Chapter 10

서버에 데이터 요청하고 응답받기

이번 장에서는 무엇을 다룰까요?

서버나 다른 단말에 연결하는 네트워킹에 대해 알고 싶어요.

- 네트워킹이 어떤 것인지 살펴볼까요?
- 소켓으로 간단하게 연결하는 방법에 대해 알아볼까요?
- 웹으로 연결하는 방법에 대해 알아볼까요?
- 라이브러리를 이용해 서버의 데이터를 간단하게 가져오는 방법에 대해 알아볼까요?
- 영화 정보를 가져와서 보여줘 볼까요?

이번 장에서는 무엇을 다룰까요?

네트워킹에 대해 어떻게 생각해야 하나요?
- 네트워킹이란?

▼

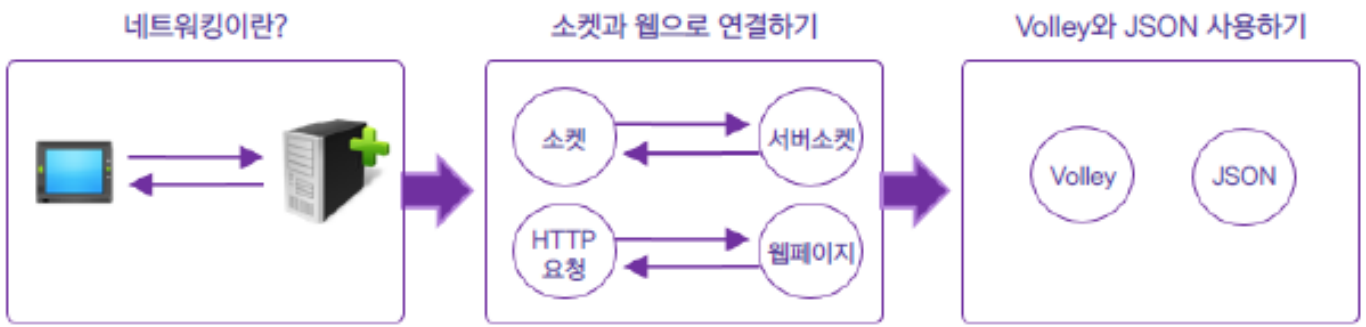
소켓이나 웹 연결은 어떻게 하나요?
- 소켓 사용하기
- 웹으로 요청하기

▼

라이브러리를 이용해 간단하게 가져오는 방법도 있나요?
- Volley 사용하기
- JSON 데이터 다루기

▼

영화 정보를 가져와 보여주고 싶어요
- 영화 정보 가져와 보여주기



강의 주제 및 목차

강의 주제

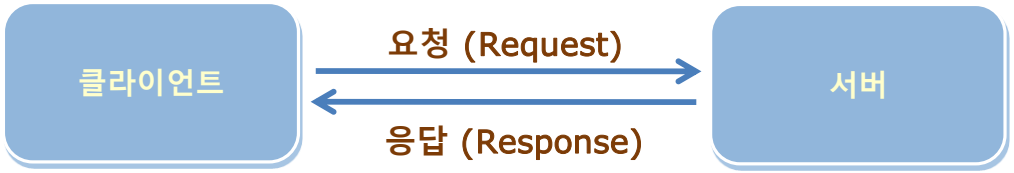
기본적인 네트워킹 방법에 대한 이해

1. 네트워킹이란?

네트워킹이란?

■ 원격지의 서버를 연결하는 방식

- 2-tier C/S 모델
  - 클라이언트와 서버가 일대일로 연결하는 방식



[2-tier C/S 모델]

- 3-tier 모델
  - 서버를 좀 더 유연하게 구성
  - 응용 서버와 데이터 서버로 구성하는 경우, 데이터베이스를 분리시킴

[3-tier 모델]

소켓 사용하기

■ 네트워킹에 대한 이해

- TCP/IP 수준의 통신 방식을 제공하는 소켓을 이용해 서버에 연결해 보면 이해하기 쉬움
- 일반적인 프로그래밍에서는 대부분 TCP 연결 사용
- 비연결성(stateless) 특성으로 인해 실시간으로 데이터를 처리하는 애플리케이션의 경우, 응답 속도를 높이기 위해 HTTP보다 소켓 연결 선호

■ 소켓 연결 방식

- 안드로이드에서는 표준 자바의 소켓을 그대로 사용할 수 있음
- 서버쪽에는 서버소켓을 만들어 실행함 (포트 지정)
- 클라이언트쪽에서는 소켓을 만들어 서버소켓으로 연결함 (IP와 포트 지정)
- Stream 객체를 이용해 데이터를 보내거나 받을 수 있음

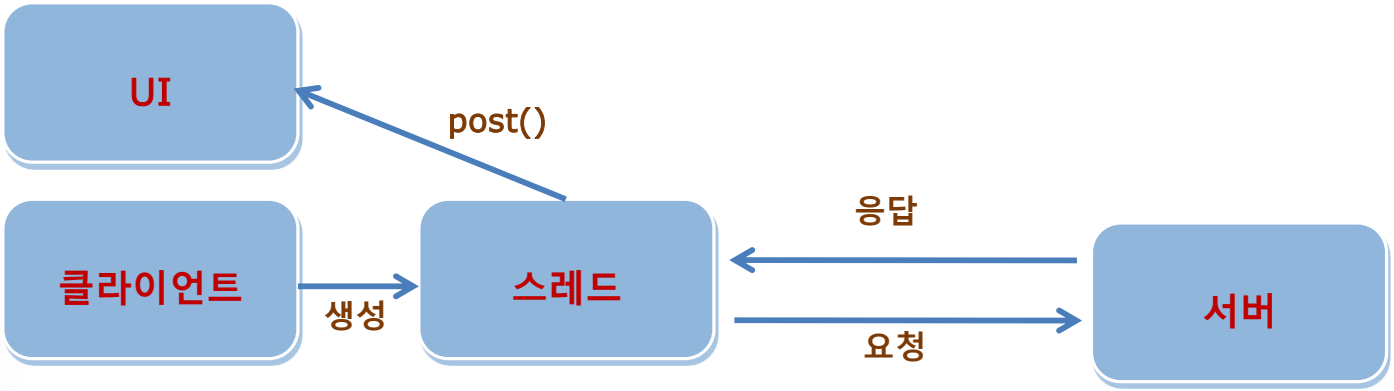
네트워킹 사용 시 주의할 점

■ 네트워킹을 사용할 때는 **반드시 스레드** 사용

- 최신 버전의 안드로이드에서는 네트워킹을 사용할 때는 반드시 스레드를 사용하도록 변경되었음 (이전에는 스레드 없이도 가능했음)

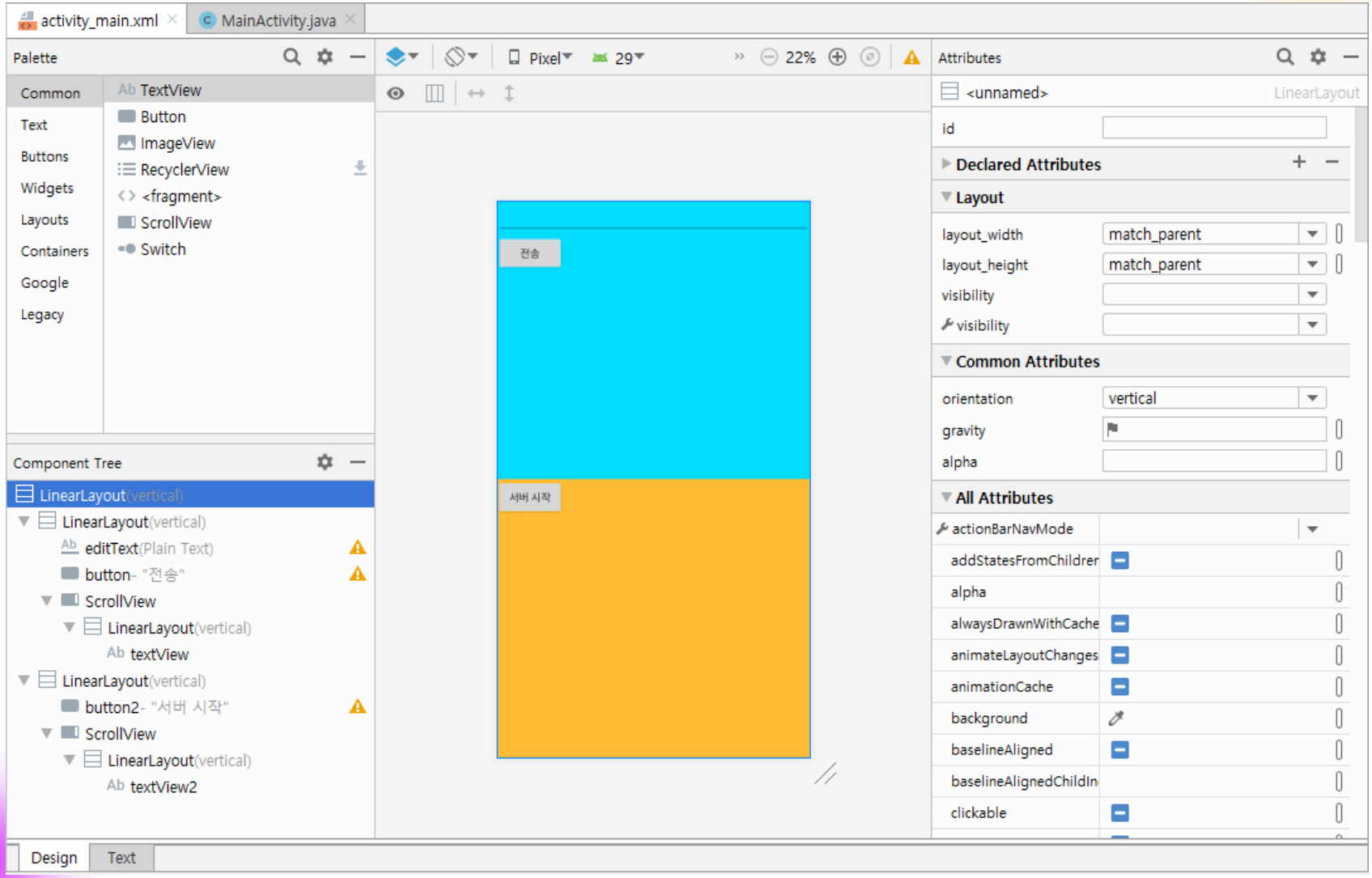
■ 스레드를 사용하므로 UI 업데이트를 위해서는 **반드시 핸들러** 사용

- 네트워킹을 위해 새로 만든 스레드 안에서 그 결과를 보여주기 위해 UI 업데이트를 하는 경우 스레드 부분에서 공부한 바와 같이 핸들러를 사용해야 함
- 가장 간단한 방법으로 post() 메소드 사용 권장



2. 소켓 사용하기

화면 레이아웃 만들기


activity_main.xml
MainActivity.java
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Google
Legacy
TextView
Button
ImageView
RecyclerView
<fragment>
ScrollView
Switch
Component Tree
LinearLayout(vertical)
editText(Plain Text)
button- "전송"
textView
button2- "서버 시작"
textView2
Pixel
29
22%
전송
서버 시작
Attributes
<unnamed>
LinearLayout
id
Declared Attributes
Layout
layout_width
match_parent
layout_height
match_parent
visibility
Common Attributes
orientation
vertical
gravity
alpha
All Attributes
actionBarNavMode
addStatesFromChildren
alpha
alwaysDrawnWithCache
animateLayoutChanges
animationCache
background
baselineAligned
clickable
Design
Text

버튼 누르면 스레드 안에서 send

```
@Override
public void onClick(View v) {
  final String data = editText.getText().toString();
  new Thread(new Runnable() {
    @Override
    public void run() {
      send(data);
    }
  }).start();
}
```

❶ 스레드 안에서 send() 메서드 호출하기

버튼 누르면 스레드 안에서 startServer

```
@Override
public void onClick(View v) {
  new Thread(new Runnable() {
    @Override
    public void run() {
      startServer();
    }
  }).start();
}
```

❷ 스레드 안에서 startServer() 메서드 호출하기

send 메서드 정의

```
public void send(String data) {
  try {
    int portNumber = 5001;
    Socket sock = new Socket("localhost", portNumber);
    printClientLog("소켓 연결함.");

    ObjectOutputStream outstream = new ObjectOutputStream(sock.getOutputStream());
    outstream.writeObject(data);
    outstream.flush();
    printClientLog("데이터 전송함.");

    ObjectInputStream instream = new ObjectInputStream(sock.getInputStream());
    printClientLog("서버로부터 받음: " + instream.readObject());
    sock.close();
  } catch (Exception ex) {
    ex.printStackTrace();
  }
}
```

❶ 소켓 객체 만들기

❷ 소켓 객체로 데이터 보내기

startServer 메서드 정의

```
public void startServer() {
  try {
    int portNumber = 5001;

    ServerSocket server = new ServerSocket(portNumber);
    printServerLog("서버 시작함: " + portNumber);
```

❶ 소켓 서버 객체 만들기

```
    while (true) {
      Socket sock = server.accept();
      InetAddress clientHost = sock.getLocalAddress();
      int clientPort = sock.getPort();
      printServerLog("클라이언트 연결됨: " + clientHost + " : " + clientPort);

      ObjectInputStream instream = new ObjectInputStream(sock.getInputStream());
      Object obj = instream.readObject();
      printServerLog("데이터 받음: " + obj);

      ObjectOutputStream outstream = new ObjectOutputStream(sock.getOutputStream());
      outstream.writeObject(obj + " from Server.");
      outstream.flush();
      printServerLog("데이터 보냄.");

      sock.close();
    }
  } catch (Exception ex) {
    ex.printStackTrace();
  }
}
```

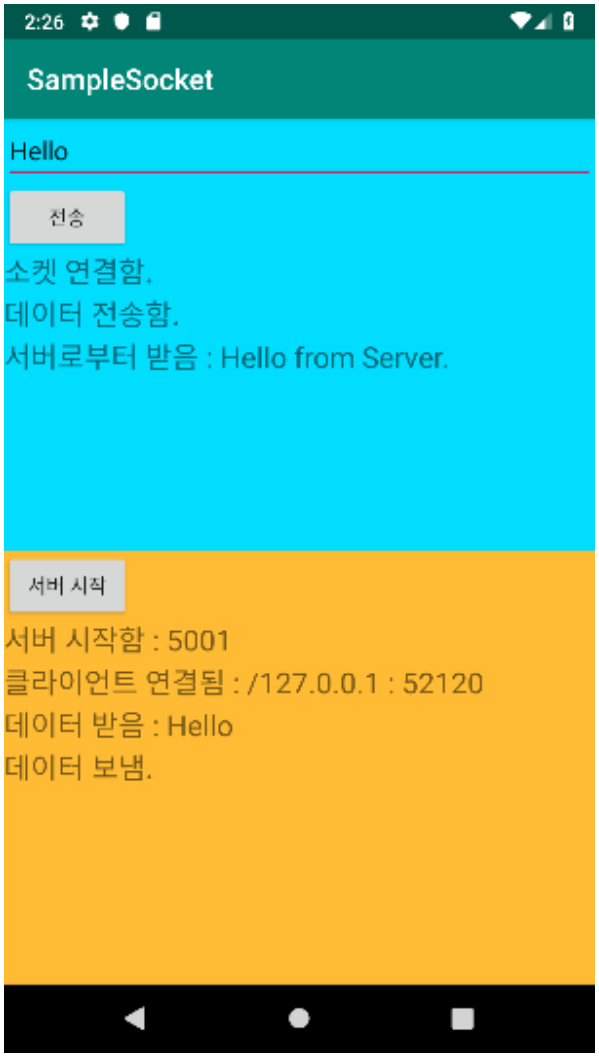
❷ 클라이언트가 접속했을 때 만들어지는 소켓 객체 참조하기

INTERNET 권한 추가

참조파일 SampleSocket>/app/manifests/AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="org.techtown.socket">

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

중략…
```

앱 실행


2:26
SampleSocket
Hello
전송
소켓 연결함.
데이터 전송함.
서버로부터 받음 : Hello from Server.
서버 시작
서버 시작함 : 5001
클라이언트 연결됨 : /127.0.0.1 : 52120
데이터 받음 : Hello
데이터 보냄.

3. 웹으로 요청하기

■ HTTP 연결 방식

- 예전 휴대 단말은 데이터 통신의 송수신 속도가 느려서 소켓으로 연결하거나 웹 페이지를 보기 위해서는 많이 기다려야 함
- 비연결성(stateless)인 HTTP 프로토콜은 새로 연결을 만드는 데 따른 지연 시간이 길게 발생
- 최근 스마트폰 및 무선 네트워크 환경이 좋아져서 HTTP 프로토콜을 이용한 웹의 사용이 자연스러울 뿐만 아니라 일반 웹 사이트를 보는 풀 브라우징(full browsing)도 가능함
- 자바에서 사용하던 HTTP 관련 클래스를 그대로 사용할 수 있음

URLConnection 클래스

■ URLConnection 객체

[API]

```
public URLConnection openConnection()
```

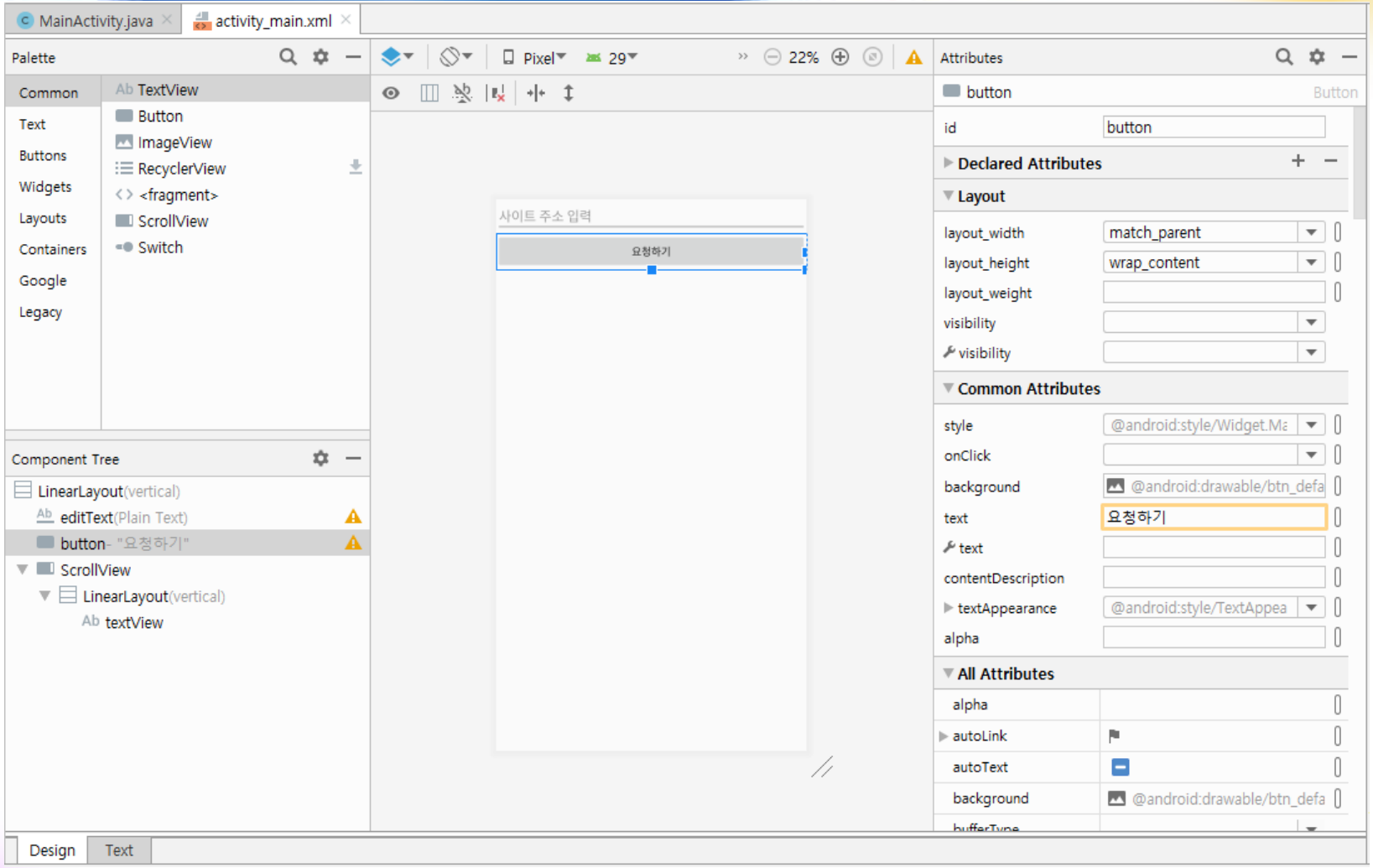
* HttpURLConnection 객체로 형변환(casting)하여 사용

■ 요청 관련 메소드

[API]

```
public void setRequestMethod(String method)
public void setRequestProperty(String field, String newValue)
```

화면 레이아웃 만들기


MainActivity.java
activity_main.xml
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Google
Legacy
TextView
Button
ImageView
RecyclerView
<fragment>
ScrollView
Switch
Component Tree
LinearLayout(vertical)
editText(Plain Text)
button- "요청하기"
ScrollView
LinearLayout(vertical)
textView
Pixel
29
22%
사이트 주소 입력
요청하기
Attributes
button
Button
id
button
Declared Attributes
Layout
layout_width
match_parent
layout_height
wrap_content
layout_weight
visibility
visibility
Common Attributes
style
@android:style/Widget.Ma
onClick
background
@android:drawable/btn_defa
text
요청하기
text
contentDescription
textAppearance
@android:style/TextAppea
alpha
All Attributes
alpha
autoLink
autoText
background
@android:drawable/btn_defa
Design
Text

버튼 눌렀을 때 호출할 request 메서드 정의

```
public void request(String urlStr) {
  StringBuilder output = new StringBuilder();
  try {
    URL url = new URL(urlStr);

    HttpURLConnection conn = (HttpURLConnection) url.openConnection();
    if (conn != null) {
      conn.setConnectTimeout(10000);
      conn.setRequestMethod("GET");
      conn.setDoInput(true);

      int resCode = conn.getResponseCode();
      BufferedReader reader = new BufferedReader(new InputStreamReader(conn.getInputStream()));
      String line = null;
      while (true) {
        line = reader.readLine();
        if (line == null) {
          break;
        }

        output.append(line + "\n");
      }
      reader.close();
      conn.disconnect();
```

❶ HttpURLConnection 객체 만들기

❷ 입력 데이터를 받기 위한 Reader 객체 생성하기

인터넷 권한과 usesCleartextTraffic 속성 추가

참조파일 SampleHttp>/app/manifests/AndroidManifest.xm

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="org.techtown.http">

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

  <application
    android:usesCleartextTraffic="true"

중략…
```

앱 실행하고 사이트 주소로 요청

http://www.kobis.or.kr/kobisopenapi/webservice/rest/boxoffice/searchDailyBoxOfficeList.json?key=430156241533f1d058c603178cc3ca0e&targetDt=20120101

HttpURLConnection으로 웹페이지 요청 화면

4. Volley 사용하기

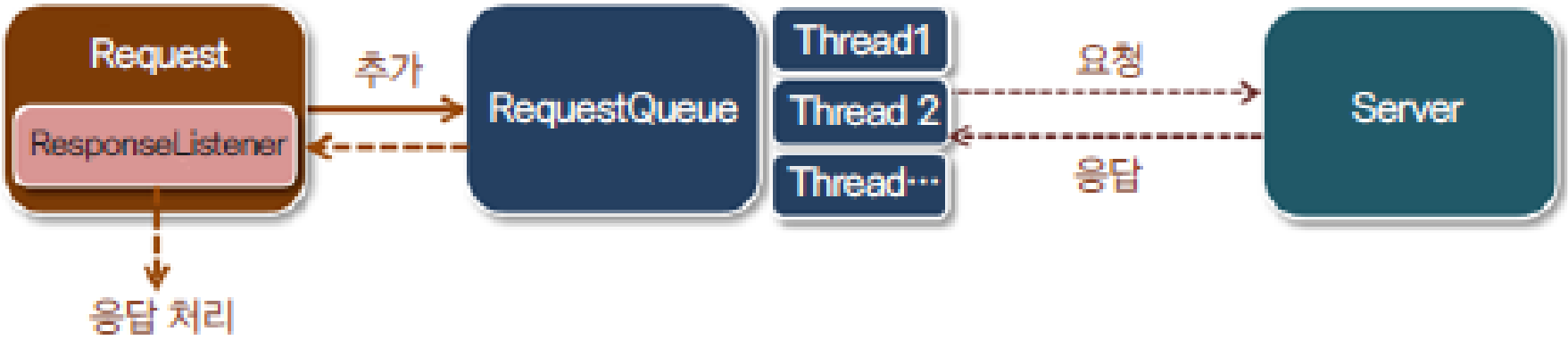
Volley를 이용한 요청과 응답 처리

- Volley
  - Android 앱의 네트워킹을 더 쉽고, 무엇보다도 더 빠르게 하는 HTTP 라이브러리

- Volley 장점
  - 네트워크 요청의 자동 예약.
  - 여러 개의 동시 네트워크 연결
  - 표준 HTTP 캐시 일관성을 갖춘 투명한 디스크 및 메모리 응답 캐싱
  - 요청 우선순위 지정 지원
  - 취소 요청 API. 단일 요청을 취소하거나 취소할 요청의 블록 또는 범위를 설정할 수 있습니다.
  - 용이한 맞춤설정(예: 재시도, 백오프)
  - 강력한 정렬 기능을 이용하여 네트워크에서 비동기식으로 가져온 데이터로 UI를 올바로 채우는 작업을 쉽게 실행할 수 있음.
  - 디버깅 및 추적 도구.

요청 보내기

- 요청객체 생성-> 요청객체를 요청큐에 추가->요청큐가 웹서버에 요청하고 응답을 받아줌.
- 요청큐가 내부적으로 스레드를 만들어 요청처리



Volley 라이브러리 추가

참조파일 SampleRequest>/Gradle Scripts/build.gradle(Module:app)

```
중략…

dependencies {

  중략…

  implementation 'com.android.volley:volley:1.1.0'
}
```

인터넷 권한 추가

참조파일 SampleRequest>/app/manifests/AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="org.techtown.request">

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

중략...
```

RequestQueue 객체 생성

참조파일 SampleRequest>/app/java/org.techtown.request/MainActivity.java

```
public class MainActivity extends AppCompatActivity {
  EditText editText;
  TextView textView;

  static RequestQueue requestQueue;

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

    editText = findViewById(R.id.editText);
    textView = findViewById(R.id.textView);

    Button button = findViewById(R.id.button);
    button.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        makeRequest();
      }
    });

    if (requestQueue == null) {
      requestQueue = Volley.newRequestQueue(getApplicationContext());
    }
  }
```

❶ RequestQueue 객체 생성하기

요청 메서드 추가

```
public void makeRequest() {
  String url = editText.getText().toString();

  StringRequest request = new StringRequest(Request.Method.GET, url,
    new Response.Listener<String>() {
      @Override
      public void onResponse(String response) {
        println("응답-> " + response);
      }
    },
    new Response.ErrorListener() {
      @Override
      public void onErrorResponse(VolleyError error) {
        println("에러-> " + error.getMessage());
      }
    }
  ) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
      Map<String,String> params = new HashMap<String,String>();

      return params;
    }
  };
```

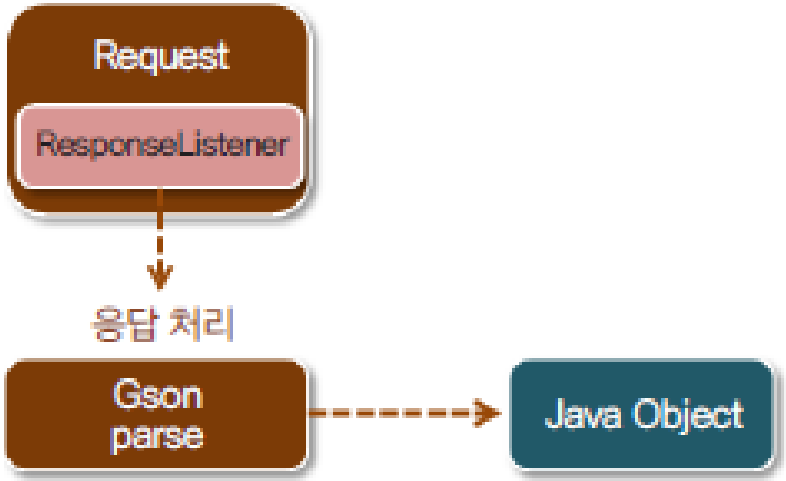
❷ 요청을 보내기 위한 StringRequest 객체 생성하기

요청 메서드 추가

```
        request.setShouldCache(false);
        requestQueue.add(request);
        println("요청 보냄.");
    }

    public void println(String data) {
        textView.append(data + "\n");
    }
}
```

5. JSON 데이터 다루기

Gson의 역할



Gson 라이브러리 추가

참조파일 SampleRequest2>/Gradle Scripts/build.gradle(Module:app)

```
중략…

dependencies {

  중략…

  implementation 'com.android.volley:volley:1.1.0'
  implementation 'com.google.code.gson:gson:2.8.2'
}
```

응답 JSON 객체화를 위한 클래스 정의

참조파일 SampleRequest2>/app/java/org.techtown.request/MovieList.java

```
public class MovieList {
  MovieListResult boxOfficeResult;
}
```

참조파일 SampleRequest2>/app/java/org.techtown.request/MovieListResult.java

```
public class MovieListResult {

  String boxofficeType;
  String showRange;

  ArrayList<Movie> dailyBoxOfficeList = new ArrayList<Movie>();
}
```

응답 JSON 객체화를 위한 클래스 정의

참조파일 SampleRequest2>/app/java/org.techtown.request/Movie.java

```
public class Movie {

  String rnum;
  String rank;
  String rankInten;
  String rankOldAndNew;
  String movieCd;
  String movieNm;
  String openDt;
  String salesAmt;
  String salesShare;
  String salesInten;
  String salesChange;
  String salesAcc;
  String audiCnt;
  String audiInten;
  String audiChange;
  String audiAcc;
  String scrnCnt;
  String showCnt;

}
```

Gson을 이용해 JSON 변환

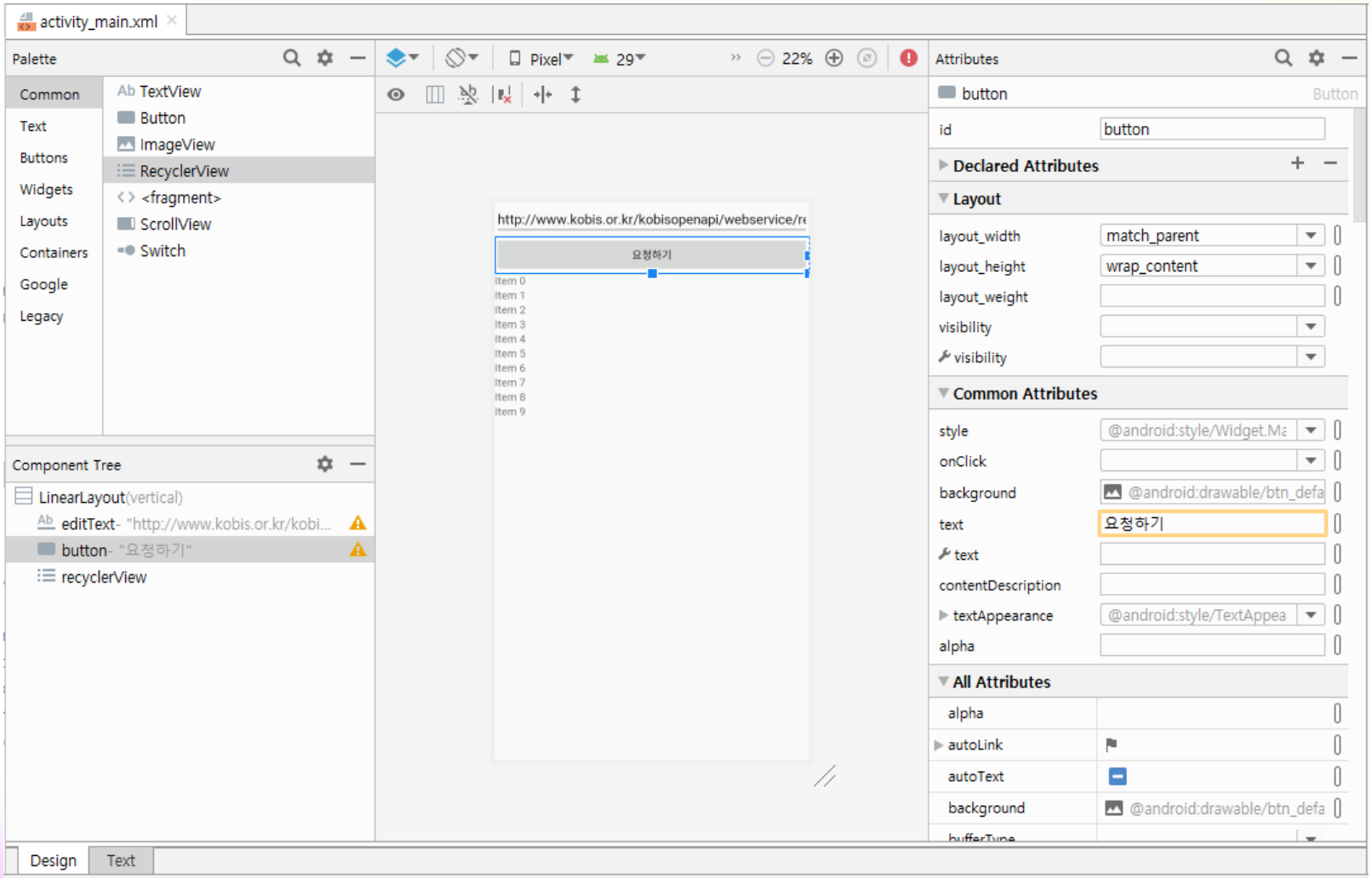
참조파일 SampleRequest2>/app/java/org.techtown.request/MainActivity.java

```
중략...
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                println("응답-> " + response);

                processResponse(response);
            }

            public void processResponse(String response) {
                Gson gson = new Gson();
                MovieList movieList = gson.fromJson(response, MovieList.class);
                println("영화 정보의 수: " + movieList.boxOfficeResult.dailyBoxOfficeList.size());
            }

중략...
```

JSON 문자열을 MovieList 객체로 변환하기

6. 영화 정보 가져와 보여주기

화면 레이아웃 만들기


activity_main.xml
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Google
Legacy
TextView
Button
ImageView
RecyclerView
<fragment>
ScrollView
Switch
Component Tree
LinearLayout(vertical)
editText- "http://www.kobis.or.kr/kobi...
button- "요청하기"
recyclerView
Pixel
29
22%
http://www.kobis.or.kr/kobisopenapi/webservice/re
요청하기
Item 0
Item 1
Item 2
Item 3
Item 4
Item 5
Item 6
Item 7
Item 8
Item 9
Attributes
button
Button
id
button
Declared Attributes
Layout
layout_width
match_parent
layout_height
wrap_content
layout_weight
visibility
visibility
Common Attributes
style
@android:style/Widget.Ma
onClick
background
@android:drawable/btn_defa
text
요청하기
text
contentDescription
textAppearance
@android:style/TextAppea
alpha
All Attributes
alpha
autoLink
autoText
background
@android:drawable/btn_defa
Design
Text

리싸이클러뷰를 위한 어댑터 정의

참조파일 SampleMovie>/app/java/org.techtown.movie/MovieAdapter.java

```
public class MovieAdapter {

  static class ViewHolder extends RecyclerView.ViewHolder {
    TextView textView;
    TextView textView2;

    public ViewHolder(View itemView) {
      super(itemView);

      textView = itemView.findViewById(R.id.textView);
      textView2 = itemView.findViewById(R.id.textView2);
    }

    public void setItem(Movie item) {
      textView.setText(item.movieNm);
      textView2.setText(item.audiCnt + " 명");
    }
  }
}
```

리싸이클러뷰의 각 아이템을 위한 XML 레이아웃 정의

참조파일 SampleMovie>/app/res/layout/movie_item.xml

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

  <android.support.v7.widget.CardView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="8dp"
    android:layout_marginRight="8dp"
    android:layout_marginTop="4dp"
    android:layout_marginBottom="4dp"
    app:cardBackgroundColor="#FFFFFFFF"
    app:cardCornerRadius="10dp"
    app:cardElevation="5dp" >

    <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="horizontal">
```

어댑터에 코드 추가

참조파일 SampleMovie>/app/java/org.techtown.movie/MovieAdapter.java

```
public class MovieAdapter extends RecyclerView.Adapter<MovieAdapter.ViewHolder> {
  ArrayList<Movie> items = new ArrayList<Movie>();

  @NonNull
  @Override
  public ViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int viewType) {
    LayoutInflater inflater = LayoutInflater.from(viewGroup.getContext());
    View itemView = inflater.inflate(R.layout.movie_item, viewGroup, false);

    return new ViewHolder(itemView);
  }

  @Override
  public void onBindViewHolder(@NonNull ViewHolder viewHolder, int position) {
    Movie item = items.get(position);
    viewHolder.setItem(item);
```

메인 액티비티에 코드 추가

참조파일 SampleMovie>/app/java/org.techtown.movie/MainActivity.java

```
public class MainActivity extends AppCompatActivity {
  EditText editText;
  TextView textView;

  RecyclerView recyclerView;
  MovieAdapter adapter;

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

    editText = findViewById(R.id.editText);
    textView = findViewById(R.id.textView);

    recyclerView = findViewById(R.id.recyclerView); ⟶ ❶ XML 레이아웃에 정의한 리싸이클러뷰 객체 참조하기

    LinearLayoutManager layoutManager =
                new LinearLayoutManager(this, LinearLayoutManager.VERTICAL, false);
    recyclerView.setLayoutManager(layoutManager);

    adapter = new MovieAdapter();
    recyclerView.setAdapter(adapter); ⟶ ❷ 리싸이클러뷰에 어댑터 설정하기

  }
}
```

Volley와 Gson 라이브러리 추가

참조파일 SampleMovie>/Gradle Scripts/build.gradle(Module:app)

```
중략…

dependencies {
  중략…

  implementation 'com.android.volley:volley:1.1.0'
  implementation 'com.google.code.gson:gson:2.8.2'
}
```

인터넷 권한 추가

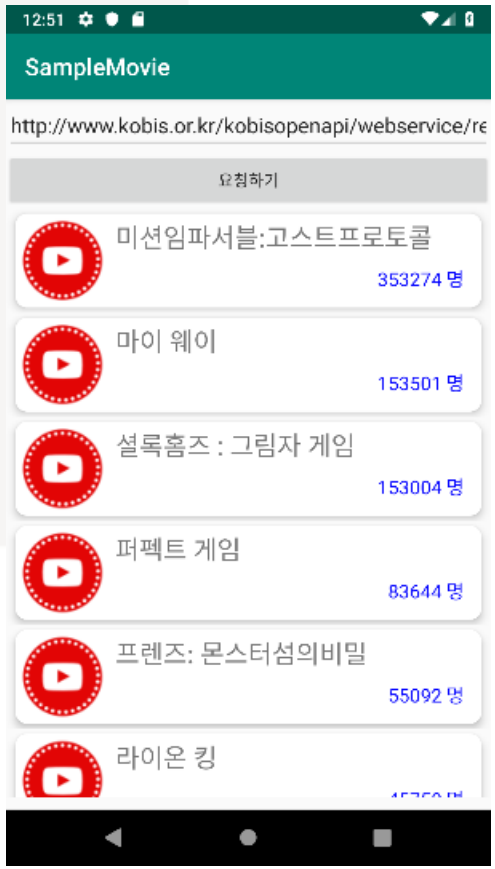
참조파일 SampleMovie>/app/manifests/AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="org.techtown.movie">

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

  <application
    android:usesCleartextTraffic="true"

중략…
```

버튼 클릭 시 요청 보내고 응답 처리

```
public void processResponse(String response) {
  Gson gson = new Gson();
  MovieList movieList = gson.fromJson(response, MovieList.class);
  println("영화 정보 수: " + movieList.boxOfficeResult.dailyBoxOfficeList.size());

  for (int i = 0; i < movieList.boxOfficeResult.dailyBoxOfficeList.size(); i++) {
    Movie movie = movieList.boxOfficeResult.dailyBoxOfficeList.get(i);

    adapter.addItem(movie);
  }

  adapter.notifyDataSetChanged();
 }
}
```

❷ 응답받은 JSON 문자열을 MovieList로 변환하기



Jsoup 사용

```
implementation 'org.jsoup:jsoup:1.11.3'
```

```
public class MainActivity extends AppCompatActivity {

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

        //샘플 html 코드 제시
        String html = "<html><head><title>첫번째 에제입니다.</title></head>"
                + "<body><h1>테스트</h1><p>간단히 HTML을 파싱해 보는 샘플예제입니다.</p></body></html>";

        Document doc = Jsoup.parse(html);

        Elements title = doc.select("title");
        Log.d("result: ", "doc= "+ doc);
        Log.d("result: ", "title= " + title);
    }
}
```

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

navaer 영화에서 상영작.예정작 jsoup로 크롤링하여 recyclerView에 출력하기.