglądarki,
- Przybliżanie i oddalanie,
- Wykorzystanie stylów CSS oraz skryptów JavaScript,
- Wymiana danych z urządzeniem przez interfejs JavaScript,
- Wyszukiwanie tekstu, pobieranie zdjęć,

Dodawanie obiektu WebView

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

Komponent WebView rozszerza klasę View, co oznacza, że może być dodany do interfejsu użytkownika (UI) jak każdy inny widok w Androidzie (np. TextView, Button).

Obsługa obiektu WebView

Aby wyświetlić stronę internetową w WebView, wystarczy załadować odpowiedni URL za pomocą metody `.loadURL(http://...)`

```
public class MainActivity extends AppCompatActivity {  
  
    private WebView webView;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        webView = findViewById(R.id.webView);  
        webView.loadUrl("https://bartoszewskia.github.io/uthrad/");  
    }  
}
```

Nie działa?
Patrz następny slajd

Uprawnienia

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.webvirwtest">

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="WebVirwTest"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

W pliku AndroidManifest dodajemy uprawnienia do korzystania z Internetu

Obsługa nawigacji i linków

Domyślnie po kliknięciu na link w WebView, Android otworzy przeglądarkę. Aby zmienić to zachowanie i obsługiwać nawigację wewnątrz WebView, należy nadpisać metodę `shouldOverrideUrlLoading()` w klasie `WebViewClient`.

```
webView.setWebViewClient(new WebViewClient() {  
    @Override  
    public boolean shouldOverrideUrlLoading(WebView view, String url) {  
        view.loadUrl(url);  
        return true;  
    }  
});
```

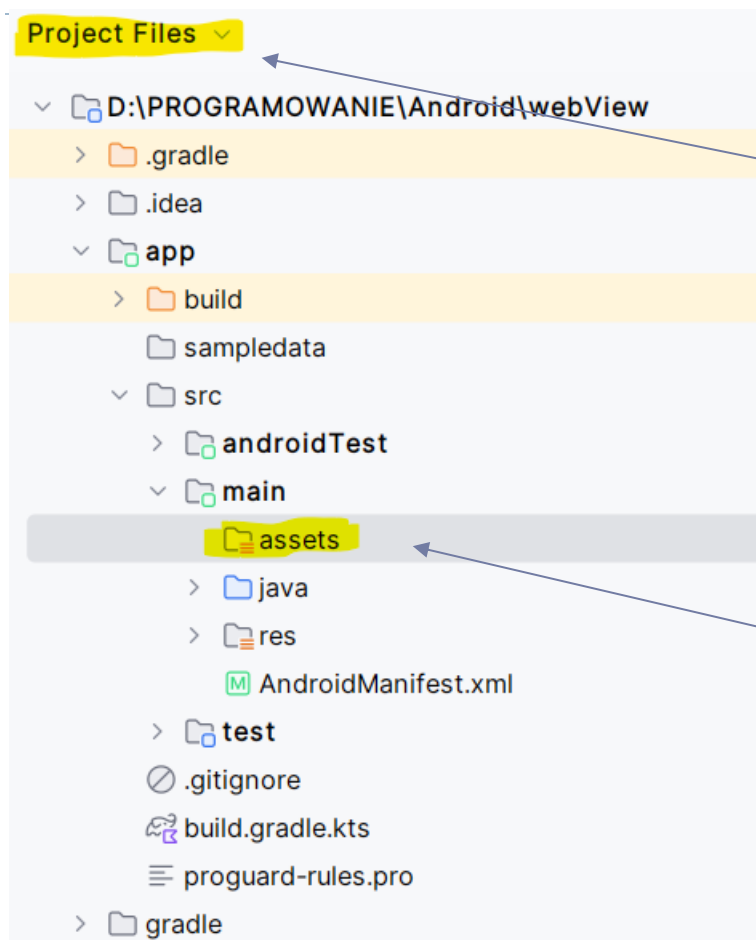

Obsługa lokalnych plików HTML

WebView może również ładować lokalne pliki HTML, które są przechowywane w katalogu **assets**

Folder assets powinien znajdować się na tym samym poziomie co katalogi java i res

Aby załadować plik, należy użyć ścieżki:

```
webView.loadUrl("file:///android_asset/lokalny_plik.html");
```



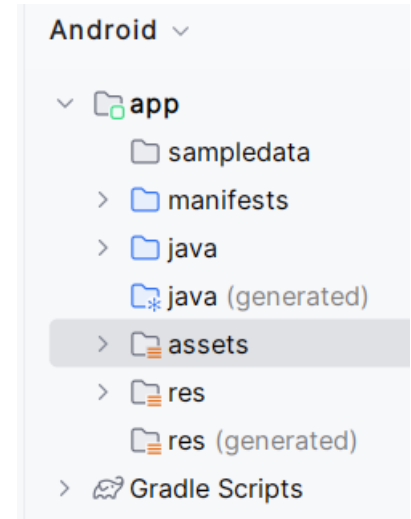
Przełącz widok na Project Files

Utwórz folder assets w tym miejscu

Obsługa lokalnych plików HTML

Katalog assets w projekcie Androida jest miejscem, w którym można przechowywać różnego rodzaju zasoby, takie jak pliki HTML, CSS, JavaScript, obrazy czy inne pliki, które nie są kompilowane przez Androida, ale mogą być potrzebne w aplikacji.

Pliki umieszczone w katalogu assets są wczytywane bezpośrednio w takiej formie, w jakiej zostały dodane.



Domyślnie WebView ma wyłączoną obsługę JavaScript. Aby ją włączyć, należy ustawić odpowiednie opcje w WebSettings.

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

Tworzymy klasę, którą będziemy mogli wywołać z poziomu JavaScript

```
package com.example.webview;
```

```
public class WebAppInterface {  
    Context mContext;
```

```
    WebAppInterface(Context c) {  
        mContext = c;  
    }  
}
```

```
@JavascriptInterface
```

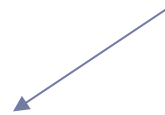
```
public void showToast(String toast) {  
    Toast.makeText(mContext, toast, Toast.LENGTH_SHORT).show();  
}  
}
```

W klasie tworzymy metody, które będzie można wywołać (w naszym przykładzie metoda wyświetlająca okienko komunikatu z Androida)

Metody mogą przyjmować parametry

WebView umożliwia interakcję pomiędzy kodem Java a JavaScript za pomocą metody `addJavascriptInterface()`.

Parametrem metody jest obiekt utworzonej właśnie klasy



```
webView.addJavascriptInterface(new WebAppInterface(this), "AndroidInterface");
```

W HTML-u wywołanie metody utworzonej przez nas klasy możemy podpiąć np. pod zdarzenie kliknięcia przycisku

```
<button type="button" onClick='AndroidInterface.showToast("To ja JavaScript!")'>  
    Wyświetl komunikat  
</button>
```

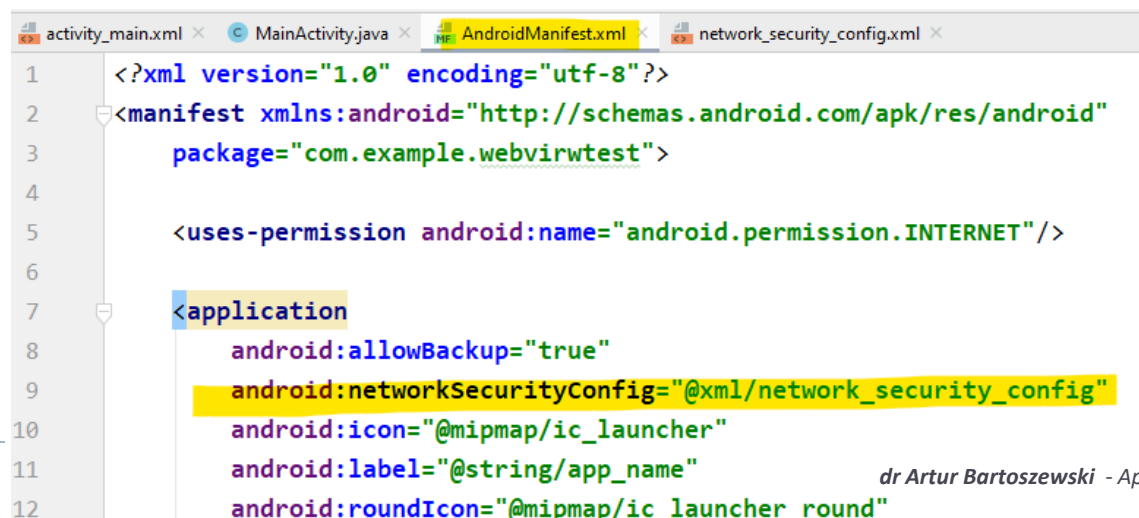
Problem z „brakiem s” w http://..

Próba wczytania do WebView strony o niezabezpieczonym protokole http:// zakończy się niepowodzeniem.

Jednym z rozwiązań jest dodanie domeny do listy wyjątków.



```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
  <domain-config cleartextTrafficPermitted="true">
    <domain includeSubdomains="true">uniwersytetradom.pl</domain>
  </domain-config>
</network-security-config>
```



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.webvirwtest">

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

  <application
    android:allowBackup="true"
    android:networkSecurityConfig="@xml/network_security_config"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
```

