Network Programming

[CACS355]

BCA 6th Sem

Er. Sital Prasad Mandal

(Email : info.sitalmandal@gmail.com)

Bhadrapur, Jhapa, Nepal

https://networkprogam-mmc.blogspot.com/

https://networkprogam-mmc.blogspot.com/

Unit-5

URLConnections

1. Opening URLConnections
2. Reading Data from a Server
3. Reading the Header
    i. Retrieving Specific Header Fields
    ii. Retrieving Arbitrary Header Fields
4. Caches
    i. Web Cache for Java
5. Configuring the Connection
    i. protected URL url
    ii. protected boolean connected
    iii. protected boolean allowUserInteraction
    iv. protected boolean doInput
    v. protected boolean doOutput
    vi. protected boolean ifModifiedSince
    vii. protected boolean useCaches
    viii. Timeouts
6. Configuring the Client Request HTTP Header
7. Writing Data to a Server
8. Security Considerations for URLConnections
9. Guessing MIME Media Types
10. HttpURLConnection
    i. The Request Method
    ii. Disconnecting from the Server
    iii. Handling Server Responses
    iv. Proxies
    v. Streaming Mode

https://networkprogam-mmc.blogspot.com/

Unit-5

URLConnections

By using URLConnection class, you can establish a connection to a resource specified by URL.

By using URLConnection object,

a. You can read the headers
b. You can read data from URL
c. You can write data to URL

https://networkprogam-mmc.blogspot.com/

Unit-5

URLConnections

By using URLConnection class, you can establish a connection to a resource specified by URL.

By using URLConnection object,

a. You can read the headers
b. You can read data from URL
c. You can write data to URL

Unit-5

URLConnection : Reading data from a server

It is a four step process:

1. Construct URL Object

```
URL url = new URL(resource);
```

2. Open connection to URL object

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

3. Get the input stream from URL connection

4. Read data and close the connection

Unit-5

URLConnection : Reading data from a server

Method-1

```
try {
    URL location = new URL("http://spm.com.np/");
    URLConnection conn = location.openConnection();
    InputStream is = conn.getInputStream();
    int Line;
    while ((Line = is.read()) != -1) {
        System.out.print((char) Line);
    }
    is.close();
} catch (MalformedURLException me) {
    System.out.println("MalformedURLException: " + me);
} catch (IOException ioe) {
    System.out.println("IOException: " + ioe);
}
```

Note:
Invoking the close() methods on the InputStream or OutputStream of an URLConnection after a request may free network resources associated with this instance.

Unit-5

URLConnection : Reading data from a server

Method-2

```
public class readDataP1 {
    public static void main(String[] args) {
        try {
            URL location = new URL("http://spm.com.np/");
            URLConnection conn = location.openConnection();
            DataInputStream dis = new DataInputStream(conn.getInputStream());
            BufferedReader br = new BufferedReader(new InputStreamReader(dis));
            String Line;
            while ((Line = br.readLine()) != null) {
                System.out.println(Line);
            }
            dis.close();
          } catch (MalformedURLException me) {
            System.out.println("MalformedURLException: " + me);
          } catch (IOException ioe) {
            System.out.println("IOException: " + ioe);
          }
    }
}
```

Note

Invoking the close() methods on the InputStream or OutputStream of an URLConnection after a request may free network resources associated with this instance.

Unit-5

3. Reading the Header

i. Retrieving Specific Header Fields

The first six methods request fields from the header. These are:

1. Content-type
2. Content-length
3. Content-encoding
4. Date
5. Last-modified
6. Expires

https://www.tufohss.edu.np/

Inspection in browser(F12 keypress)

Six Convenience Methods

These return the values of six header fields:

1. public int getContentLength()
2. public String getContentType()
3. public String getContentEncoding()
4. public long getExpiration()
5. public long getDate()
6. public long getLastModified()

```
String resource = "http://spm.com.np/";
/* Construct URL object */
URL url = new URL(resource);
/* Open URLConnection to this URL */
URLConnection conn = url.openConnection();
String contentType = conn.getContentType();
System.out.println("contentType = " + contentType);
```

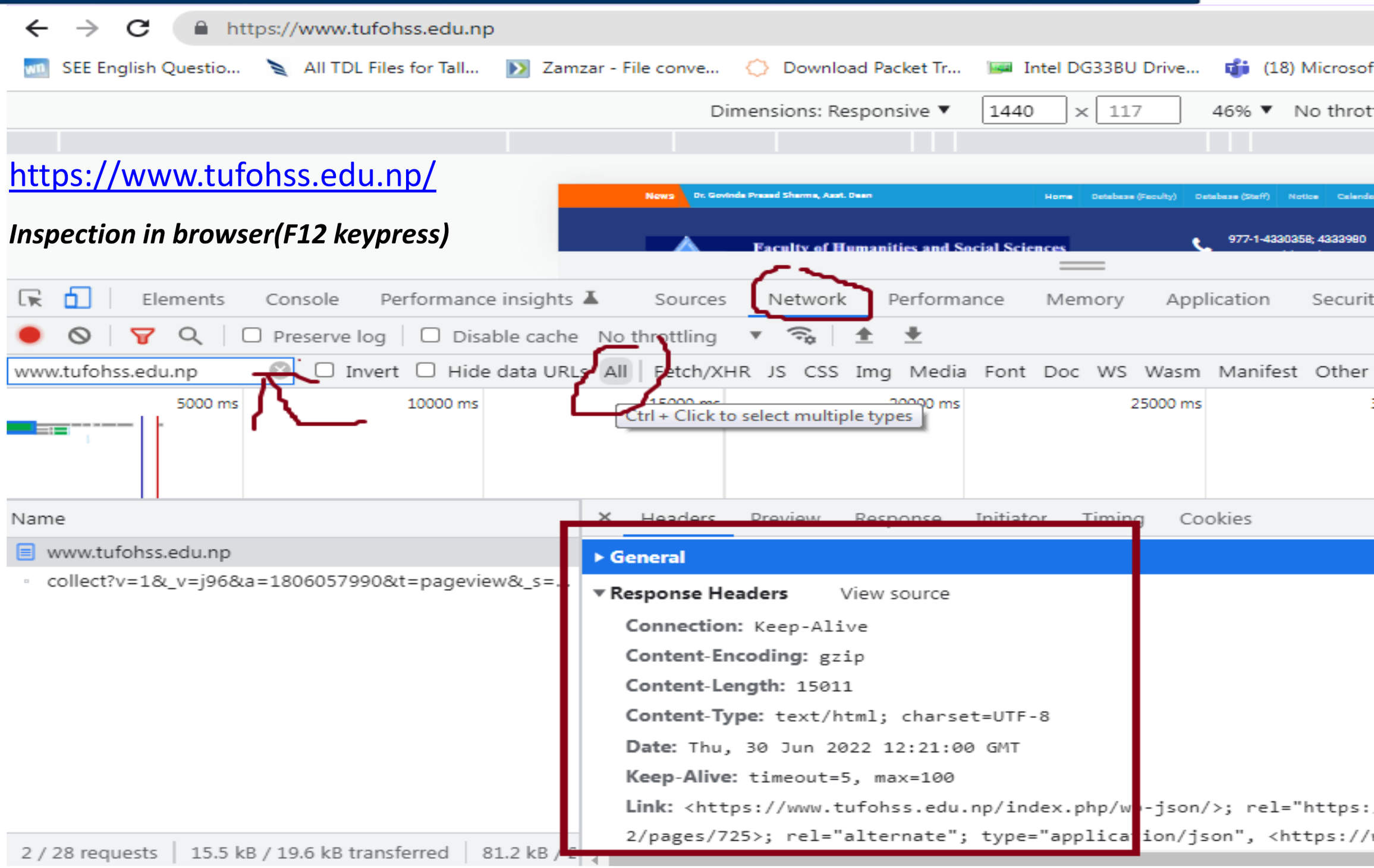
https://networkprogam-mmc.blogspot.com/

Unit-5

i. Retrieving Specific Header Fields

https://www.tufohss.edu.np/

Inspection in browser(F12 keypress)



Unit-5

i. Retrieving Specific Header Fields

```
public class HeaderViewer {
    public static void main(String args[]) throws Exception {
        String resource = "http://spm.com.np/";
        /* Construct URL object */
        URL url = new URL(resource);
        /* Open URLConnection to this URL */
        URLConnection conn = url.openConnection();
        String contentType = conn.getContentType();
        System.out.println("contentType = " + contentType);
        int conLengthInt = conn.getContentLength();
        System.out.println("conLengthInt = " + conLengthInt);
        long contentLengthLong = conn.getContentLengthLong();
        System.out.println("contentLengthLong = " + contentLengthLong);
        String contentEncoding = conn.getContentEncoding();
        System.out.println("contentEncoding = " + contentEncoding);

        long dateInMillis = conn.getDate();
        Date documentSent = new Date(dateInMillis);
        System.out.println("documentSent = " + documentSent);
        long expirationMillis = conn.getExpiration();
        Date expireDate = new Date(expirationMillis);
        System.out.println("expireDate = " + expireDate);
        long lastModifiedMills = conn.getLastModified();
        Date lastModifiedDate = new Date(lastModifiedMills);
        System.out.println("lastModifiedDate = " + lastModifiedDate);

    }
}
```

```
contentType = text/html
conLengthInt = 923
contentLengthLong = 923
contentEncoding = null
documentSent = Thu Jun 30 18:34:18 NPT 2022
expireDate = Thu Jan 01 05:30:00 NPT 1970
lastModifiedDate = Wed May 18 17:29:40 NPT 2022
```

Unit-5

ii. Retrieving Arbitrary Header Fields

URLConnection : Get specific header field

URLConnection class provides getHeaderField method, by using this you can get the value of specific header field.

public String getHeaderField(String name)

Returns the value of the named header field, or null if there is no such field in the header.

Unit-5

ii. Retrieving Arbitrary Header Fields

```
public class SpecificHeader {
    public static void main(String args[]) throws Exception {
        String resource = "http://spm.com.np/";

        /* Construct URL object */
        URL url = new URL(resource);

        /* Open URLConnection to this URL */
        URLConnection conn = url.openConnection();

        String contentType = conn.getHeaderField("Content-Type");
        String tranferEncoding = conn.getHeaderField("Transfer-Encoding");
        String lastModified = conn.getHeaderField("Last-Modified");

        System.out.println("contentType : " + contentType);
        System.out.println("tranferEncoding : " + tranferEncoding);
        System.out.println("lastModified : " + lastModified);
    }
}
```

```
contentType : text/html
tranferEncoding : null
lastModified : Wed, 18 May 2022 11:44:40 GMT
```

Unit-5

ii. Retrieving Arbitrary Header Fields

URLConnection : getHeaderFieldKey : Get the nth header field

public String getHeaderFieldKey(int n)

Returns the key for the nth header field. The request method is header zero and has a null key.

```
public class nthheaderfield {
    public static void main(String args[]) throws Exception {
        String resource = "http://spm.com.np/";
        /* Construct URL object */
        URL url = new URL(resource);
        /* Open URLConnection to this URL */
        URLConnection conn = url.openConnection();
        for (int i = 1;; i++) {
            String headerField = conn.getHeaderFieldKey(i);
            if (headerField == null) {
                break;
            }
            System.out.println(headerField + " : " + conn.getHeaderField(i));
        }
    }
}
```

Unit-5

ii. Retrieving Arbitrary Header Fields

Read All Header Fields

```
try{
    URL url=new URL("https://www.tufohss.edu.np");
    HttpURLConnection huc = (HttpURLConnection) url.openConnection();
    for(int i=1;i<=8;i++){
       System.out.println(huc.getHeaderFieldKey(i) + " = " + huc.getHeaderField(i));
    }
    huc.disconnect();
}catch(Exception e){System.out.println(e);}
```

Output:

```
Date = Thu, 22 Jul 2021 18:08:17 GMT
Server = Apache
Location = https://www.ambition.edu.np/contact
Cache-Control = max-age=2592000
Expires = Sat, 21 Aug 2021 18:08:17 GMT
Content-Length = 248
Keep-Alive =timeout=5, max=1500
Connection = Keep-Alive
```

Unit-5

4. Caches

- Web browsers have been caching pages and images for years.
- By default, the assumption is that a page accessed with GET over HTTP can and should be cached.
- A page accessed with HTTPS or POST usually shouldn't be.

HTTP Cache Headers Format

HTTP/1.1 200 OK

Date: Sun, 21 Apr 2013 15:12:46 GMT

Server: Apache

Connection: close

Content-Type: text/html; charset=ISO-8859-1

Cache-control: max-age=604800

Expires: Sun, 28 Apr 2013 15:12:46 GMT

Last-modified: Sat, 20 Apr 2013 09:55:04 GMT

ETag: "67099097696afcf1b67e"

Unit-5

4. Caches

Web Cache

- Web caching is **the activity of storing data for reuse**, such as a copy of a web page served by a web server.
- It is cached or stored the first time a user visits the page and the next time a user requests the same page, a cache will serve the copy, which helps keep the origin server from getting overloaded.
- By default, Java does not cache anything. To install a system-wide cache of the URL class will use, you need the following:

a) A concrete subclass of ResponseCache

b) A concrete subclass of CacheRequest

c) A concrete subclass of CacheResponse

Unit-5

5. Configuring the Connection

- protected URL getURL()
- protected void setDoInput(boolean doInput) // true = read; false = do not read
- protected boolean getDoInput()
- protected void setDoOutput(boolean doOutput) // true = write; false = do not write
- protected boolean getDoOutput()
- protected void setAllowUserInteraction(boolean allowUserInteraction) //user interaction is allowed;
- protected boolean getAllowUserInteraction()
- protected void setUseCaches(boolean useCaches) // whether a cache will be used if it's available
- protected boolean getUseCaches()
- protected void setIfModifiedSince(long ifModifiedSince) //set modified date;
- protected long getIfModifiedSince()

Unit-5

5. Configuring the Connection

Example 7-12. Print the URL of a URLConnection to http://www.oreilly.com/

```
import java.io.*;
import java.net.*;
public class URLPrinter {
public static void main(String[] args) {
    try {
    URL u = new URL("http://www.oreilly.com/");
    URLConnection uc = u.openConnection();
    System.out.println(uc.getURL());
    } catch (IOException ex) {
    System.err.println(ex);
    }
}
}
```

O/P:

% **java URLPrinter**
http://www.oreilly.com/

```
URL u = new
URL("http://www.example.org");
URLConnuction uc = u.openConnection();
uc.setConnectTimeout(30000);
uc.setReadTimeout(45000);
```

Unit-5

6. Configuring the Client Request HTTP Header

An HTTP client (e.g., a browser) sends the server a request line and a header.
For example, here's an HTTP header that Chrome sends:

```
Accept:text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Charset:ISO-8859-1,utf-8;q=0.7,*;q=0.3
Accept-Encoding:gzip,deflate,sdch
Accept-Language:en-US,en;q=0.8
Cache-Control:max-age=0
Connection:keep-alive
Cookie:reddit_first=%7B%22firsttime%22%3A%20%22first%22%7D
Host:lesswrong.com
User-Agent:Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_3) AppleWebKit/537.31
(KHTML, like Gecko) Chrome/26.0.1410.65 Safari/537.31
```

- ✓ **public void setRequestProperty(String name, String value)**

```
uc.setRequestProperty("Cookie“, "username=elharo;
password=ACD0X9F23JJJn6G; session=100678945");
```

whenever the client requests a URL from that server, it includes a Cookie field in the HTTP request header that looks like this:

```
Cookie: username=elharo; password=ACD0X9F23JJJn6G; session=100678945;
```

- ✓ **public void addRequestProperty(String name, String value)**
- ✓ **public String getRequestProperty(String name)**
- ✓ **public Map<String,List<String>> getRequestProperties()**

Unit-5

7. Writing Data to a Server Steps only

```
public class writeDataToServer {
    public static void writeDataToServer(String url, String data) {
        try {
            URL u = new URL(url);

            URLConnection urlConnection = u.openConnection();
            urlConnection.setDoOutput(true);

            OutputStream outputStream = urlConnection.getOutputStream();
            OutputStream buffered = new BufferedOutputStream(outputStream);
            OutputStreamWriter out = new OutputStreamWriter(buffered, "8859_1");

            out.write(data);
            out.flush();
            out.close();
        } catch (IOException ex) {
            System.err.println(ex);
        }
    }
}
```

Unit-5

8. Security Considerations for URLConnections

- URLConnection objects are subject to all the usual **security restrictions** about making network connections, reading or writing files.
- Before attempting to connect a URL, you may want to know whether that connection will be allowed.
- the URLConnection class has a getPermission( )

```
URL u = new URL("http://www.java2s.com");
URLConnection uc = u.openConnection();
System.out.println(uc.getPermission());
```

Output

("java.net.SocketPermission" "www.java2s.com:80" "connect,resolve")

Unit-5

9. Guessing MIME Media Types

- A **media type** (also known as a **Multipurpose Internet Mail Extensions or MIME type**) indicates the nature and format of a document, file, or assortment of bytes

Examples of MIME types are:

- **`text/html`** for normal web pages
- **`text/plain`** for plain text (Default)
- **`application/octet-stream`** meaning "download this file"
- **`application/x-java-applet`** for Java™ applets
- **`application/pdf`** for Adobe® PDF documents.

```
#def    image/x-bitmap
<!      text/html
<body   text/html
<head>  text/html
<html>  text/html
! XPM2  image/x-pixmap
GIF8    image/gif
```

public static String guessContentTypeFromName(String name)
public static String guessContentTypeFromStream(InputStream in)

MIME types—also sometimes called ***Internet media types*** or ***Content-types***—describe the media type of content either contained in email or served by web servers or web applications, and are intended to help guide a web browser to correctly process and display the content.

MIME is an expansion of the original Internet e-mail protocol that exchanges different kinds of data files on the Internet: text, audio, video, images, application programs, and others.

https://www.geeksforgeeks.org/mime-media-types/

Unit-5

10. HttpURLConnection

1. The Request Method
2. Disconnecting from the Server
3. Handling Server Responses
4. Proxies
5. Streaming Mode

Unit-5

10. HttpURLConnection

Working with HttpURLConnection class

- HttpURLConnection class is used to establish a connection to http end points.
- By using HttpURLConnection class, we can open a connection to given http end point, read the content, write content, read and write headers, read response status codes etc.

In this we can use all these things:

- Read URL Contents using HttpURLConnection
- Get headers from url using HttpURLConnection
- Adding headers using HttpURLConnection
- Post data to url using HttpURLConnection

Unit-5

10. HttpURLConnection

- **java.net.HttpURLConnection** is an abstract subclass of **URLConnection** that provides some additional methods specific to the HTTP protocol.
- URL connection objects that are returned by an http URL will be instances of **java.net.HttpURLConnection**.
- It provides some additional methods that are helpful when working specifically with **httpURLs**.

Cast that URLConnection to HttpURLConnection like this:

```
URL u = new URL("http://lesswrong.com/");
URLConnection uc = u.openConnection();
HttpURLConnection http = (HttpURLConnection) uc;
```

Or, skipping a step, like this:

```
URL u = new URL("http://lesswrong.com/");
HttpURLConnection http = (HttpURLConnection) u.openConnection();
```

Unit-5

10. HttpURLConnection

HttpURLConnection Method

Method	Description
void disconnect()	It shows that other requests from the server are unlikely in the near future.
Static boolean getFollowRedirects()	It returns a boolean value to check whether or not HTTP redirects should be automatically followed.
String getHeaderField(int n)	It returns the value of nth header file.
long getHeaderFieldDate(String name, long Default)	It returns the value of the named field parsed as a date.
String getHeaderFieldKey(int n)	It returns the key for the nth header file.
String getRequestMethod()	It gets the request method.
int getResponseCode()	It gets the response code from an HTTP response message.
String getResponseMessage()	It gets the response message sent along with the response code from a server.
void setRequestMethod(String method)	Sets the method for the URL request, one of: GET POST HEAD OPTIONS PUT DELETE TRACE are legal, subject to protocol restrictions.

Unit-5

10. HttpURLConnection

Method	Action performed
disconnect()	Indicated that requests to the server are highly unlikely in the future.
getFollowRedirects()	Returns true or false depending on automatic redirection or not.
getHeaderField()	Returns the nth header field, or null if it does not exist. It overrides the getHeaderField method of URLConnection class.
getPermission()	Retrieves the permission required to connect to a destination host and port.
getResponseCode()	Used to retrieve the response status from server.
getResponseMessage()	Retrieves the response message.
getRequestMethod()	Returns the request method.
setRequestMethod()	Used to set the request method. Default is GET
usingProxy()	Returns true if connection is established using a proxy, else false

Unit-5

URLConnection : Get all header fields and values

URLConnection class provides getHeaderFields method, which returns a Map of header field.

public Map<String,List<String>> getHeaderFields()
Returns a map of header fields and respective values.

Unit-5

URLConnection : Get all header fields and values

```
public static void main(String args[]) throws Exception {
    String resource = "http://spm.com.np/";

    /* Construct URL object */
    URL url = new URL(resource);

    /* Open URLConnection to this URL */
    URLConnection conn = url.openConnection();

    Map<String, List<String>> headerFields = conn.getHeaderFields();

    Set<String> keys = headerFields.keySet();

    for (String key : keys) {
        List<String> values = headerFields.get(key);
        System.out.print(key + ": ");

        for (String value : values) {
            System.out.print(value + "\t");
        }
        System.out.println();
    }
}
```

```
date: Thu, 30 Jun 2022 13:02:21 GMT
Keep-Alive: timeout=5, max=100
null: HTTP/1.1 200 OK
server: LiteSpeed
content-length: 923
last-modified: Wed, 18 May 2022 11:44:40 GMT
vary: User-Agent
Connection: Keep-Alive
content-type: text/html
accept-ranges: bytes
```

Unit-5

URLConnection : Get all header fields and values

```
public static void main(String[] args) {
        try {
            String resource = "http://spm.com.np/";
            /* Construct URL object */
            URL u = new URL(resource);
            URLConnection uc = u.openConnection();
            for (int j = 1; ; j++) {
                String header = uc.getHeaderField(j);
                if (header == null) break;
                System.out.println(uc.getHeaderFieldKey(j) + ": " + header);
            }
        } catch (MalformedURLException ex) {
            System.err.println(" it not a URL I understand.");
        } catch (IOException ex) {
            System.err.println(ex);
        }
        System.out.println();
}
```

```
Connection: Keep-Alive
Keep-Alive: timeout=5, max=100
content-type: text/html
last-modified: Wed, 18 May 2022 11:44:40 GMT
accept-ranges: bytes
content-length: 923
date: Thu, 30 Jun 2022 13:10:26 GMT
server: LiteSpeed
vary: User-Agent
```

Unit-5

Read URL Contents by Response Code using HttpURLConnection

Follow below steps to read the contents from an url.

Step 1: Create a connection to the url.

```
URL url = new URL(urlToConnect);
HttpURLConnection httpUrlConnection = (HttpURLConnection)
url.openConnection();
```

Step 2: Create input stream from the url connection.

```
int responseCode = httpUrlConnection.getResponseCode();
InputStream inputStream = null;

if (responseCode >= 200 && responseCode < 400) {
    inputStream = httpUrlConnection.getInputStream();
} else {
    inputStream = httpUrlConnection.getErrorStream();
}
```

Step 3: Read the content from input stream and print them to console.

```
BufferedReader br = new BufferedReader(new
InputStreamReader(inputStream));

String line = null;

while ((line = br.readLine()) != null) {
    System.out.println(line);
}
```

Unit-5

Read URL Contents by Response Code using HttpURLConnection

```
public class httpConnection {

    public static void main(String[] args) throws IOException {
        String url = "http://spm.com.np/";
        HttpURLConnection httpUrlConnection = getURLConnection(url);
        InputStream inputStream = getContent(httpUrlConnection);
        printInputStream(inputStream);
    }

    private static HttpURLConnection getURLConnection(String urlToConnect) throws IOException {
        URL url = new URL(urlToConnect);
        HttpURLConnection httpUrlConnection = (HttpURLConnection) url.openConnection();
        return httpUrlConnection;
    }

    private static InputStream getContent(HttpURLConnection httpUrlConnection) throws IOException {
        int responseCode = httpUrlConnection.getResponseCode();
        InputStream inputStream = null;
        if (responseCode >= 200 && responseCode < 400) {
            inputStream = httpUrlConnection.getInputStream();
        } else {
            inputStream = httpUrlConnection.getErrorStream();
        }
        return inputStream;
    }

    private static void printInputStream(InputStream inputStream) throws IOException {
        BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
        String line = null;
        while ((line = br.readLine()) != null) {
            System.out.println(line);
        }
    }
}
```