

ID2212 Network Programming with Java
Lecture 7

Working with Web Resources and
URL Connections

Leif Lindbäck, Vladimir Vlassov
KTH/ICT/SCS
HT 2016

Outline

- **Accessing Web Resources in Java**
 - **Locating Web resources: URL**
 - **Communicating with HTTP servers: URL connections**
 - **Downloading and uploading data on the Web**
 - **Retrieving Web resources pointed to by URLs: content handlers**
 - **Images, audio, HTML documents, classes, files**
- **Developing custom protocol and content handlers.**

Locating Web Resources: [java.net.URL](#)

- **[java.net.URL](#)** represents a URL - Uniform Resource Locator of a Web resource (service)
 - A URL object is used to locate and to grab Web resources.
 - The JDK provides implementations for many different protocols, for example HTTP and FTP.
- A URL with the http scheme:

<http://www.it.kth.se:80/labs/se/index.html>

protocol

host

port

resource name (path)

URL Constructors

URL (...)

- (String locator)
- (URL context, String locator)
- (String protocol, String serverName, String resource)
- (String protocol, String serverName, int port, String resource)

```
try {
    URL url1 =
        new URL("https://docs.oracle.com/javase/8/docs/api/");
    // create an absolute URL from a base and a
    // relative URL:
    URL url2 = new URL(url1, "java/net/URL.html");
} catch (MalformedURLException e) {
    e.printStackTrace();
}
```

URL Stream Handler

- A stream protocol handler that knows how to make a connection for a particular protocol type, such as http, ftp.
- A stream handler can be found by a **URL** object the following two ways.
 - If **URLStreamHandlerFactory** has been set by a call to **URL.setURLStreamHandlerFactory**, call its method **createURLStreamHandler**
 - Else, read the system property **java.protocol.handler.pkgs** and look for a package with that name plus the protocol, then look in that package for a **URLStreamHandler** subclass called **Handler**.

Communicating with a Web Server: java.net.URLConnection

- The **URLConnection** class represents a communication link to the resource.

```
URL url =  
    new URL("http://www.kth.se");  
URLConnection urlc = url.openConnection();
```

- The server is contacted only when needed, for example when the content or a header field is read:

```
myURLConnection.getContent()  
myURLConnection.getContentLength(), etc.
```

Input and Output Streams of URLConnection

- **URLConnection provides input and output streams**
 - **Input stream for downloading the resource contents, e.g. classes, images.**

```
URLConnection urlc =url.openConnection();
InputStream in = urlc.getInputStream();
```
 - **Output stream for uploading data to the server at the corresponding URL, e.g. posting query to a CGI script or a servlet**

```
URLConnection urlc =url.openConnection();
OutputStream o = urlc.getOutputStream();
```

Retrieving Resource Information and Content (HTTP)

- **Methods of `URLConnection` to get information about the resource and its content:**

<code>getLength</code>	<code>getDate</code>
<code>getContentType</code>	<code>getExpiration</code>
<code>getContent</code>	<code>getLastModified</code>
<code>getContentEncoding</code>	

- The information is obtained via the **GET** request.
- The server sends a **MIME** header and resource data in reply.

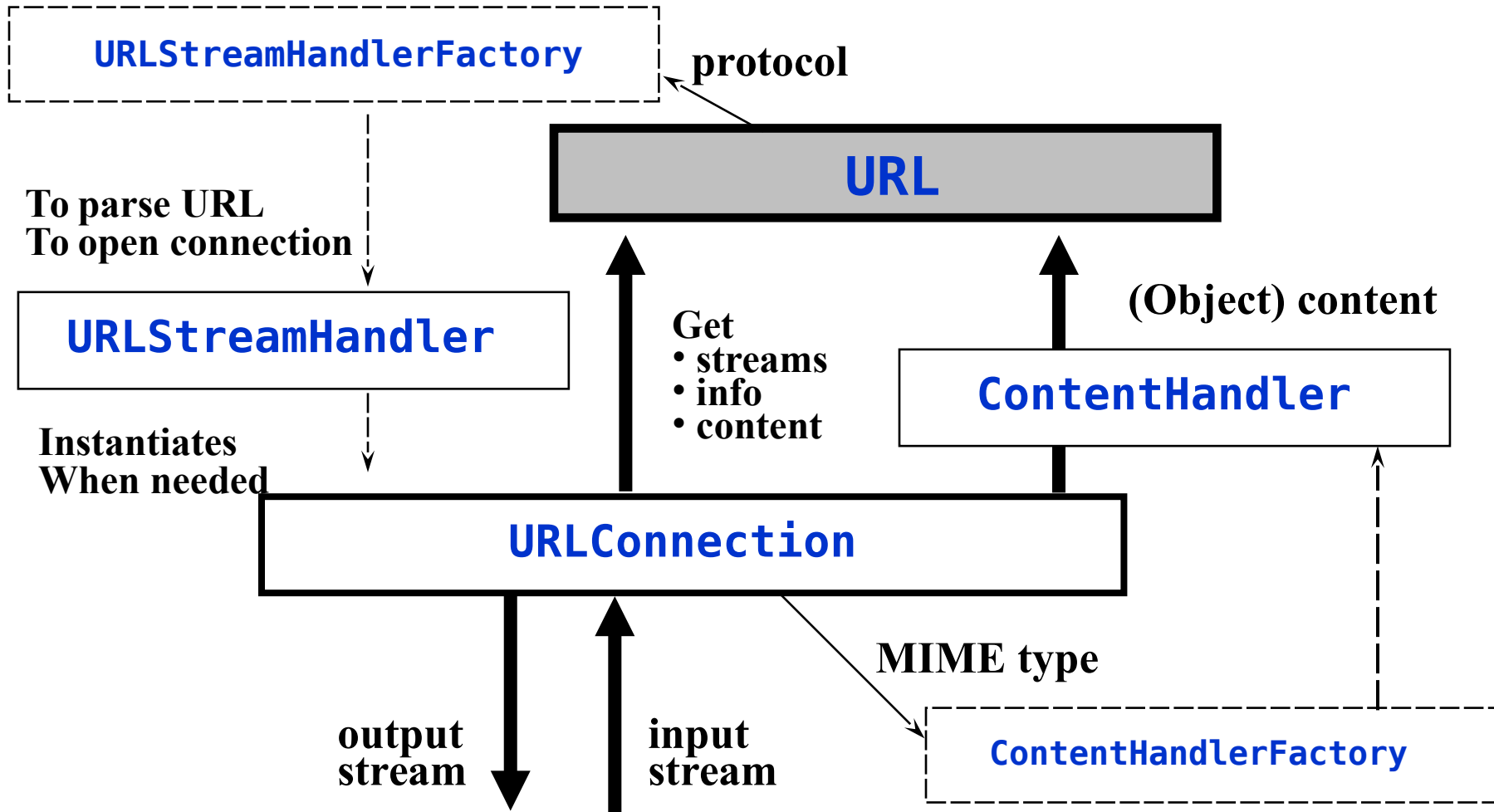
Handling downloaded content: java.net.ContentHandler

- A ContentHandler has a **getContent** method that reads data from the input stream of the given URL connection and converts it into a Java object.
- The handler is associated with one or several MIME types that it can handle.
- The handler can support caching of content.

Looking for a Content Handler

- A content handler can be found by a **URLConnection** object the following two ways.
 - If a **ContentHandlerFactory** has been set by a call to **URLConnection.setContentHandlerFactory**, call its method **createContentHandler(String mimeType)**, otherwise:
 - Read the system property **java.content.handler.pkgs** and look for a package with that name plus the major part of the mime type, then look in that package for a **ContentHandler** subclass named after the minor part of the mime type.

Architecture of URL Related Classes



The Shorthand Method

`url.getContent()`

1. Creates a `URLStreamHandler` instance
2. Calls `URLStreamHandler.openConnection(URL)`
 - The stream handler creates a URL connection
3. Calls `URLConnection.getContent()`
 - The connection connects to server
4. Calls `URLConnection.getContentHandler()`
 - The connection creates a content handler for a given MIME type such as “text/html”
5. Calls `ContentHandler.getContent()`
 - The handler converts downloaded data into a Java object

An Example, `GetHttpRequest`

```
/** Downloads the resource at the specified url ...6 lines */
public GetHttpRequest(String url) throws MalformedURLException {
    this.resourceUrl = new URL(url);
}
```

```
/** Loads the resource specified by this instance ...5 lines */
public void loadResource() throws IOException {
    final Image image;
    System.out.println("MIME: " + resourceUrl.openConnection().getContentType());
    Object resource = resourceUrl.getContent();
    if (resource instanceof ImageProducer) {
        handleImageResource(resource);
    } else if (resource instanceof InputStream) {
        handleStreamResource(resource);
    }
}
```

To Develop a Custom Protocol, *Daytime*

1. Develop a **URLStreamHandler** subclass for the protocol.
2. Either develop and register a **URLStreamHandlerFactory** implementation for the protocol, or name the **URLStreamHandler** in a way that the URL API can find it.
3. Develop a **URLConnection** subclass for the protocol.
4. Optional, develop a **ContentHandler** subclass for the protocol.
5. Optional, either develop and register a **ContentHandlerFactory** implementation for the protocol, or name the **ContentHandler** in a way that the URL API can find it.

Step 2, There is no URLStreamHandlerFactory

- The system property `java.protocol.handler.pkgs` is set to `se.kth.id2212.lecture7.daytime.handlers`
- The stream handler shall be in a subpackage named after the protocol, daytime, and is called Handler

Step 1, URLStreamHandler

All that is required from a **URLStreamHandler**, is that it creates a **URLConnection** subclass.

```
public class Handler extends URLStreamHandler {  
    /** Creates a <code>URLConnection</code> that can  
    @Override  
    protected URLConnection openConnection(URL url) {  
        return new DaytimeConnection(url);  
    }  
}
```


Step 2, There is no URLStreamHandlerFactory

1. The system property `java.protocol.handler.pkgs` is set to `se.kth.id2212.lecture7.daytime.handlers`
2. The stream handler shall be in a subpackage named after the protocol, `daytime`, and be called `Handler`

Step 3, URLConnection

The **URLConnection** communicates with the server, and knows the MIME type of the response.

```
public class DaytimeConnection extends URLConnection {

    private static final int PORT_NO = 13;
    private static final String MIME_TYPE = "daytime/prs.daytime";
    private Socket theConnection;

    /** Constructs an instance that can connect to the daytime ser
    public DaytimeConnection(URL url) {
        super(url);
    }

    /**...3 lines */
    @Override
    public void connect() throws IOException {
        if (!connected) {
            theConnection = new Socket(url.getHost(), PORT_NO);
        }
    }

    /**...3 lines */
    @Override
    public InputStream getInputStream() throws IOException {
        if (!connected) {
            connect();
        }
        return theConnection.getInputStream();
    }

    /**...3 lines */
    @Override
    public String getContentType() {
        return MIME_TYPE;
    }
}
```

Step 4, ContentHandler

The **ContentHandler** creates an appropriate Java object, based on the server's response.

```
public class prs_daytime extends ContentHandler {  
  
    /**...8 lines */  
    @Override  
    public Object getContent(URLConnection urlc) throws IOException {  
        BufferedReader content = new BufferedReader(new InputStreamReader(  
            (InputStream) urlc.getInputStream()));  
        StringBuilder msg = new StringBuilder();  
        String lastPart = null;  
        while ((lastPart = content.readLine()) != null) {  
            msg.append(lastPart);  
        }  
        return msg.toString();  
    }  
}
```

Step 5, There is no ContentHandlerFactory

1. The system property `java.content.handler.pkgs` is set to `se.kth.id2212.lecture7.daytime.handlers`
2. The content handler shall be in a subpackage named after the major part of the mime type, `daytime`, and be called as the minor part of the mime type `prs_daytime`
 - The dot in the minor part (`prs.daytime`) is converted to an underscore (`prs_daytime`)

Using the daytime protocol implementation

When the protocol implementation is completed, only one line of code is required to get the current time.

```
Object daytimeResponse =  
    new URL("daytime://" + DAYTIME_SERVER).getContent();
```