**1. INTRODUCTION**

A HttpURLConnection to download file from an HTTP URL can be used when User want to automatically download file using URL i.e Offline.

For Example: If User want to download pdf File . First user need to copy the address (link) from a particular Website (mysql.org) in that user should search the download link in order to get full .extension link to download from browser and then user should paste the link in program.

So when the user runs the Program, File will be automatically download in a mentioned directory [C, D, E] and User will get output saying File Download Successfully.

User can also specify his own name for the File, So when the file gets downloaded it will give new name for the particular File.

These can be used to download any size project (Images ,Applications ,Music ) etc.

In Java, user can use the classes URL and HttpURLConnection in the package java.net to programmatically download a file from a given URL by following these steps:

URL object for a given URL. The URL Should be : A direct link which contains the real file name at the end .

For example: *http://jdbc.postgresql.org/download/postgresql-9.2-1002.jdbc4.jar*

Open connection on the URL object – which would return an HttpURLConnection object if the URL is an HTTP URL.

Open the input stream of the opened connection.

Create an output stream to save file to disk.

Repeatedly read array of bytes from the input stream and write them to the output stream, until the input stream is empty.

Close the input stream, the output stream and the connection.

In the static method downloadFile(), we have to check HTTP response code from the server to make sure the URL is available (HTTP status 200). Then we extract the file name either from the HTTP header Content-Disposition or from the URL itself (in case the URL is the direct link). We also print out some debugging information like Content-Type, Content-Disposition, Content-Length and file name.

This program downloads the file postgresql-9.2-1002.jdbc4.jar and saves it into the directory C:/Download. It would produce the following output:

Content-Type = application/java-archive

Content-Disposition = null

Content-Length = 579785

fileName = postgresql-9.2-1002.jdbc4.jar

File downloaded

**2 ENVIRONMENT SPECIFICATION**

**2.1 HARDWARE REQUIREMENTS**

|  |
| --- |
| * **HARD DISK**  **:** 1.2 GB * **MONITOR :** SVGA/VGA/CGA * **RAM**  **:** 4 GB * **PROCESSOR**  **:** AMD A6 |

**SOFTWARE REQUIREMENTS**

|  |
| --- |
| * **OPERATING SYSTEM** : WINDOWS 7 * **FRONT END** : ECLIPSE * **CODING LANGUAGE** : JAVA |

* 1. **SYSTEM ANALYSIS**

System analysis is the process of gathering and interpreting facts, diagnosing problems, and using the information to recommend improvements to the system. This is the job of the systems analyst.

System analysis is a problem-solving technique that decomposes a system into its component pieces for the purpose of studying how well those component parts work and interact to accomplish their purpose.

Systems analysis is defined as those development phases in a project that primarily focus on the business problem, independent of any technology that can or will be used to implement a solution to that problem.

**Functional Requirements:**

In [Software engineering](https://en.wikipedia.org/wiki/Software_engineering) and [systems engineering](https://en.wikipedia.org/wiki/Systems_engineering), a functional requirement defines a function of a [system](https://en.wikipedia.org/wiki/System) or its component. A function is described as a set of inputs, the behavior, and outputs.

Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in [use cases](https://en.wikipedia.org/wiki/Use_case). Functional requirements are supported by [non-functional requirements](https://en.wikipedia.org/wiki/Non-functional_requirement) (also known as quality requirements), which impose constraints on the design or implementation (such as performance requirements, security, or reliability). Generally, functional requirements are expressed in the form "system must do <requirement>", while non-functional requirements are "system shall be <requirement>". The plan for implementing functional requirements is detailed in the system design. The plan for implementing non-functional requirements is detailed in the system architecture.

As defined in requirements engineering, functional requirements specify particular results of a system. This should be contrasted with non-functional requirements which specify overall characteristics such as cost and reliability. Functional requirements drive the application architecture of a system, while non-functional requirements drive the technical architecture of a system.

In some cases a requirements analyst generates use cases after gathering and validating a set of functional requirements. The hierarchy of functional requirements is: user/stakeholder request → feature → use case → business rule. Each use case illustrates behavioral scenarios through one or more functional requirements. Often, though, an analyst will begin by eliciting a set of use cases, from which the analyst can derive the functional requirements that must be implemented to allow a user to perform each use case.

* 1. **SYSTEM DESIGN**

**4.1 USE CASE DIAGRAM**

Download File interface

Search File URL



Save URL to program



Run the program and Downaload File



File Downloaded Successfully

User

* 1. **I/O Design**
  2. **SCREENSHOTS OF OUTPUT**
  3. **PROJECT DESCRIPTION**

In Java, user can use the classes URL and HttpURLConnection in the package java.net to programmatically download a file from a given URL by following these steps:

URL object for a given URL. The URL Should be : A direct link which contains the real file name at the end .

For example: http://jdbc.postgresql.org/download/postgresql-9.2-1002.jdbc4.jar

Open connection on the URL object – which would return an HttpURLConnection object if the URL is an HTTP URL.

Open the input stream of the opened connection.

Create an output stream to save file to disk.

Repeatedly read array of bytes from the input stream and write them to the output stream, until the input stream is empty.

Close the input stream, the output stream and the connection.

In the static method downloadFile(), we have to check HTTP response code from the server to make sure the URL is available (HTTP status 200). Then we extract the file name either from the HTTP header Content-Disposition or from the URL itself (in case the URL is the direct link). We also print out some debugging information like Content-Type, Content-Disposition, Content-Length and file name.

This program downloads the file postgresql-9.2-1002.jdbc4.jar and saves it into the directory C:/Download. It would produce the following output:

Content-Type = application/java-archive

Content-Disposition = null

Content-Length = 579785

fileName = postgresql-9.2-1002.jdbc4.jar

File downloaded

It downloads the entire application from a webserver as fast as it can.

The purpose of this java application, as you might guess from its name , is to Stress-test the web server.

java.net.URL and java.net.HttpURLConnection :

Java provides a HTTP client API to access resources via the HTTP or HTTPS protocol. The main classes to access the Internet are the java.net.URL class and the java.net.HttpURLConnection class.

The URL class can be used to define a pointer to a web resource while the HttpURLConnection class can be used to access a web resource.

HttpURLConnection allows you to create an InputStream.

Once you have accessed an InputStream you can read it similarly to an InputStream from a local file.

* 1. **FUTURE ENHANCEMENT**

It is not possible to develop a system that meets all the requirements of the user. User requirements keep changing as the system is being used. Some of the future enhancements that can be done to this system are:

* As the technology emerges, it is possible to upgrade the system that can be adaptable to desired environment.
* More User Friendly.
* Able to Download any file offline by giving Url.
  1. **CONCLUSION**

I have used URL and HttpURLConnection in the package java.net to programmatically download a file from a given URL.

If User want to download any File . First user need to copy the address (link) from a particular Website (mysql.org) in that user should search the download link in order to get full .extension link to download from browser and then user should paste the link in program.

So when the user runs the Program, File will be automatically download in a mentioned directory [C, D, E] and User will get output saying File Download Successfully.

User can also specify his own name for the File, So when the file gets downloaded it will give new name for the particular File.

These can be used to download any size project (Images ,Applications ,Music ) etc.

This program downloads the file postgresql-9.2-1002.jdbc4.jar and saves it into the directory C:/Download. It would produce the following output:

Content-Type = application/java-archive

Content-Disposition = null

Content-Length = 579785

fileName = postgresql-9.2-1002.jdbc4.jar

File downloaded

* 1. **BIBLIOGRAPHY AND REFERENCES**

For completion of this project we have referred the following websites namely:

* http://www.codejava.net/java-se/networking/use-httpurlconnection-to-download-file-from-an-http-url
* http://www.chillyfacts.com/java-download-file-url/
  1. **APPENDIX**
  2. **SOURCE CODE:**

HttpDownloadUtility:

package net.codejava.networking;

import java.io.File;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.net.HttpURLConnection;

import java.net.URL;

public class HttpDownloadUtility {

private static final int BUFFER\_SIZE = 4096;

public static void downloadFile(String fileURL, String saveDir)

throws IOException {

URL url = new URL(fileURL);

HttpURLConnection httpConn = (HttpURLConnection) url.openConnection();

int responseCode = httpConn.getResponseCode();

if (responseCode == HttpURLConnection.HTTP\_OK) {

String fileName = "";

String disposition = httpConn.getHeaderField("Content-Disposition");

String contentType = httpConn.getContentType();

int contentLength = httpConn.getContentLength();

if (disposition != null) {

int index = disposition.indexOf("filename=");

if (index > 0) {

fileName = disposition.substring(index + 10,

disposition.length() - 1);

}

} else {

fileName = fileURL.substring(fileURL.lastIndexOf("/") + 1,

fileURL.length());

}

System.out.println("Content-Type = " + contentType);

System.out.println("Content-Disposition = " + disposition);

System.out.println("Content-Length = " + contentLength);

System.out.println("fileName = " + fileName);

InputStream inputStream = httpConn.getInputStream();

String saveFilePath = saveDir + File.separator + fileName;

FileOutputStream outputStream = new FileOutputStream(saveFilePath);

int bytesRead = -1;

byte[] buffer = new byte[BUFFER\_SIZE];

while ((bytesRead = inputStream.read(buffer)) != -1) {

outputStream.write(buffer, 0, bytesRead);

}

outputStream.close();

inputStream.close();

System.out.println("File downloaded Successfully");

} else {

System.out.println("No file to download. Server replied HTTP code: " + responseCode);

}

httpConn.disconnect();

}

}

HttpDownloader.java

package net.codejava.networking;

import java.io.IOException;

public class HttpDownloader {

public static void main(String[] args) {

String fileURL = "https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.46.zip";

String saveDir = "C:/Download";

try {

HttpDownloadUtility.downloadFile(fileURL, saveDir);

} catch (IOException ex) {

ex.printStackTrace();}}}