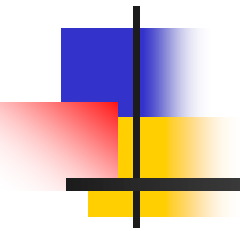


HttpURLConnection 실습

배 희호 교수
경복대학교
소프트웨어융합과

HttpURLConnection 클래스

- HttpClient를 삭제하면서 Google에서 제시한 대안이 HttpUrlConnection 임
- URL(HTTP Protocol 사용)을 통해 Server와 통신하는 JAVA Program을 개발하기 위해 URLConnection 및 HttpURLConnection Class를 사용
  - 기존의 URLConnection에 HTTP를 다루는데 필요한 Method를 추가한 Class
- URL.openConnection()으로 얻어진 URLConnection Object를 HttpURLConnection으로 Casting하여 Data 송수신을 행하고 disconnect()로 접속을 종료하는 방식으로 사용
- 예) File, Web Page를 Upload 및 Download, HTTP 요청 및 응답 전송 및 검색 등을 위한 Code를 작성할 수 있음

HttpURLConnection 클래스

- HttpURLConnection는 URLConnection을 구현한 Class
- java.net Class에서 제공하는 URL 요청을 위한 Class
- HTTP 통신을 위한 URL 연결을 할 수 있음
- Data Type이나 길이는 거의 제한이 없음
- MIME 형식에 맞는 Data를 "GET 방식"이나 "POST 방식"으로 보내거나 요청할 수 있음
- HttpURLConnection은 기본적으로 "GET 방식"을 사용
  - setRequestMethod() Method를 사용해서 기능 변경 가능
- URLConnection은 JAVA Application과 URL 간의 연결 관련 모든 Class의 Super Class
- URLConnection Class는 일반적인 URL에 대한 API를 제공하고, Sub Class인 HttpURLConnection는 HTTP 고유의 기능에 대한 추가 지원을 제공

HttpURLConnection 클래스

- 이 두 Class는 모두 Abstract Class이므로, URLConnection 및 HttpURLConnection의 새 Instance를 직접 만들 수 없음
- URL Object에서 연결을 통해 URLConnection의 Instance를 얻음
- URLConnection과 마찬가지로 protected로 선언되어 있음으로 기본적으로 개발자가 생성이 불가능 함

```
URL url = new URL("http://java.sun.com");
HttpURLConnection con = (HttpURLConnection)
                                    url.openConnection();
```

  - Http URL을 사용하는 URL Object의 openConnection() Method가 반환하는 URLConnection Class는 HttpURLConnection의 Instance가 될 수 있기에 반환된 URLConnection을 HttpURLConnection으로 Casting해서 사용

HttpURLConnection 클래스

- Method

Method	설 명
disconnect()	Server와의 연결 종료
getErrorStream()	Server를 연결할 수 없거나 오류가 발생한 경우 오류 Stream을 가져옴. Server에서 오류를 수정하는 방법에 대한 정보를 포함 할 수 있음
getFollowRedirects()	자동 Redirection 여부에 따라 true 또는 false를 반환
getHeaderField()	n번째 Header Field를 반환하고, 존재하지 않는 경우 null을 반환. URLConnection Class의 getHeaderField() Method를 재정의
getInstance FollowRedirects()	자동 Instance Redirection이 설정되었는지 여부에 따라 true 또는 false를 반환
getPermission()	대상 Host 및 Port에 연결하는 데 필요한 권한을 검색

HttpURLConnection 클래스

- Method

Method	설 명
getResponseCode()	Server에서 응답 상태를 검색하는 데 사용
getResponseMessage()	응답 Message를 검색
getRequestMethod()	요청 Method를 반환
setInstance FollowRedirects()	응답 Code 요청이 이 HTTP URL 연결 Instance에 의해 자동으로 Redirection되는지 여부를 설정. 보다 일반적인 setFollowRedirects()를 재정의
setRequestMethod()	요청 Method를 설정하는 데 사용. 기본값은 GET
setFixedLength StreamingMode()	미리 알려진 경우 출력 Stream에 기록된 Contents의 길이를 설정하는 데 사용

HttpURLConnection 클래스

- Method

Method	설 명
setConnectTimeout(int timeout)	✓ 연결 제한 시간을 1/1000초 단위로 지정 ✓ 0이면 무한 대기
setReadTimeout(int timeout)	✓ 읽기 제한 시간을 지정 ✓ 0이면 무한 대기
setUseCaches(boolean newValue)	✓ Cache 사용 여부를 지정
setDoInput(boolean newValue)	✓ 입력을 받을 것인지를 지정
setDoOutput(boolean newValue)	✓ 출력을 할 것인지를 지정
setRequesProperty (String field, String newValue)	✓ 요청 Header에 값을 설정
addRequestProperty (String field, String newValue)	✓ 요청 Header에 값을 추가 ✓ 속성의 이름이 같더라도 덮어 쓰지는 않음

HttpURLConnection 클래스

- Method

Method	설 명
setFollowRedirects()	3xx 응답 Code 요청이 자동으로 Redirection되는지 여부를 설정
setChunked StreamingMode()	Contents 길이를 알 수 없을 때 사용. 고정 길이의 Buffer를 만들어 Server에 쓰는 대신 Contents는 Chunk로 나뉘어 작성 일부 Server는 이 Mode를 지원하지 않음
usingProxy()	Proxy를 사용하여 연결이 설정된 경우 true를 반환하고, 그렇지 않으면 false를 반환

HttpURLConnection 클래스

- 일반적으로 Client Program은 URL을 통해 Server와 통신할 때 다음 단계를 따름
  - ① URL Object 만들기
  - ② URL에서 URLConnection Object 얻기
  - ③ URL 연결 구성
  - ④ Header Field 읽기
  - ⑤ Input Stream 가져오기 및 Data 읽기
  - ⑥ Output Stream 가져오기 및 Data 쓰기
  - ⑦ Stream 닫기
  - ⑧ 연결 종료

HttpURLConnection 클래스

- URL Object 생성

```
URL url = new URL("http://www.google.com");
```

  - 이 생성자는 URL 형식이 잘못된 경우 MalformedURLException을 throw함
  - 이 예외는 IOException의 Sub Class

```
URI uri = new URI("http://www.google.com");
URL url = uri.toURL();
```

HttpURLConnection 클래스

- URL에서 URLConnection Object 얻기
  - URLConnection Instance는 URL Object의 openConnection() Method 호출에 의하여 얻어 짐

```
URLConnection conn = url.openConnection();
```

  - Protocol이 http://인 경우 반환된 Object를 HttpURLConnection Object로 Casting할 수 있음

```
HttpURLConnection conn = (HttpURLConnection)
                         url.openConnection();
```

HttpURLconnection 클래스

- URLConnection과 마찬가지로 protected로 선언되어 있음으로 기본적으로 개발자가 직접 생성이 불가능 함
- 생성 방법

```
URI uri = new URI("http://www.naver.com");
URL url = uri.toURL();
HttpURLConnection conn = (HttpURLConnection)
                                    url.openConnection();
```

  - URL을 사용하는 URL Object의 openConnection() Method가 반환하는 URLConnection Class는 HttpURLConnection의 Instance가 될 수 있기에 반환된 URLConnection을 HttpURLConnection으로 Casting해서 사용
  - HTTP & TCP 연결을 시도하는 Object http 반환

HttpURLConnection 클래스

- URLConnection 구성
  - 실제로 연결을 설정하기 전에 TimeOut, Cache, HTTP 요청 방법 등과 같이 Client와 Server 간의 다양한 Option을 설정할 수 있음
  - URLConnection Class는 연결을 구성하기위한 다음과 같은 Method를 제공
  - setConnectTimeout(int timeout)
    - 연결 TimeOut 값을 설정(단위 : 밀리초)
    - java.net.SocketTimeoutException는 연결이 설정되기 전에 제한 시간이 만료되면 발생
    - 시간 초과가 0이면, 무한대 TimeOut(기본값)을 의미

HttpURLConnection 클래스

- URLConnection 구성
  - setReadTimeout(int timeout)
    - 읽기 TimeOut 값을 설정(단위 : 밀리초)
    - 제한 시간이 만료되고 연결의 입력 Stream에서 읽을 수 있는 Data가 없으면 SocketTimeoutException 발생
    - 시간 초과가 0이면, 무한대 TimeOut(기본값)을 의미
  - setDefaultUseCaches(boolean default)
    - URLConnection이 기본적으로 Cache를 사용하는지 여부를 설정(기본값 : true)
    - 이 Method는 URLConnection Class의 다음 Instance에 영향을 줌
  - setUseCaches (boolean useCaches)
    - 연결이 Cache를 사용하는지 여부를 설정(기본값 : true)

HttpURLConnection 클래스

- URLConnection 구성
  - setDoInput(boolean doInput)
    - URLConnection을 Server에서 Contents를 읽는 데 사용할 수 있는지 여부를 설정(기본값 : true)
  - setDoOutput (boolean doOutput)
    - URLConnection이 Server에 Data를 보내는 데 사용할 수 있는지 여부를 설정(기본값 : false)
  - setIfModifiedSince (long time)
    - 주로 HTTP Protocol에 대해 Client가 검색한 Contents의 마지막 수정 시간을 새로 설정
    - 예) Server가 지정된 시간 이후에 정적 Contents(이미지, HTML 등)가 변경되지 않았으면 Contents를 가져오지 않고 상태 Code 304(수정되지 않음)를 반환
    - Client는 지정된 시간보다 최근에 수정된 경우 새로운 Contents를 받게 됨

HttpURLConnection 클래스

- URLConnection 구성
  - setAllowUserInteraction (boolean allow)
    - 사용자 상호 작용을 활성화 또는 비활성화 함
    - 예) 필요한 경우 인증 대화 상자를 표시(기본값 : false)

HttpURLConnection 클래스

- HTTP 요청
  - URLConnection 자체는 Abstract Class이며 Protocol에 따라 Http, Https, Jar 등의 연결 Object가 반환
  - 연결에 성공한 후 URLConnection Class의 Method들로 연결의 속성을 설정
  - 기본 속성을 설정한 후 각 연결 Type별로 속성을 추가로 설정 할 수 있으며 Http 연결의 경우 요청 방식을 지정해야 함

HttpURLConnection 클래스

- HTTP 요청
  - 다음 Method는 URL의 각 부분을 분리

Method	반환 값	설 명
String getProtocol()	http	통신 방법을 정의하는 Protocol
int getDefaultPort()	80	Protocol이 정의하는 Default Port
int getPort()	-1	✓ URL에 정의된 Port ✓ 없을 경우 -1이 반환
String getHost()	www.chatting.com	Server 주소
String getFile()	/showroom?girl=3	Path와 Query
String getPath()	/showroom	Server내의 경로
String getQuery()	girl=3	Server로 전달되는 Query 변수 값

MalformedURLException예외가 발생

HttpURLConnection 클래스

- HTTP 요청 방식 지정 및 HTTP 통신 연결
  - HttpURLConnection Object를 얻어낸 후에는 필요한 경우 setRequestMethod(String) Method를 사용해 HTTP 요청 방식(GET, POST 등)을 지정하고 connect() Method를 사용해 HTTP 통신을 연결
  - setRequestMethod() Method를 생략할 경우 HTTP의 요청 방식은 기본적으로 GET 방식이므로 GET으로 지정

```
//get 방식의 경우 생략 가능
  connection.setRequestMethod(“GET");

  connection.connect();
 //주어진 url로 통신을 연결한다. 경우에 따라 생략 가능
```

HttpURLConnection 클래스

- 요청 방식 설정
  - HttpURLConnection은 기본적으로 GET 방식 사용
  - setRequestMethod() Method를 사용해서 요청 방식 변경 가능
    - 요청 방식은 대문자로 전달해야 함
    - 지정된 요청 방식 이외의 매개 변수 전달 시 java.net.ProtocolException이 발생

HttpURLConnection 클래스

- 요청 방식
  - HEAD : 문서의 Header 정보만 요청
  - GET : Web Server로부터 Resource를 가져옴
  - POST : Form에 입력된 내용을 Server로 전송
  - DELETE
    - Web Server의 Resource를 지움
    - 대부분의 Server는 기본적으로 DELETE를 허용하지 않거나 인증을 요구함
    - Server는 이 요청을 거절하거나 인증을 요청할 수 있으며, 허용하는 경우에도 응답은 구현에 따라 차이 발생
      - Server 설정에 따라 File을 지우기, 휴지통으로 이동, File을 읽을 수 없도록 표시 하는 등의 행위를 하게 됨

HttpURLConnection 클래스

- 요청 방식
  - PUT
    - Web Server로 Resource를 전송
    - PUT 요청도 File을 지울 때와 마찬가지로 보통 사용자 인증을 요구하며, PUT 메소드를 지원하도록 설정해줘야 함
  - OPTIONS
    - 특정 URL에 대해 지원되는 요청 Method의 목록을 반환
    - 요청 URL이 *인 경우 해당 요청의 대상은 Server에 있는 하나의 특정 URL이 아니라 Server 전체에 적용된다는 것을 의미

HttpURLConnection 클래스

- 요청 방식
  - TRACE
    - 요청을 추적함
    - Client가 보낸 요청이 Client와 Server사이에 있는 Proxy Server에서 변경되었는 지를 확인할 필요가 있는 경우 등에 쓰임

HttpURLConnection 클래스

- URLConnection 구성
  - setAllowUserInteraction (boolean allow)
    - 사용자 상호 작용을 활성화 또는 비활성화 함
    - 예) 필요한 경우 인증 대화 상자를 표시(기본값 : false)

- URLConnection 확인
  - conn.getResponseCode() Method를 통해 요청 전송
    - 정상적으로 응답이 오면(resCode == HTTP_OK) BufferReader를 통해 응답을 읽어 출력
    - 그렇지 않으면 연결 종료

HttpURLConnection 클래스

- Header Field 읽기
  - 연결이 이루어지면 Server는 URL 요청을 처리하고 Meta Data와 실제 Contents로 구성된 응답을 다시 보냄
  - MetaData는 Header Field라고 하는 <키-값> 쌍의 모음
  - Header Field는 Server에 대한 정보, 상태 Code, Protocol 정보 등을 나타냄
  - 실제 내용은 문서의 유형에 따라 Text, HTML, Image 등이 될 수 있음
  - URLConnection Class는 Header Field를 읽기 위한 다음과 같은 Method를 제공
  - getHeaderFields ()
    - 모든 Header Field를 포함하는 Map을 반환
    - Key는 Field 이름이고 값은 해당 Field 값을 나타내는 문자열 목록

HttpURLConnection 클래스

- Header Field 읽기
  - getHeaderField(int n)
    - n 번째 Header Field의 값을 읽음
  - getHeaderField(String name)
    - 명명된 Header Field의 값을 읽음
  - getHeaderFieldKey(int n)
    - n 번째 Header Field의 Key를 읽음
  - getHeaderFieldDate(String name, long default)
    - Date로 구문 분석된 명명된 Field의 값을 읽음
    - Field가 없거나 값 형식이 잘못된 경우 기본값이 대신 반환
  - getHeaderFieldInt(String name, int default)
    - 정수로 구문 분석된 명명된 Field의 값을 읽음
    - Field가 없거나 값 형식이 잘못된 경우 기본값이 대신 반환

HttpURLConnection 클래스

- Header Field 읽기
  - getHeaderFieldLong(String name, long default)
    - 긴 숫자로 구문 분석된 명명된 Field의 값을 읽음
    - Field가 없거나 값 형식이 잘못된 경우 기본값이 대신 반환 (이것은 Header Field를 읽는 일반적인 Method)
    - 자주 Access하는 일부 Header Field의 경우 URLConnection Class는 보다 구체적인 Method를 제공
  - getContentEncoding()
    - Contents의 Encoding 유형을 나타내는 Contents Encoding Header Field의 값을 읽음
  - getContentLength()
    - Contents의 크기(Byte)를 나타내는 Contents 길이 Header Field의 값을 읽음

HttpURLConnection 클래스

- Header Field 읽기
  - getContentType()
    - Contents의 유형을 나타내는 Contents 유형 Header Field의 값을 읽음
  - getDate()
    - Server의 날짜 시간을 나타내는 날짜 Header Field의 값을 읽음
  - getExpiration()
    - 만료 Header Field의 값을 읽고 응답이 오래된 것으로 간주되는 시간을 나타냄. 이는 Cache 제어를 위한 것
  - getLastModified()
    - Contents의 마지막 수정 시간을 나타내는 last-modified Header Field의 값을 읽음
    - 그리고 Sub Class HttpURLConnection은 추가 Method를 제공

HttpURLConnection 클래스

- Header Field 읽기
  - getResponseCode()
    - Server에서 보낸 HTTP 상태 Code를 반환
    - Header Field를 읽을 때 connect()를 호출하지 않고 암시적으로 연결이 설정

HttpURLConnection 클래스

- 입력 Stream을 사용한 Data 주고받기
  - HTTP 통신을 연결되면 연결된 Connection Object로부터 입출력 Stream을 얻어내서 Data를 주고받을 수 있음
  - 입력 Stream은 HttpURLConnection Object로부터 getInputStream() Method를 사용해서, 출력 Stream은 getOutputStream() Method를 사용해서 얻어냄

```
//입력 스트림을 얻어냄
  InputStream in = connection.getInputStream();
  int x = in.read(); //입력 스트림으로부터 실제의 데이터를 읽음

//출력 스트림을 얻어냄
  OutputStream out = connection.getOutputStream();
  out.write();       //출력 스트림으로 실제의 데이터를 씀(출력)
```

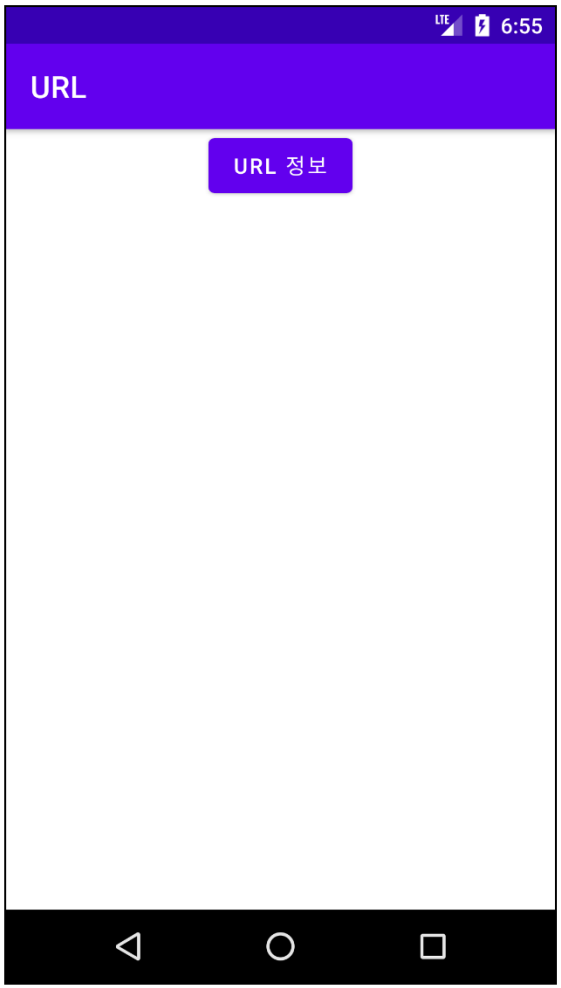
HttpURLConnection 클래스

- 연결 해제
  - 더 이상 HTTP 통신을 사용하지 않거나, 재사용하기 위해 connection Resource를 해제할 때는 HttpURLConnection Object의 disconnect() Method를 사용

```
connection.disconnect();    //커넥션 리소스를 해제
```

HttpURLConnection 예제 1

- https://m.naver.com의 속성 값을 출력해보자



HttpURLConnection 예제 1

- 사용자 인터페이스

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

    <Button
        android:id="@+id/load"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="URL 정보" />
```

HttpURLConnection 예제 1

- 사용자 인터페이스

```
    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView
            android:id="@+id/result"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="" />
    </ScrollView>
</LinearLayout>
```

HttpURLConnection 예제 1

- MainActivity.JAVA

```
public class MainActivity extends AppCompatActivity {
    String page = "https://m.naver.com";

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        StrictMode.ThreadPolicy policy =
                new StrictMode.ThreadPolicy.Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);
        TextView result = findViewById(R.id.result);
        Button button = findViewById(R.id.load);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {
                    URL url = new URL(page);
                    HttpURLConnection conn = (HttpURLConnection)
                                                    url.openConnection();
```

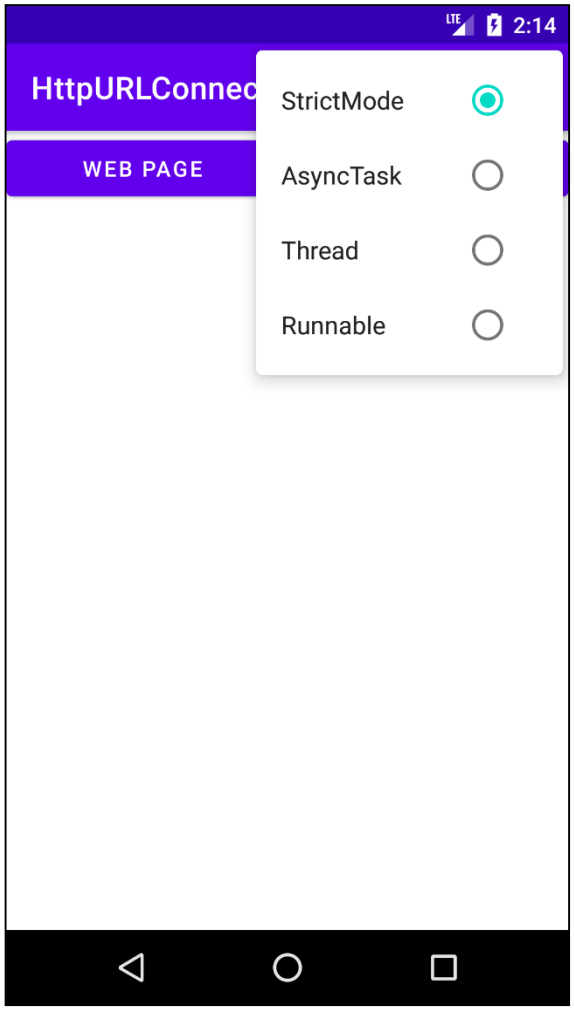
HttpURLConnection 예제 1

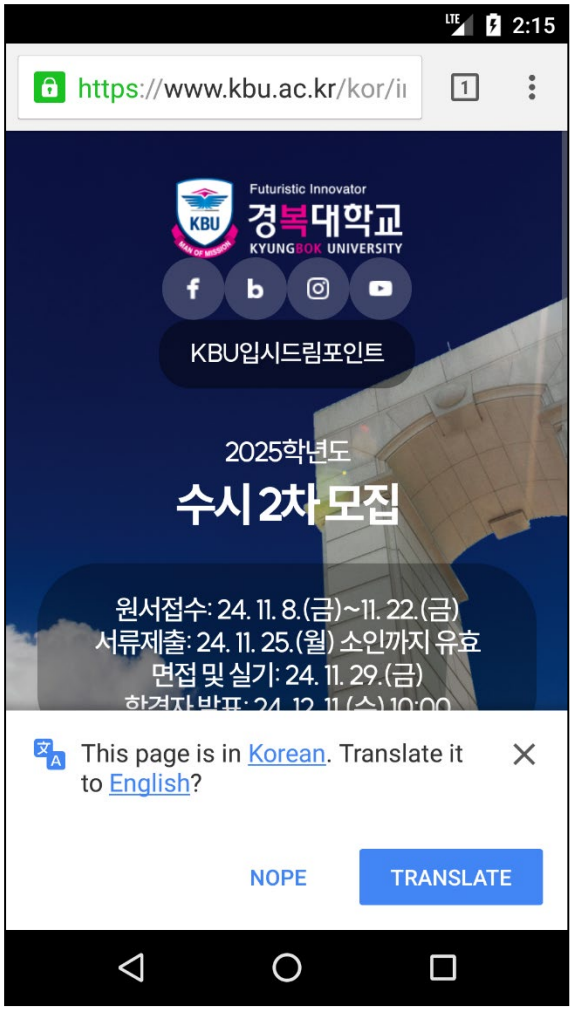
- MainActivity.JAVA

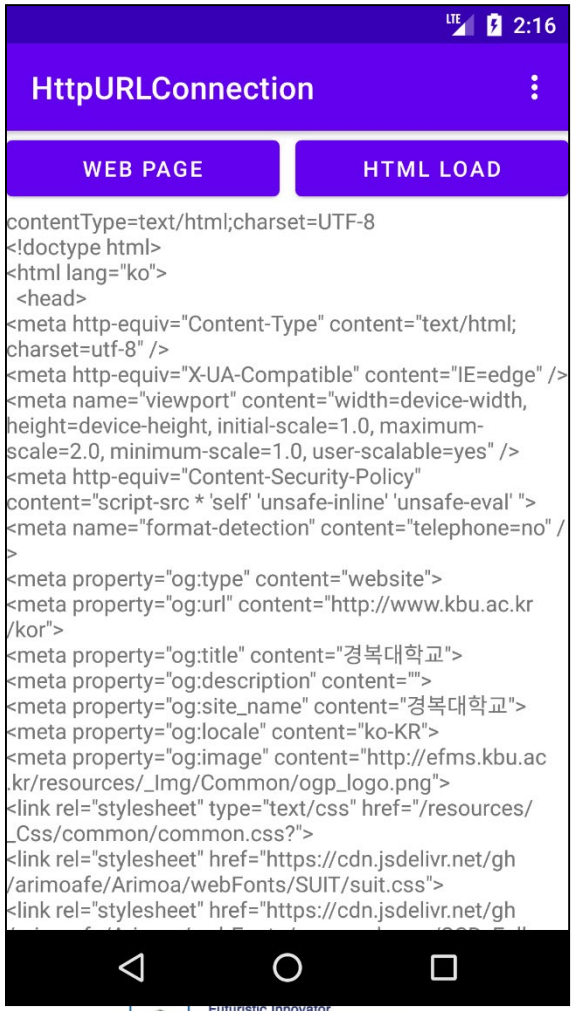
```
result.append("conn.toString() : " + conn.toString());
result.append("₩ngetAllowUserInteraction() : " +
                                        conn.getAllowUserInteraction());
result.append("₩ngetConnectTimeout() : " +
                                          conn.getConnectTimeout());
result.append("₩ngetContent() : " + conn.getContent());
result.append("₩ngetContentEncoding() : " +
                                       conn.getContentEncoding());
result.append("₩ngetContentLength() : " +
                                         conn.getContentLength());
result.append("₩ngetContentType() : " + conn.getContentType());
result.append("₩ngetDate() : " + conn.getDate());
result.append("₩ngetDefaultAllowUserInteraction() : " +
                                    conn.getDefaultAllowUserInteraction());
result.append("₩ngetUseChaches() : " + conn.getUseCaches());
result.append("₩ngetDefaultUserCaches() : " +
                                      conn.getDefaultUseCaches());
result.append("₩ngetDoInput() : " + conn.getDoInput());
```

HttpURLConnection 예제 2

- http://www.kbu.ac.kr의 html 문서를 다운받아보자



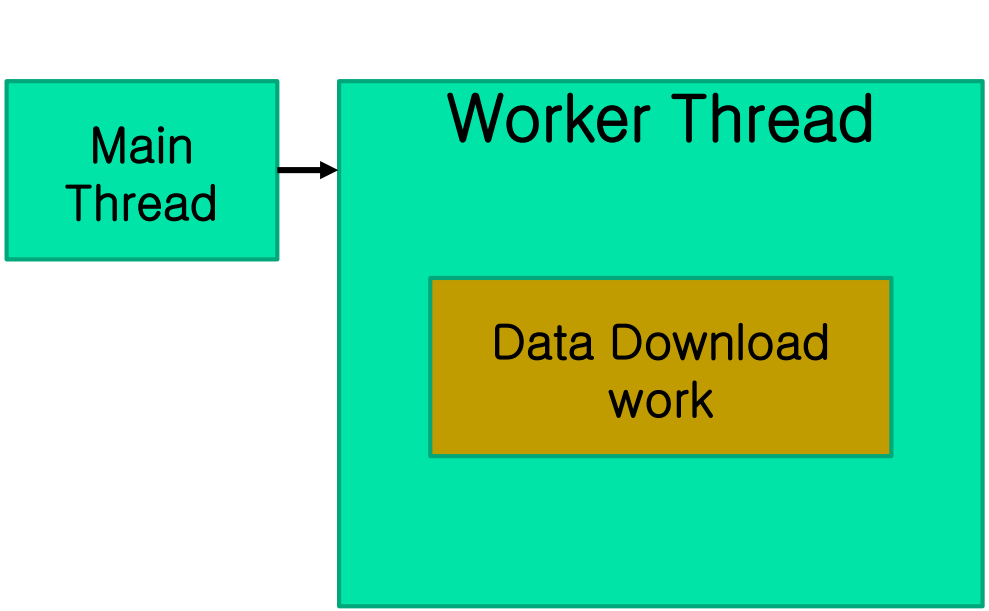


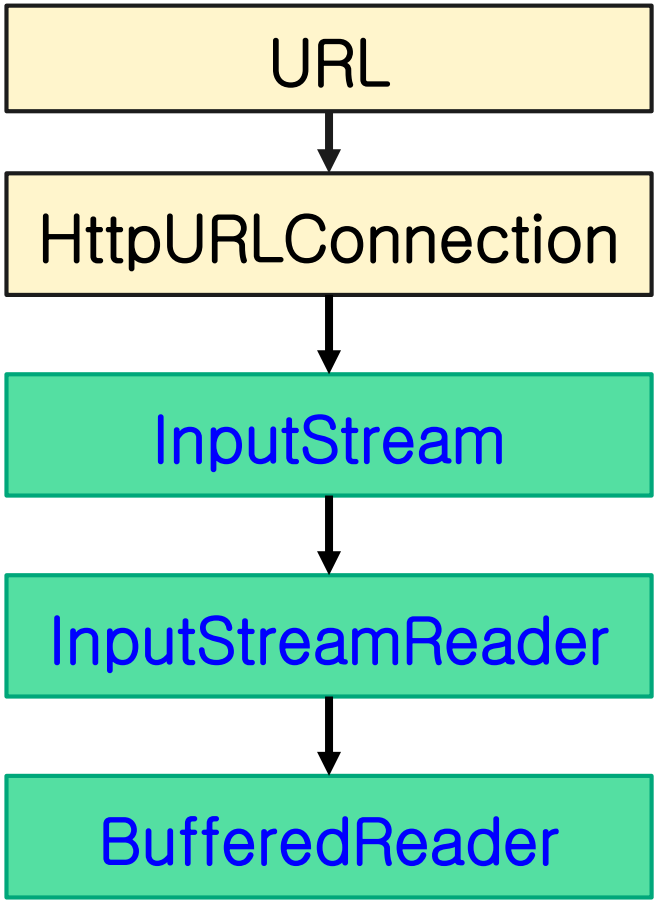


Futuristic Innovator
京福大學校
KYUNGBOK UNIVERSITY

HttpURLConnection 예제 2

- Web Server에서 Text 문서 받아오는 순서





HttpURLConnection 예제 2

- HTTP Protocol을 이용하여서 Network에서 File을 Download 할 때는 HttpURLConnection 사용
- Main Thread에서 직접 File을 Download하면 NetworkOnMainThreadException 예외가 발생

HttpURLConnection 예제 2

- MainActivity.JAVA

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main5);

    EditText editText = findViewById(R.id.editText);
    TextView textView = findViewById(R.id.textView);
    Button button = findViewById(R.id.button);
    button.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            String url = editText.getText().toString(); //URL 문자열 참조
            String output = request(url);               //request() 메소드 호출
            textView.setText(output);                   //결과물 표시
        }
    });
}
```

HttpURLConnection 예제 2

- MainActivity.JAVA

```
private String 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);
            conn.setDoOutput(true);
            int resCode = conn.getResponseCode(); //서버에 접속하여 요청
            if (resCode == HttpURLConnection.HTTP_OK) {
                BufferedReader reader = new BufferedReader( //스트림 객체 생성
                        new InputStreamReader(conn.getInputStream()));
```

HttpURLConnection 예제 2

- MainActivity.JAVA

```
                String line;
                while ((line = reader.readLine()) != null) {
                    output.append(line + "₩n");
                }
                reader.close();
                conn.disconnect();
            }
        }
    } catch (IOException e) {
        Toast.makeText(getBaseContext(), e.getMessage(),
                                                    Toast.LENGTH_SHORT).show();
    }
    return output.toString();
}
```

StrictMode

- Android StrictMode란?
  - StrictMode는 Android에서 실행 시간에 개발자가 잘못된 동작을 감지하고 수정할 수 있도록 도와주는 Debugging Tool
  - 주로 Main(UI) Thread에서 Network 작업이나 Disk 읽기/쓰기 같은 성능 저하 작업을 감지하는 데 사용
  - StrictMode는 개발용 Tool이며, 실제 출시 버전에서는 비활성화해야 함

StrictMode

- StrictMode의 주요 기능
  - StrictMode는 Thread 정책과 VM(가상 머신) 정책을 통해 성능 저하 및 비 효율적인 Code 감지를 수행
  - Thread Policy (스레드 정책)
    - UI Thread에서 Network 작업 또는 Disk 접근을 감지
    - StrictMode.ThreadPolicy.Builder 사용
  - VM Policy (가상 머신 정책)
    - Memory 누수 감지 (Activity, Cursor, File 등의 자원 누수)
    - SQLite Object가 닫히지 않는 경우 감지
    - StrictMode.VmPolicy.Builder 사용

StrictMode

- StrictMode를 사용하는 이유
  - UI Thread에서의 Blocking 작업 방지
    - App이 멈추는 문제 예방
  - Memory 누수 감지
    - 불필요한 Resource 사용 방지
  - File 및 Network I/O 감지
    - App 성능 최적화 가능
  - SQLite 누수 방지
    - Database 안정성 유지

StrictMode

- StrictMode 정책 Option

Method	설명
detectAll()	모든 문제 감지 (추천)
detectNetwork()	Network 작업 감지 (Main Thread에서 실행 금지)
detectDiskReads()	Disk 읽기 감지
detectDiskWrites()	Disk 쓰기 감지
detectLeakedClosableObjects()	닫히지 않은 Closeable Object 감지
detectLeakedSqlLiteObjects()	닫히지 않은 SQLite Object 감지
penaltyLog()	로그 출력 (기본적으로 사용)
penaltyDeath()	App을 즉시 종료 (강력한 디버깅 도구)
penaltyDialog()	사용자에게 경고 Dialog 표시
penaltyFlashScreen()	화면을 깜빡이게 하여 문제를 강조 (개발 중에만 사용)

StrictMode

- StrictMode 설정 방법
  - Thread Policy 설정
    - UI Thread에서의 부적절한 동작 감지

```
StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.Builder()
        .detectAll()  // 모든 문제 감지
        .penaltyLog() // 로그 출력
        .penaltyDialog() // 사용자에게 경고 다이얼로그 표시
        .build());
```

  - VM Policy 설정
    - Memory 누수 및 Resource 누수 감지

```
StrictMode.setVmPolicy(new StrictMode.VmPolicy.Builder()
     .detectLeakedSqlLiteObjects() // 닫히지 않은 SQLite 객체 감지
     .detectLeakedClosableObjects() // 닫히지 않은 리소스 감지
     .penaltyLog() // 로그 출력
     .build());
```

StrictMode

- StrictMode 적용

```
StrictMode.ThreadPolicy policy =
        new StrictMode.ThreadPolicy.Builder().permitAll().build();
StrictMode.setThreadPolicy(policy);
```

HttpURLConnection 예제 2

```
<?xml version="1.0" encoding="utf-8" ?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <group
        android:checkableBehavior="single"
        android:enabled="true">
        <item
            android:id="@+id/item1"
            android:checked="true"
            android:title="StrictMode" />
        <item
            android:id="@+id/item2"
            android:title="AsyncTask" />
        <item
            android:id="@+id/item3"
            android:title="Thread" />
        <item
            android:id="@+id/item4"
            android:title="Runnable" />
    </group>
</menu>
```

HttpURLConnection 예제 2

- 사용자 인터페이스

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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <Button
            android:id="@+id/button1"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_weight="1"
            android:text="Web Page" />
```

HttpURLConnection 예제 2

- 사용자 인터페이스

```
    <Button
        android:id="@+id/button2"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginStart="10dp"
        android:layout_weight="1"
        android:text="Html Load" />
</LinearLayout>

<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">

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

HttpURLConnection 예제 2

- 사용자 인터페이스

```
        <ScrollView
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/textViewt"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="" />
        </ScrollView>
    </FrameLayout>
</LinearLayout>
```

HttpURLConnection 예제 2

- MainActivity.JAVA

```
public class MainActivity2 extends AppCompatActivity {
    final String page = "http://www.kbu.ac.kr";
    private TextView textView;
    private WebView webView;
    private int type = 1;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);

        textView = findViewById(R.id.textView);
        webView = findViewById(R.id.webView);
        webView.getSettings().setJavaScriptEnabled(true);
```

HttpURLConnection 예제 2

- MainActivity.JAVA

```
Button button1 = findViewById(R.id.button1);
button1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        textView.setText("");
        webView.loadUrl(page);
    }
});
```

HttpURLConnection 예제 2

```
Button button2 = findViewById(R.id.button2);
button2.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        webView.clearView();
        textView.setText("");
        if (type == 1) {
            StrictMode.ThreadPolicy policy =
                    new StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(policy);
            HtmlDown downLoad = new HtmlDown(getBaseContext());
            textView.setText(downLoad.download(page));
        } else if (type == 2) {
            HtmlDownTask task = new HtmlDownTask(getBaseContext());
            try {
                textView.setText(task.execute(page).get());
            } catch (ExecutionException | InterruptedException e) {
                Toast.makeText(getBaseContext(), e.getMessage(),
                                                    Toast.LENGTH_SHORT).show();
            }
```

HttpURLConnection 예제 2

- MainActivity.JAVA

```
}else if (type == 3) {
    HtmlDownThread thread = new HtmlDownThread(
                                        getBaseContext(), page);
    thread.start();
    try {
        thread.join();
    } catch (InterruptedException e) {
        Toast.makeText(getBaseContext(), e.getMessage(),
                                        Toast.LENGTH_SHORT).show();
    }
    textView.setText(thread.getResult());
} else {
```

HttpURLConnection 예제 2

- MainActivity.JAVA

```
            } else {
                HtmlDownRunnable runnable = new HtmlDownRunnable(
                                                        getBaseContext(), page);
                Thread thread = new Thread(runnable);
                thread.start();
                try {
                    thread.join();
                } catch (InterruptedException e) {
                    Toast.makeText(getBaseContext(), e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
                textView.setText(runnable.getResult());
            }
        }
    });
}
```

HttpURLConnection 예제 2

- MainActivity.JAVA

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater menuInflater = getMenuInflater();
    menuInflater.inflate(R.menu.menu, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.item1:
            type = 1;
            break;
        case R.id.item2:
            type = 2;
            break;
```

HttpURLConnection 예제 2

- MainActivity.JAVA

```
            case R.id.item3:
                type = 3;
                break;
            case R.id.item4:
                type = 4;
        }
        webView.clearView();
        textView.setText("");
        item.setChecked(true);
        return true;
    }
}
```

HttpURLConnection 예제 2

- DownLoad.JAVA

```
public class DownLoad {
    private Context context;

    public DownLoad(Context context) {
        this.context = context;
    }

    public String download(String texturl) {
        StringBuilder buffer = new StringBuilder();
        try {
            URL url = new URL(texturl);
            URLConnection conn = url.openConnection();
            InputStream inputStream = conn.getInputStream();
            InputStreamReader streamReader = new InputStreamReader(inputStream);
            BufferedReader reader = new BufferedReader(streamReader);
            buffer.append("contentType=").append(conn.getContentType()).
                                                    append("₩n");
```

HttpURLConnection 예제 2

- DownLoad.JAVA

```
            String line;
            while ((line = reader.readLine()) != null) {
                buffer.append(line).append("₩n");
            }
            inputStream.close();
            streamReader.close();
            reader.close();
        } catch (Exception e) {
            Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
        }

        return buffer.toString();
    }
}
```

HttpURLConnection 예제 2

- DownLoadTask.JAVA

```
public class DownLoadTask extends AsyncTask<String, String, String> {
    private Context context;

    public DownLoadTask(Context context) {
        this.context = context;
    }

    @Override
    protected String doInBackground(String... strings) {
        StringBuffer buffer = new StringBuffer();
        try {
            URL url = new URL(strings[0]);
            URLConnection conn = url.openConnection();
            InputStream inputStream = conn.getInputStream();
            InputStreamReader streamReader = new InputStreamReader(inputStream);
            BufferedReader reader = new BufferedReader(streamReader);
            buffer.append("contentType=" + conn.getContentType() + "₩n");
```

HttpURLConnection 예제 2

- DownLoadTask.JAVA

```
            String line;
            while ((line = reader.readLine()) != null) {
                buffer.append(line + "₩n");
            }
            inputStream.close();
            streamReader.close();
            reader.close();
        } catch (Exception e) {
            publishProgress(e.getMessage());
        }
        return buffer.toString();
    }

    @Override
    protected void onProgressUpdate(String... values) {
        Toast.makeText(context, values[0], Toast.LENGTH_SHORT).show();
    }
}
```

HttpURLConnection 예제 2

- DownLoadThread.JAVA

```
public class DownLoadThread extends Thread{
    private Context context;
    private String page;
    private Handler handler = new Handler();
    private StringBuffer buffer = new StringBuffer();

    public DownLoadThread(Context context, String page) {
        this.context = context;
        this.page = page;
    }

    @Override
    public void run() {
        try {
            URL url = new URL(page);
            URLConnection conn = url.openConnection();
            InputStream inputStream = conn.getInputStream();
            InputStreamReader streamReader = new InputStreamReader(inputStream);
            BufferedReader reader = new BufferedReader(streamReader);
```

HttpURLConnection 예제 2

```
            buffer.append("contentType=" + conn.getContentType() + "₩n");
            String line;
            while ((line = reader.readLine()) != null) {
                buffer.append(line + "₩n");
            }
        } catch (Exception e) {
            handler.post(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(context, e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
            });
        }
    }

    public String getResult() {
        return buffer.toString();
    }
}
```

HttpURLConnection 예제 2

- DownLoadRunnable.JAVA

```
public class DownLoadRunnable implements Runnable{
    private Context context;
    private String page;
    private Handler handler = new Handler();
    private StringBuffer buffer = new StringBuffer();

    public DownLoadRunnable(Context context, String page) {
        this.context = context;
        this.page = page;
    }

    @Override
    public void run() {
        try {
            URL url = new URL(page);
            URLConnection conn = url.openConnection();
            InputStream inputStream = conn.getInputStream();
            InputStreamReader streamReader = new InputStreamReader(inputStream);
            BufferedReader reader = new BufferedReader(streamReader);
```

HttpURLConnection 예제 2

- DownLoadRunnable.JAVA

```
            buffer.append("contentType=" + conn.getContentType() + "₩n");
            String line;
            while ((line = reader.readLine()) != null) {
                buffer.append(line + "₩n");
            }
        } catch (Exception e) {
            handler.post(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(context, e.getMessage(),
                                                    Toast.LENGTH_SHORT).show();
                }
            });
        }
    }
    public String getResult() {
        return buffer.toString();
    }
}
```

HttpURLConnection 예제 2

- HtmlDownTask.JAVA

```
public class HtmlDownTask extends AsyncTask<String, String, String> {
    private Context context;

    public HtmlDownTask(Context context) {
        this.context = context;
    }

    @Override
    protected String doInBackground(String... strings) {
        StringBuilder builder = new StringBuilder();
        try {
            URL url = new URL(strings[0]);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            InputStream inputStream = conn.getInputStream();
            InputStreamReader streamReader = new InputStreamReader(inputStream);
            BufferedReader reader = new BufferedReader(streamReader);
            builder.append("contentType=" + conn.getContentType() + "₩n");
```

HttpURLConnection 예제 2

- HtmlDownTask.JAVA

```
            String line;
            while ((line = reader.readLine()) != null) {
                builder.append(line + "₩n");
            }
            inputStream.close();
            streamReader.close();
            reader.close();
            conn.disconnect();
        } catch (Exception e) {
            publishProgress(e.getMessage());
        }
        return builder.toString();
    }

    @Override
    protected void onProgressUpdate(String... values) {
        Toast.makeText(context, values[0], Toast.LENGTH_SHORT).show();
    }
}
```

HttpURLConnection 예제 2

- HtmlDownThread.JAVA

```
public class HtmlDownThread extends Thread{
    private Context context;
    private String page;
    private Handler handler;
    private StringBuilder builder;

    public HtmlDownThread(Context context, String page) {
        this.context = context;
        this.page = page;
        handler = new Handler();
        builder = new StringBuilder();
    }
```

HttpURLConnection 예제 2

- HtmlDownThread.JAVA

```
        } catch (Exception e) {
            handler.post(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(context, e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
            });
        }
    }

    public String getResult() {
        return builder.toString();
    }
}
```

HttpURLConnection 예제 2

- HtmlDownThread.JAVA

```
@Override
public void run() {
    try {
        URL url = new URL(page);
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        InputStream inputStream = conn.getInputStream();
        InputStreamReader streamReader = new InputStreamReader(inputStream);
        BufferedReader reader = new BufferedReader(streamReader);
        builder.append("contentType=" + conn.getContentType() + "₩n");
        String line;
        while ((line = reader.readLine()) != null) {
            builder.append(line + "₩n");
        }
        reader.close();
        streamReader.close();
        inputStream.close();
        conn.disconnect();
```

HttpURLConnection 예제 2

- HtmlDownNIO.JAVA

```
public class HtmlDownNIO extends Thread {
    private Context context;
    private String page;
    private Handler handler;
    private StringBuilder builder;

    public HtmlDownNIO(Context context, String page) {
        this.context = context;
        this.page = page;
        this.handler = new Handler();
        this.builder = new StringBuilder();
    }
```

HttpURLConnection 예제 2

```
@Override
public void run() {
    try {
        URL url = new URL(page);
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setRequestMethod("GET");
        ReadableByteChannel channel =
                                Channels.newChannel(conn.getInputStream());
        ByteBuffer buffer = ByteBuffer.allocate(4096);
        while (channel.read(buffer) > 0) {
            buffer.flip();
            builder.append(StandardCharsets.UTF_8.decode(buffer));
            buffer.clear();
        }
        conn.disconnect();
    } catch (IOException e) {
        handler.post(() -> Toast.makeText(context, e.getMessage(),
                                                Toast.LENGTH_SHORT).show());
    }
}
```

HttpURLConnection 예제 2

```
    public String getResult() {
        return builder.toString();
    }
}
```

HttpURLConnection 예제 2

- AsyncTask의 동작 순서
  - execute( ) 명령어를 통해 AsyncTask을 실행
  - AsyncTask로 Background 작업을 실행하기 전에 onPreExcuted( )가 실행. 이 부분에는 Image Loading 작업이라면 Loading 중 Image를 띄워 놓기 등, Thread 작업 이전에 수행할 동작을 구현
  - 새로운 Thread에서 Background 작업을 수행 execute( ) 메소드를 호출할 때 사용한 매개 변수를 전달 받음
  - doInBackground( )에서 중간 중간 진행 상태를 UI에 Update 하도록 하려면 publishProgress( ) Method 호출
  - onProgressUpdate( ) Method는 publishProgress( )가 호출 될 때 마다 자동으로 호출
  - doInBackground( ) Method에서 작업이 끝나면 onPostExcuted( )로 결과 매개 변수를 반환하면서 그 반환 값을 통해 Thread 작업이 끝났을 때의 동작을 구현

HttpURLConnection 예제 3

- URLConnection을 이용하여 image를 Down 받아보자

HttpURLConnection 예제 3

- 사용자 인터페이스

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity3">
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="top|center_horizontal"
        android:layout_margin="25dp"
        android:text="Display Image From URL" />
    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

HttpURLConnection 예제 3

- MainActivity.JAVA

```
public class MainActivity3 extends AppCompatActivity {
    String url = "http://192.168.219.100:8080/image/image.jpg";
    ImageView image;
    int type = 1;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main3);

        image = findViewById(R.id.imageView);
        Button button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (type == 1) {
```

HttpURLConnection 예제 3

- MainActivity.JAVA

```
        StrictMode.ThreadPolicy policy =
                new StrictMode.ThreadPolicy.Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);
        ImageDown downLoad = new ImageDown(MainActivity3.this);
        image.setImageBitmap(downLoad.download(url));
    } else if (type == 2) {
        ImageDownTask task = new ImageDownTask(MainActivity3.this);
        try {
            image.setImageBitmap(task.execute(url).get());
        } catch (ExecutionException | InterruptedException e) {
            Toast.makeText(getBaseContext(), e.getMessage(),
                                                Toast.LENGTH_SHORT).show();
        }
    } else if (type == 3) {
```

HttpURLConnection 예제 3

- MainActivity.JAVA

```
            ImageDownThread thread =
                                    new ImageDownThread(MainActivity3.this, url);
            thread.start();
            try {
                thread.join();
            } catch (InterruptedException e) {
                Toast.makeText(getBaseContext(), e.getMessage(),
                                                Toast.LENGTH_SHORT).show();
            }
            image.setImageBitmap(thread.getResult());
        } else {
```

HttpURLConnection 예제 3

- MainActivity.JAVA

```
                ImageDownRunnable runnable =
                        new ImageDownRunnable(MainActivity3.this, url);
                Thread thread = new Thread(runnable);
                thread.start();
                try {
                    thread.join();
                } catch (InterruptedException e) {
                    Toast.makeText(getBaseContext(), e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
                image.setImageBitmap(runnable.getResult());
            }
        }
    });
}
```

HttpURLConnection 예제 3

- MainActivity.JAVA

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater menuInflater = getMenuInflater();
    menuInflater.inflate(R.menu.menu, menu);
    return true;
}
```

HttpURLConnection 예제 3

```
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.item1:
                type = 1;
                break;
            case R.id.item2:
                type = 2;
                break;
            case R.id.item3:
                type = 3;
                break;
            case R.id.item4:
                type = 4;
        }
        image.setImageBitmap(null);
        item.setChecked(true);
        return true;
    }
}
```

HttpURLConnection 예제 3

- ImageDown.JAVA

```
public class ImageDown {
    private Context context;

    public ImageDown(Context context) {
        this.context = context;
    }

    public Bitmap download(String page) {
        Bitmap bitmap = null;
        try {
            URL url = new URL(page);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            if (conn != null) {
                conn.setConnectTimeout(10000);
                conn.setRequestMethod("GET");
                conn.setDoInput(true);
                conn.setDoOutput(false);
```

HttpURLConnection 예제 3

- ImageDown.JAVA

```
            if (conn.getResponseCode() == HttpURLConnection.HTTP_OK) {
                InputStream inputStream = conn.getInputStream();
                bitmap = BitmapFactory.decodeStream(inputStream);
                inputStream.close();
            }
            conn.disconnect();
        } else {
            Toast.makeText(context, "Network이 연결되지 않았음",
                                        Toast.LENGTH_SHORT).show();
        }
    } catch (IOException e) {
        Toast.makeText(context, e.getMessage(),
                                            Toast.LENGTH_SHORT).show();
    }

    return bitmap;
  }
}
```

HttpURLConnection 예제 3

- ImageDownTask.JAVA

```
public class ImageDownTask extends AsyncTask<String, String, Bitmap> {
    private Context context;

    public ImageDownTask(Context context) {
        this.context = context;
    }

    @Override
    protected Bitmap doInBackground(String... strings) {
        Bitmap bitmap = null;
        try {
            URL url = new URL(strings[0]);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            if (conn != null) {
                conn.setConnectTimeout(10000);
                conn.setRequestMethod("GET");
                conn.setDoInput(true);
                conn.setDoOutput(false);
```

HttpURLConnection 예제 3

- ImageDownTask.JAVA

```
            if (conn.getResponseCode() == HttpURLConnection.HTTP_OK) {
                InputStream inputStream = conn.getInputStream();
                bitmap = BitmapFactory.decodeStream(inputStream);
                inputStream.close();
            }
            conn.disconnect();
        } else {
            publishProgress("Network이 연결되지 않았음");
        }
    } catch (IOException e) {
        publishProgress(e.getMessage());
    }

    return bitmap;
}
```

HttpURLConnection 예제 3

- ImageDownTask.JAVA

```
    @Override
    protected void onProgressUpdate(String... values) {
        Toast.makeText(context, values[0], Toast.LENGTH_SHORT).show();
    }
}
```

HttpURLConnection 예제 3

- ImageDownThread.JAVA

```
public class ImageDownThread extends Thread{
    private Context context;
    private String page;
    private Handler handler = new Handler();
    private Bitmap bitmap = null;

    public ImageDownThread(Context context, String page) {
        this.context = context;
        this.page = page;
    }

    @Override
    public void run() {
        try {
            URL url = new URL(page);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            if(conn != null) {
```

HttpURLConnection 예제 3

- ImageDownThread.JAVA

```
            conn.setConnectTimeout(10000);
            conn.setRequestMethod("GET");
            conn.setDoInput(true);
            conn.setDoOutput(false);
            if(conn.getResponseCode() == HttpURLConnection.HTTP_OK) {
                InputStream inputStream = conn.getInputStream();
                bitmap = BitmapFactory.decodeStream(inputStream);
                inputStream.close();
            }
            conn.disconnect();
        } else {
            handler.post(new Runnable() {
                public void run() {
                    Toast.makeText(context, "Network이 연결되지 않았음",
                                                    Toast.LENGTH_SHORT).show();
                }
            });
        }
```

HttpURLConnection 예제 3

- ImageDownThread.JAVA

```
        } catch (IOException e) {
            handler.post(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(context, e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
            });
        }
    }

    public Bitmap getResult() {
        return bitmap;
    }
}
```

HttpURLConnection 예제 3

- ImageDownRunnable.JAVA

```
public class ImageDownRunnable implements Runnable {
    private Context context;
    private String page;
    private Handler handler = new Handler();
    private Bitmap bitmap = null;

    public ImageDownRunnable(Context context, String page) {
        this.context = context;
        this.page = page;
    }

    @Override
    public void run() {
        try {
            URL url = new URL(page);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            if(conn != null) {
```

HttpURLConnection 예제 3

- ImageDownRunnable.JAVA

```
            conn.setConnectTimeout(10000);
            conn.setRequestMethod("GET");
            conn.setDoInput(true);
            conn.setDoOutput(false);
            if(conn.getResponseCode() == HttpURLConnection.HTTP_OK) {
                InputStream inputStream = conn.getInputStream();
                bitmap = BitmapFactory.decodeStream(inputStream);
                inputStream.close();
            }
            conn.disconnect();
        } else {
            handler.post(new Runnable() {
                public void run() {
                    Toast.makeText(context, "Network이 연결되지 않았음",
                                                    Toast.LENGTH_SHORT).show();
                }
            });
        }
```

HttpURLConnection 예제 3

- ImageDownRunnable.JAVA

```
        } catch (IOException e) {
            handler.post(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(context, e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
            });
        }
    }

    public Bitmap getResult() {
        return bitmap;
    }
}
```

HttpURLConnection 예제 4

- Data Upload

HttpURLConnection 예제 4

- 사용자 인터페이스

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity4">

    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Data UpLoad(Post)"/>
    <TextView
        android:id="@+id/result"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="20dp"/>
</LinearLayout>
```

HttpURLConnection 예제 4

- MainActivity.JAVA

```
public class MainActivity4 extends AppCompatActivity {
    String postPage = "https://reqres.in/api/users";
    TextView textView;
    int type = 1;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main4);

        textView = findViewById(R.id.result);
        Button button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (type == 1) {
```

HttpURLConnection 예제 4

- MainActivity.JAVA

```
            UpLoad upLoad = new UpLoad(MainActivity4.this);
            textView.setText(upLoad.upload(postPage));
        } else if (type == 2) {
            UpLoadTask task = new UpLoadTask(MainActivity4.this);
            try {
                textView.setText(task.execute(postPage).get());
            } catch (ExecutionException | InterruptedException e) {
                Toast.makeText(getBaseContext(), e.getMessage(),
                                                    Toast.LENGTH_SHORT).show();
            }
        } else if (type == 3) {
```

HttpURLConnection 예제 4

- MainActivity.JAVA

```
            UpLoadThread thread =
                                new UpLoadThread(MainActivity4.this, postPage);
            thread.start();
            try {
                thread.join();
            } catch (InterruptedException e) {
                Toast.makeText(getBaseContext(), e.getMessage(),
                                                Toast.LENGTH_SHORT).show();
            }
            textView.setText(thread.getResult());
        } else {
```

HttpURLConnection 예제 4

- MainActivity.JAVA

```
                UpLoadRunnable runnable = new UpLoadRunnable(
                                                MainActivity4.this, postPage);
                Thread thread = new Thread(runnable);
                thread.start();
                try {
                    thread.join();
                } catch (InterruptedException e) {
                    Toast.makeText(getBaseContext(), e.getMessage(),
                                                    Toast.LENGTH_SHORT).show();
                }
                textView.setText(runnable.getResult());
            }
        }
    });
}
```

HttpURLConnection 예제 4

- MainActivity.JAVA

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater menuInflater = getMenuInflater();
    menuInflater.inflate(R.menu.menu, menu);
    return true;
}
```

HttpURLConnection 예제 4

```
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.item1:
                type = 1;
                break;
            case R.id.item2:
                type = 2;
                break;
            case R.id.item3:
                type = 3;
                break;
            case R.id.item4:
                type = 4;
        }
        textView.setText("");
        item.setChecked(true);
        return true;
    }
}
```

HttpURLConnection 예제 4

- UpLoad.JAVA

```
public class UpLoad {
    private Context context;
    private StringBuilder builder = new StringBuilder();
    public UpLoad(Context context) {
        this.context = context;
    }

    public String upload(String page) {
        StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().
                                                    permitAll().build();
        StrictMode.setThreadPolicy(policy);
        try {
            URL url = new URL(page);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setRequestProperty("Content-Type", "application/json");
            conn.setRequestMethod("POST");
            conn.setDoOutput(true);
            conn.setDoInput(true);
            conn.setChunkedStreamingMode(0);
```

HttpURLConnection 예제 4

- UpLoad.JAVA

```
JsonObject postData = new JsonObject();
postData.addProperty("name", "경복대");
postData.addProperty("job", "leader");

OutputStream out = new BufferedOutputStream(conn.getOutputStream());
BufferedWriter writer = new BufferedWriter(
                                    new OutputStreamWriter(out, "UTF-8"));
writer.write(postData.toString());
writer.flush();
InputStream stream = conn.getInputStream();
InputStreamReader reader = new InputStreamReader(stream);
BufferedReader buffer = new BufferedReader(reader);
String line;
while ((line = buffer.readLine()) != null) {
    builder.append(line);
}
buffer.close();
```

HttpURLConnection 예제 4

- UpLoad.JAVA

```
        } catch (IOException e) {
            Toast.makeText(context, e.getMessage(),
                                                    Toast.LENGTH_SHORT).show();
        }

        return builder.toString();
    }
}
```

HttpURLConnection 예제 4

- UpLoadTask.JAVA

```
public class UpLoadTask extends AsyncTask<String, String, String> {
    private Context context;

    public UpLoadTask(Context context) {
        this.context = context;
    }

    @Override
    protected String doInBackground(String... strings) {
        StringBuilder builder = new StringBuilder();
        try {
            URL url = new URL(strings[0]);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setRequestProperty("Content-Type", "application/json");
            conn.setRequestMethod("POST");
            conn.setDoOutput(true);
            conn.setDoInput(true);
            conn.setChunkedStreamingMode(0);
```

HttpURLConnection 예제 4

- UpLoadTask.JAVA

```
JsonObject postData = new JsonObject();
postData.addProperty("name", "경복대");
postData.addProperty("job", "leader");

OutputStream out = new BufferedOutputStream(conn.getOutputStream());
BufferedWriter writer = new BufferedWriter(
                                    new OutputStreamWriter(out, "UTF-8"));
writer.write(postData.toString());
writer.flush();
InputStream stream = conn.getInputStream();
InputStreamReader reader = new InputStreamReader(stream);
BufferedReader buffer = new BufferedReader(reader);
String line;
while ((line = buffer.readLine()) != null) {
    builder.append(line);
}
buffer.close();
```

HttpURLConnection 예제 4

- UpLoadTask.JAVA

```
        } catch (IOException e) {
            publishProgress(e.getMessage());
        }
        return builder.toString();
    }

    @Override
    protected void onProgressUpdate(String... values) {
        Toast.makeText(context, values[0], Toast.LENGTH_SHORT).show();
    }
}
```

HttpURLConnection 예제 4

- UpLoadThread.JAVA

```
public class UpLoadThread extends Thread{
    private Context context;
    private String page;
    private Handler handler = new Handler();
    private StringBuilder builder = new StringBuilder();

    public UpLoadThread(Context context, String page) {
        this.context = context;
        this.page = page;
    }

    @Override
    public void run() {
        try {
            URL url = new URL(page);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setRequestProperty("Content-Type", "application/json");
            conn.setRequestMethod("POST");
```

HttpURLConnection 예제 4

- UpLoadThread.JAVA

```
conn.setDoOutput(true);
conn.setDoInput(true);
conn.setChunkedStreamingMode(0);
JsonObject postData = new JsonObject();
postData.addProperty("name", "경복대");
postData.addProperty("job", "leader");

OutputStream out = new BufferedOutputStream(conn.getOutputStream());
BufferedWriter writer = new BufferedWriter(
                                    new OutputStreamWriter(out, "UTF-8"));
writer.write(postData.toString());
writer.flush();
InputStream stream = conn.getInputStream();
InputStreamReader reader = new InputStreamReader(stream);
BufferedReader buffer = new BufferedReader(reader);
String line;
while ((line = buffer.readLine()) != null) {
    builder.append(line);
}
```

HttpURLConnection 예제 4

- UpLoadThread.JAVA

```
            buffer.close();
        } catch (IOException e) {
            handler.post(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(context, e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
            });
        }
    }

    public String getResult() {
        return builder.toString();
    }
}
```

HttpURLConnection 예제 4

- UpLoadRunnable.JAVA

```
public class UpLoadRunnable implements Runnable{
    private Context context;
    private String page;
    private Handler handler = new Handler();
    private StringBuilder builder = new StringBuilder();

    public UpLoadRunnable(Context context, String page) {
        this.context = context;
        this.page = page;
    }

    @Override
    public void run() {
        try {
            URL url = new URL(page);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setRequestProperty("Content-Type", "application/json");
            conn.setRequestMethod("POST");
```

HttpURLConnection 예제 4

- UpLoadRunnable.JAVA

```
conn.setDoOutput(true);
conn.setDoInput(true);
conn.setChunkedStreamingMode(0);
JsonObject postData = new JsonObject();
postData.addProperty("name", "경복대");
postData.addProperty("job", "leader");

OutputStream out = new BufferedOutputStream(conn.getOutputStream());
BufferedWriter writer = new BufferedWriter(
                                new OutputStreamWriter(out, "UTF-8"));
writer.write(postData.toString());
writer.flush();
InputStream stream = conn.getInputStream();
InputStreamReader reader = new InputStreamReader(stream);
BufferedReader buffer = new BufferedReader(reader);
String line;
while ((line = buffer.readLine()) != null) {
    builder.append(line);
}
```

HttpURLConnection 예제 4

- UpLoadRunnable.JAVA

```
            buffer.close();
        } catch (IOException e) {
            handler.post(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(context, e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
            });
        }
    }

    public String getResult() {
        return builder.toString();
    }
}
```