# Developing a Download Manager

## What is a Download Manager?

Download manager is a tool to download files. A download manager has a capacity to manage multiple downloads at once. Every browser comes with a built-in download manager. Whenever you click on a link on some website then if it is a link to a file that is not supposed to be viewed but to be downloaded then the browser adds the new download to its download manager. There are some download managers in the market which can be added to browsers as extensions. These extensions take controls of all downloads and whenever a download link is encountered it downloads the file for the browser.

#### So, Browsers has a built-in download manager then why do we need another one?

The built-in download managers are very basic. The browser’s main job is to connect the user to the server and view webpages but not to download files. These dedicated download managers are specifically designed to handle downloads. These download managers even download files faster than the browsers.

#### Curious how download managers download files faster?… Stay tuned I will give every detail.

Download managers like Internet Download Manager openly claims that it downloads files 5 times faster than normal downloads. There are many other claiming themselves to be better and faster. But to be honest these are hoaxes and there is no way to prove that this claim is true. The speed depends on many factors like, network throughput, bandwidth, server bandwidth cap per connection or client and the most important, whether the server allows you to download fast (there’s a lot more to this point later in the blog and will be discussed in detail).

## Getting the hands dirty

Every download has a **URL** (Uniform Resource Locator: This is what we call a link in layman’s term) associated with it. These *URLs helps the client to locate the resource on the internet*. By the end of the blog you will have an idea on how to get started with the development of your own download manager and a plan on how the download manager will process the URL and download the file.

**How the Download Manager will function…**

**Step 1**: Grab the URL and check if it's valid.

**Step 2**: Make a thread(rootThread) and start downloading the file. This thread is responsible for downloading the whole file.

**Step 3**: Get the file name, file type, file size from the URL and update the GUI to set these parameters of download. You can do this by getting the value of header fields. Header Fields are some info which are present at the top of every HTTP request and response which browser uses to know how to render the message it has received. You may find the details about these headers here: HTTP headers.

**Step 4**: Now check if there is a header field with the name: ***accept-ranges***. If the header field is present in the HTTP response then you can request the server to give you just a part of the file otherwise you can only download the file in sequence. **If the header field is not present then you can't pause the download and you have to let the rootThread run till its completion or getting cancelled or encountering an error. If the field is there then you can download the file faster than it is being downloaded now.** You don't need to follow the following steps if you didn't find the header field.

**Step 5**: Now stop the root thread.

**Step 6**: Now we have to make new Download Threads with each of size: ***(contentLength-downloaded)/8.*** So, there will be 8 new threads each downloading a part of the file. Each thread will have its own separate connection with the server, to download a part of the file.

**Step 7**: Each download thread will have 2 properties: ***startPosition*** and ***endPosition.*** *The startPosition will mark the byte from where to start downloading and endPosition will mark the byte till where you have to download.*

**Step 8**: ·While connecting to the server each thread will set the HTTP request property ***range*** as:

1. ***"bytes=" + (startPosition+downloaded) + "-" + endPosition***

2. the value of downloaded will be 0 at this moment hence the thread will be requesting all the bytes in the range: **[startPosition, endPosition]**.

**Step 9**: Now the thread connects to the server and start receiving bytes from the server and writing them to corresponding files unless paused, completed or encountered an error. The working of pause and resume are described at the end of blog.

**Step 10**: Once all the threads have completely downloaded their parts, the files are merged into one file.

## What makes this faster than other?

There is a download speed limit every server puts on every request so it can serve more requests. We pass this by making multiple connections.

## Why 8 connections why not more?

Making too many requests at once may lead to server thinking of it as DDoS attack and hence block requests. We don't exactly know how many connections can lead to this but as the IDM uses a maximum of 8 connections, I am taking it as a standard.

## How to pause the download:

- The download threads will receive the bytes from the server unless the download is paused. This clearly calls for a while loop. But what will be the condition? Well, you can use a variable to represent the status of the download. If the status is set to DOWNLOADING it will keep on running the loop otherwise break out of it.

- Now, you will break out of the loop in three cases:

  1. The download is complete(status==COMPLETED)
  2. The download is paused(status==PAUSED)
  3. The download encountered an error(status==ERROR)

- Just take necessary actions in every case.

## How to resume the download:

Set the status back to DOWNLOADING and call the method doing all the downloading task. This will start downloading by setting the HTTP request property range as:

- ***“bytes=” + (startPosition+downloaded) + “-” + endPosition***

I have implemented a similar Download Manager with a few differences. If you want to take a look at the code go to this *link*. Don’t forget to give feedback and suggestions to improve blog or the download manager, it will be much appreciated