

Computer Networks and Internet Protocol
Prof. Soumya Kanti Gosh
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 08
Application Layer – IV (HTTP, HTML, TELNET)

Hello. So, we will continue our discussion on Computer Networks and Internet Protocols. So, we are discussing primarily on Application Layer Protocols and which today we will be primarily focusing on HTTP, HTML and TELNET. Though HTML is not a protocol it is a language but it comes hence in an with HTTP. So, I thought that it will be good to talk about little brief about HTML. Most of you may be already familiar with HTML but to for the sake of completeness, we will have a quick discussion on HTML protocol, right.

So, what we have seen in previous lectures? So, it is primarily a client server model where the server is active on some system on a particular port of the system, right; like a server program is active on a particular port and the client from other system or from the same system request the server for that service, right. So, primarily, in any client this type of client server paradigm, we require this element or five tuple to be there, one is that server IP, server port, client IP, client port and this protocol number or protocol ID, right.

So, this five will be there. Any of them is different will give a unique type of connectivity. That is why, even if you open up number of pages by the same browser, from the same server, it is not the it never happens that one page, you request a page for one, like see one particular link, you click in one browser and the it comes up in the other browser, right. So, it is the, is the this five tuple decides that how the whole uniqueness of the connectivity. One thing to be remembered that this can be anywhere in the things, the overall networking paradigm takes care of the thing, right.

So, some of the, some of the communication is connection oriented, so or some sort of a TCP type of protocols are there and some of the things are connection less or UDP type of protocols are there. The beauty of this layer or the layering technique is that every layer basically concerned with the functionality of it is sphere at that layer only, right. So, rest of the thing that taken care by the underlining layer to be there like, if I have A

layer in which it do not support reliable service, but the so, I need to have a other mechanism to take care at the upper layer etcetera to have a reliability, etcetera. So, any way that is a layer to layer phenomenon the PR stock at the layer at the that phenomenon only, right.

So, if we look at our HTTP type of protocol, then what we see it is a Hypertext Transfer Protocol allows primarily allows web document to be communicated over the network. And doing so, what we get realized in a in a sense, it is the basic foundation to for the realization of this www world wide web. So, the whole World Wide Web is one of the predominant thing which is, which makes it happening is this HTTP protocol.


So, HTTP protocol that supports communication between the web browser and the web server, so here we say HTTP server and the HTTP browser, right or in other sense, HTTP browser are clients right like our Internet Explorer or Mozilla or Chrome or anything, any browser which are a primarily HTTP client and the HTTP server as the other end of the thing. It can be in the same machine, can be in the same network, can be in a different network, different machine and anything right. The only thing is that it should have some sort of connectivity between the thing.

So, a web browser is a typically known as a is referred as a HTTP server or rather HTTP server is referred to as web browser whereas HTTP client is synonymous, more synonymous with the sorry, web server is a HTTP server and the way browser is the HTTP client. And we predominantly we are having two version; one is the HTTP 1.0 that is RFC 1945 specified.


(Refer Slide Time: 04:59)

HyperText Transfer Protocol (HTTP)


- HTTP is the protocol that supports communication between web browsers and web servers.
- A "Web Server" is a HTTP server
- A "Web Browser" is a HTTP client
- Most clients/servers run version 1.1, but 1.0 is also in use.
 - RFC 1945 (HTTP 1.0)
 - RFC 2616 (HTTP 1.1)
- HTTP version 1.1 specifies a persistent connection by default.



IIT KHARAGPUR



NPTEL ONLINE
CERTIFICATION COURSES




So, if you if those who are interested, can open up the RFC and see that how this definition is there and RFC 2616 which is the HTTP 1.1, right rather HTTP version 1.1 specifies a persistent connectivity by default otherwise, the connectivity are not persistent.


(Refer Slide Time: 05:18)

HTTP – Overview

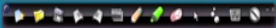
- "HTTP is an application-level protocol with the lightness and speed necessary for distributed, hypermedia information systems."
- Transport Independence
 - HTTP protocol generally takes place over a TCP connection,
 - However, the protocol itself is not dependent on a specific transport layer.



IIT KHARAGPUR



NPTEL ONLINE
CERTIFICATION COURSES



So, HTTP is a application-level protocol with the lightness and speed necessary for distributed hyper media information system. Now, one thing we need to keep in mind that in that when we are when we are exchanging information or data over across the

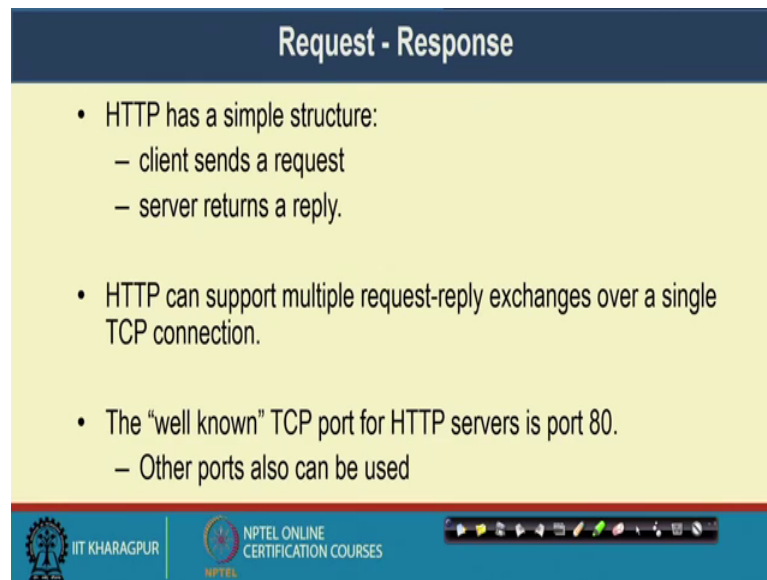
across the network, it need to be light weight so that I can have, first of all the characteristics is distributed, I heterogeneous systems are storing the data and there is a constrained on the bandwidth or the at the backbone bandwidth.

So, in order to address that, what it requires the lightness and for the speed necessary for distribution hypertext information, why this hypermedia type of information? See I should have in some generic way of representation so that it can inter operate easily, right. I do not have any control like you do not have any control that how HTML, how the data is stored in IIT Kharagpur website or a web server, right. Similarly, in your organization web server, we do not have any control or the other people have.

So, but to when I am keeping in something web server at IIT KGP, that the basic idea is to distribute the data, right. It is not for only for my consumption, but the rest of the world what I want to sow need to seed. In order to achieve these, there should be a some sort of formatting where easily which is easily perusable which we can interate, interpret or parts in a much easier way. So, this HTTP gives a basis for that which is a protocol which supports that. So, when it is transport independence; transport independence means underlining transport layer independence.

Though it is generally take place over TCP connection, HTTP comes by default with the TCP connection. However, protocol itself do not depend in on the specific transport layer when the protocol itself is not specify that this transport layer is this transport layer is mandatory etcetera, but the predominant protocol, predominant underlining protocol or by default what we consider that the there is a connection oriented or TCP type of connection is there at the transport layer, right. So, that HTTP tries to the HTTP takes care is basically work so, but this any transport layer but primarily TCP is that predominant protocol.

(Refer Slide Time: 08:02)



The slide is titled "Request - Response" in a dark blue header. The main content area is yellow and contains three bullet points. The first bullet point states that HTTP has a simple structure, with sub-points: "client sends a request" and "server returns a reply." The second bullet point states that HTTP can support multiple request-reply exchanges over a single TCP connection. The third bullet point states that the "well known" TCP port for HTTP servers is port 80, with a sub-point: "Other ports also can be used". The footer is blue and contains the IIT Kharagpur logo, the text "IIT KHARAGPUR", the NPTEL logo, the text "NPTEL ONLINE CERTIFICATION COURSES", and a navigation bar with various icons.

- HTTP has a simple structure:
 - client sends a request
 - server returns a reply.
- HTTP can support multiple request-reply exchanges over a single TCP connection.
- The “well known” TCP port for HTTP servers is port 80.
 - Other ports also can be used

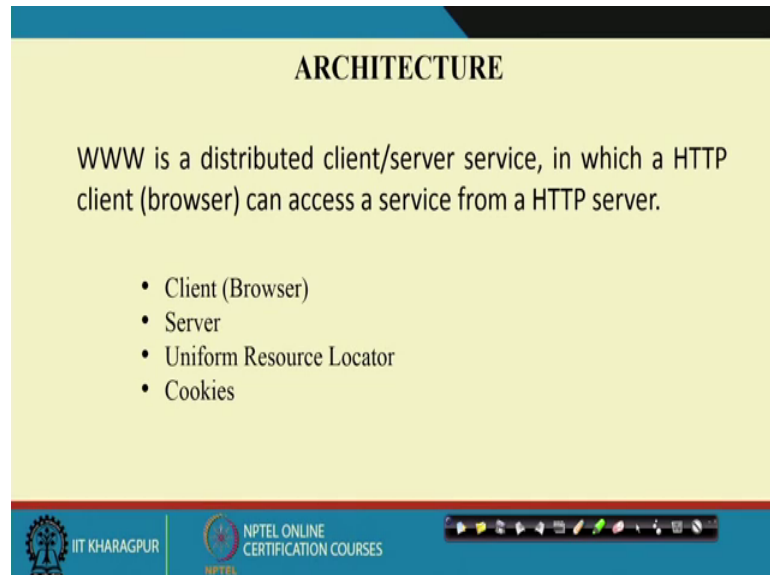
So, says a simple structure. It says a client sends a request, server returns a reply right. So, it is a very vanilla type of tracker because it was response in, HTTP can support multiple request reply exchanges over a single TCP connection right ok. If I have a underlining one transport layer connection, HTTP can support multiple request respond exchanges over a particular connection. The well known port for the HTTP is port 80, right

So, the by default if you do not specify anything, the port is port 80 right but other ports can be used. So, if you again come back to the thing like whenever you say, when we type thing into the browser say `www iit kgp dot ac dot in` or say `www npTEL` or say `npTEL dot iitm dot ac dot in` or something, `npTEL dot iit kgp ac dot in`, so what the this browser or the client does? It at the underlining, it has to because it is a name cannot be communicated.

So, the first of all the name is converted to the as a IP address, right and by default as you do not specify anything, by default it will take port 80 as the port right otherwise, you need to specify the port like you have to specify `xyz dot com colon say port 7126`; that means, you are specifying that he hit to that port. By default, as the as the HTTP server this is to port 80. So, it is the default port, other ports can also be very well used. So, overall architecture as it is a basis for www. So, distributed you look at the World Wide Web, it is a distributed client server service in which HTTP client browser can access a

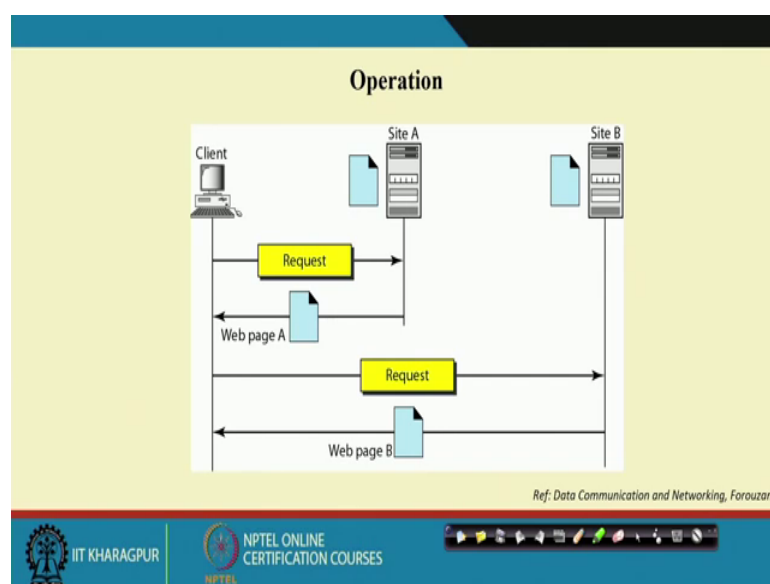
service by from a HTTP server, right. So, it is a HTTP client server service. So, there are client so, who are the parties?

(Refer Slide Time: 10:09)



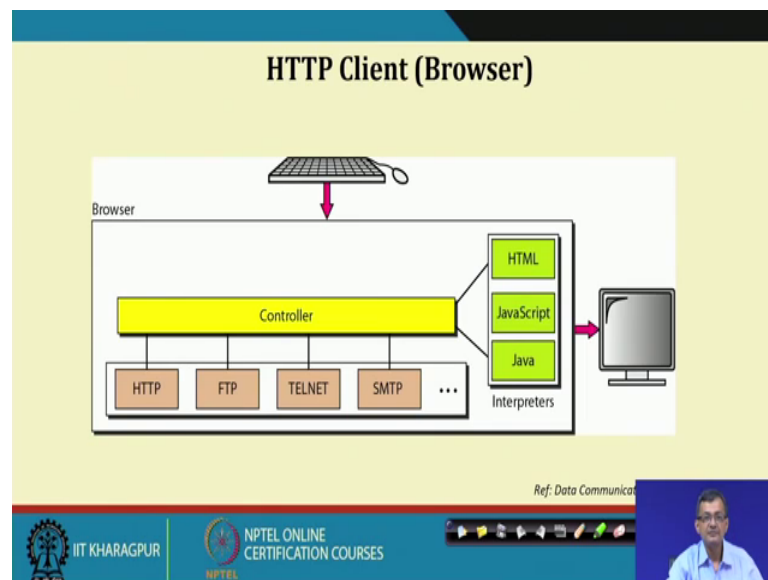
Client, Server, there is a URL or Uniform Resource Locator and there are Cookies, right. Cookies as you there is primarily to remember your previous data or it helps in a maintaining your system but primarily, what you say we require a URL for connectivity, a client or a browser for request and server for responding.

(Refer Slide Time: 10:32)



So, if you look at the thing, so that the client is sending request for a particular site, getting a web page back, it may have link for other web page connectivity, it gets this back. So, in this way it goes on referring to the web pages, right. So, this is typically or it can be any type of number of connectivity, etcetera.

(Refer Slide Time: 11:00)



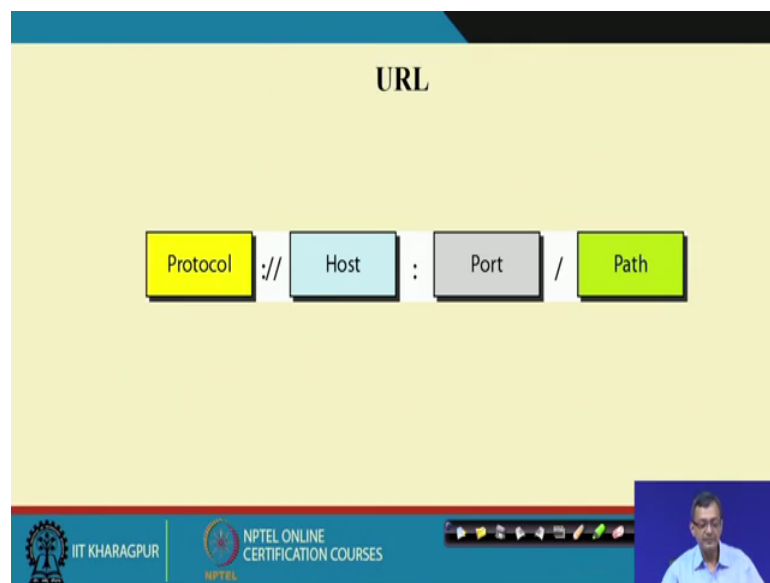
So, if we look at little bit on the browser point of view, so what at the browser what it is having? It is having a controller which is which can have different type of protocol to support right like HTTP, FTP, TELNET, SMTP and type of things and if it is the controller hits to at; if it is a normal HTML page or static page type of things, it goes and get the thing that can be JavaScript where you where you which you which is a dynamicity based things or there are other Java programs the Java is one of the thing. So, that what we say that you can, you can have program level or some sort of a API level things, right.

So, this one is that you request for a page you get the page, one you run the thing at the server site, you run something and the client site the reflection is there right; like say I send a roll number, get the rank I send a particular bank account number, gets some status report of the things. So, that can be one way of looking at that, there can be other things like some sort of a things which can be need to be checked at the client entry; like I enter a roll number; typically, say roll number is a only numeric character, numeric you

need to enter the numeric but instead if you enter a character state or say puts a character, it says that it is a invalid thing.

So, there is that that I can do at the client, no need of bringing this whole thing to the server to check it and place it back. So, that I can have a client size scripting or server site scripting those things are supported by the HTTP protocol.

(Refer Slide Time: 12:42)



And URL as we all know, it is a protocol like if it is a HTTP colon name or IP again colon port slash the path where you want to access; like if that is not by default it is a port 80, then the port is not required, right.


So, it can be HTTP, it can be FTP right; anything any protocol which support this type of thing. So it is a Unified Resource Locator or unified resource locator and we this is the overall structure, this is already known to us. If your HTTP protocol server is running in some other port, you give them that number.

(Refer Slide Time: 13:23)


WEB DOCUMENTS

Web documents can be grouped into three broad categories:



- Static
- Dynamic
- Active



IIT KHARAGPUR

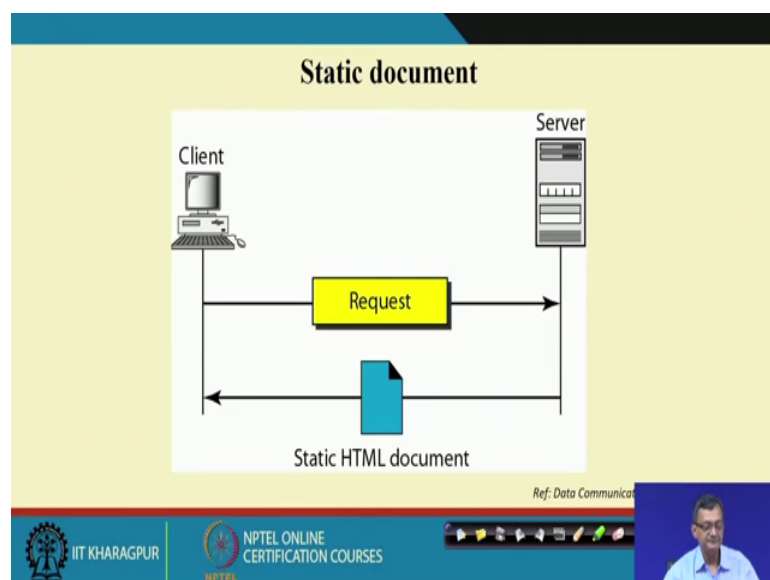


NPTEL ONLINE
CERTIFICATION COURSES

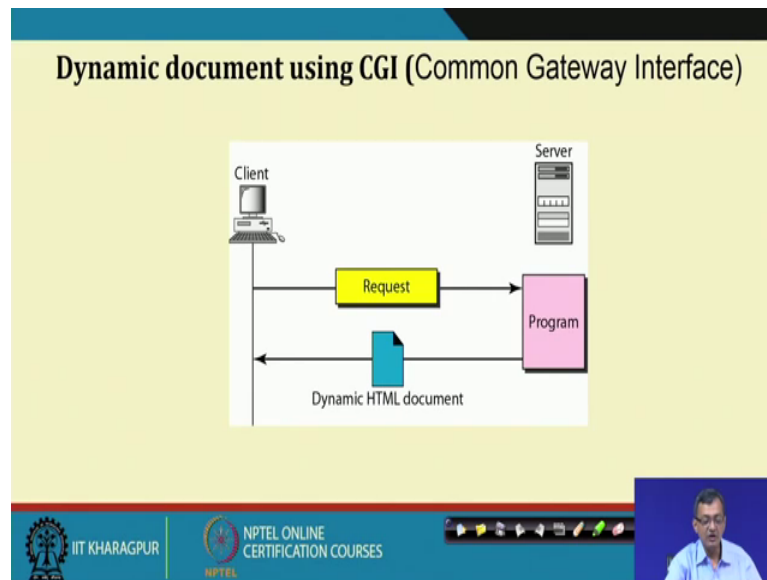


So, typically web document can grouped into three broad categories right; one is static, you request and get the page; one is dynamic, you request get something executed and get the page; one is active which is on the on your site of the browser, you value say some checking, authentication, some processing at the browser end; like static page, you request and get the static page there.

(Refer Slide Time: 13:47)



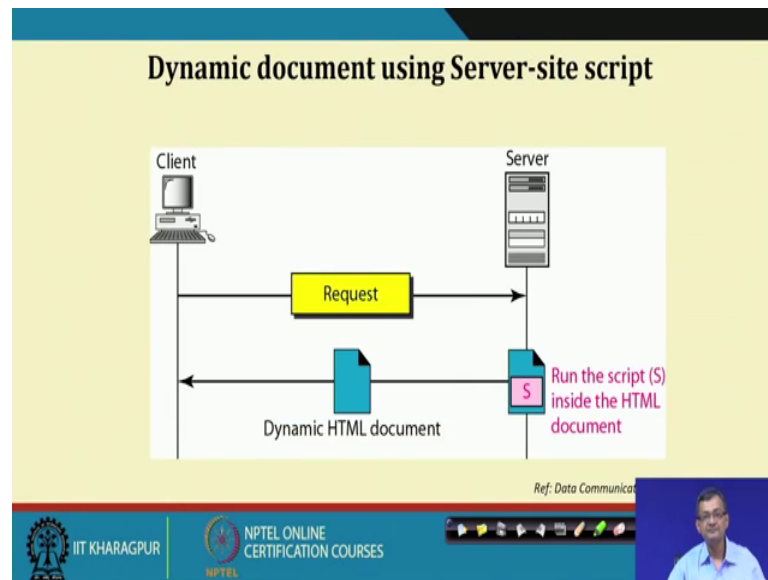
(Refer Slide Time: 13:55)



In case of like one common, in case of dynamic, we have a concept of Common Gateway Interface or CGI. So, through that I can have a request and based on that, the HTML page dynamic HTML page is written right. In the static page, say I request for a for something and I get the list of I get a static page of the list of students roll number verses the name released. In the in case of a dynamic page, I send a request with a student's roll number say and get a data related to the student roll number back to me as a HTML document right. If the student roll number changes, document also changes right.

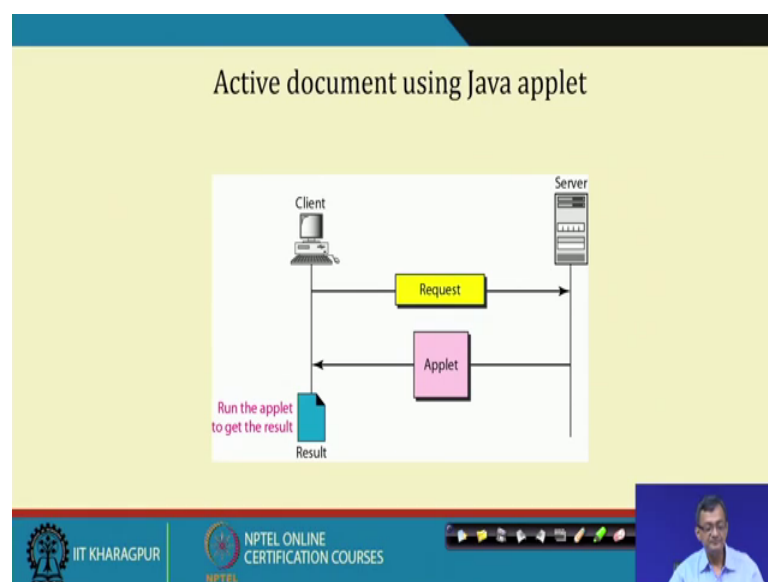
So, that is a dynamicity as well. In other sense, there should be at the program running at the other end, right. Based on your request, it execute and generate a dynamically a page and returns back. So, that is the dynamicity on the things. One of the, one of the very popular technique is using the Common Gateway Interface or CGI or CGI programming some of you might have done.

(Refer Slide Time: 15:05)



So, dynamic document also sometimes that we refer as a server site scripting right; so, the answer script is the HTML document which execute the thing and generate the generate the dynamic HTML thing. So, that is at the server end; so, server site script.

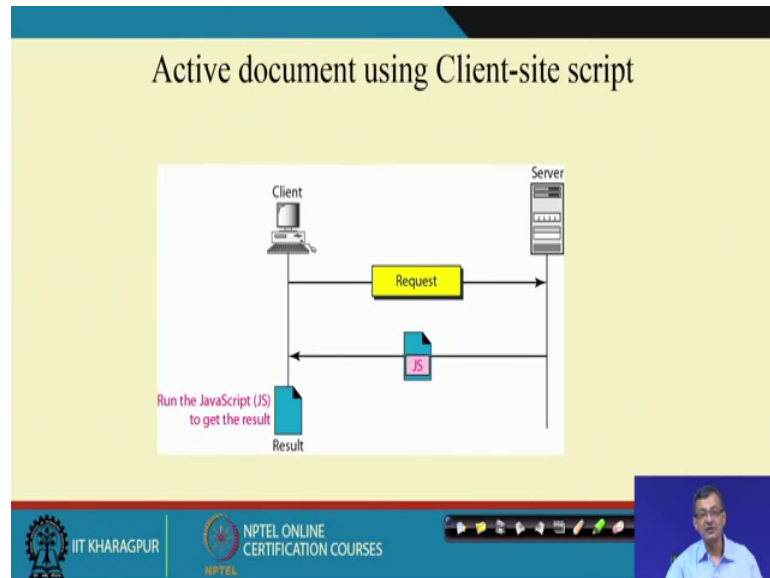
(Refer Slide Time: 15:26)



Similarly, there can be active document which are at the client side or what we say it is a client side script right. So, I can run a java applet at the client side and it gets a request, I request for the thing, it returns a applet and the applet goes on being executed at the

client side, right. So, this sort of things it is a also active document, not a static one based on your request the applet comes is returned back and it goes on executing.

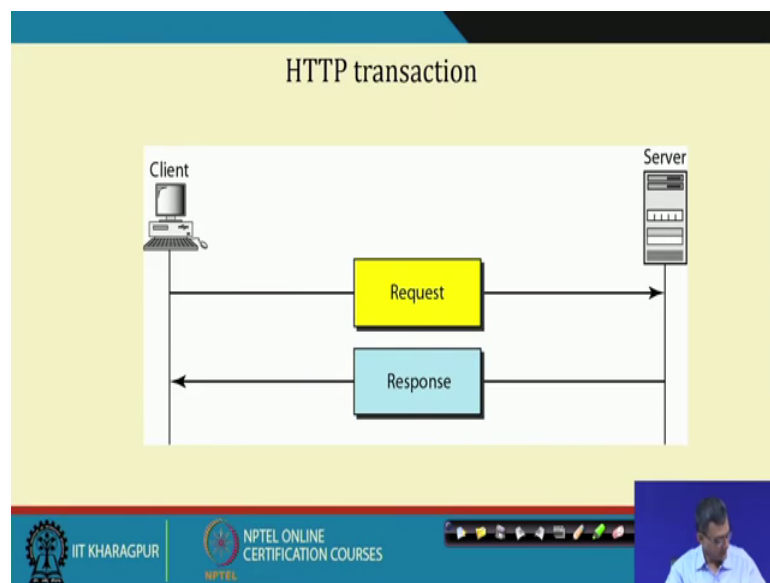
(Refer Slide Time: 16:07)



So, this is also, it is also a active page or dynamic site here or I can have a instead a JavaScript which will be executed here right. It can be used for some basic level authentication say whether it is a well client site or some sort of a those type of things which can be done at the client and without transferring the thing. So, we have this server side script which, sorry client side script and when you do that when you say that active documents, right.

So, whenever we are running on something on the server client site is active document, it is referred as active document. When you are running something at the server site and the dynamic stable comes where dynamic and if it is nothing is there, you request for a page, get the output and get the page displayed on the screen and get a static page and we have a static document.

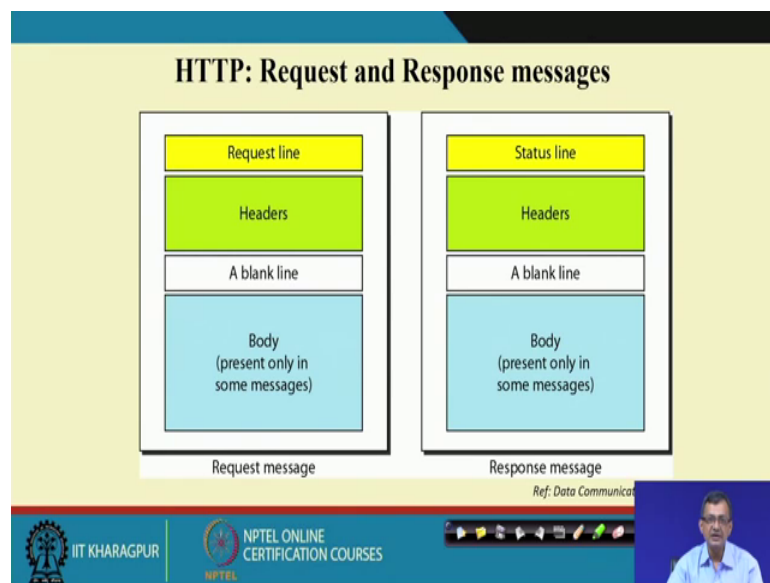
(Refer Slide Time: 17:05)



So, coming back to the thing, so what we have HTTP is a request response thing. So, again, let me little bit repeat it. So, server is running a HTTP server, typically we refer you as HTTPD, HTTP demon right. It is running at the server. Client is the sending a HTTP client request right. In our cases, it is mostly the Browser.

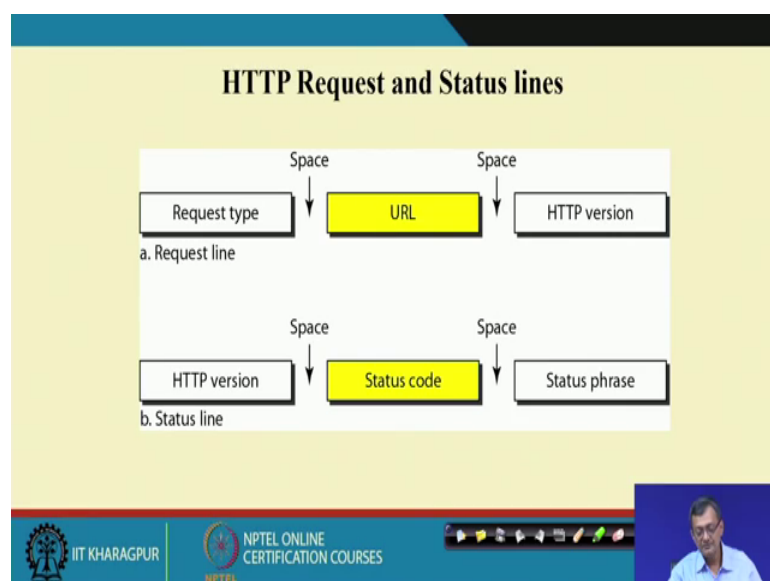
So, browser constant that it has send it to the and the server response back to that. So, that transaction is pretty simple. Every request get a response to the thing. Typically, this is typically HTTP is a request response as stateless that it is not remember what it happened in the earlier state, right. In order to do that, we need to do some other things to handle that. Anyway, though what at the present we send a request that, they return back.

(Refer Slide Time: 18:09)



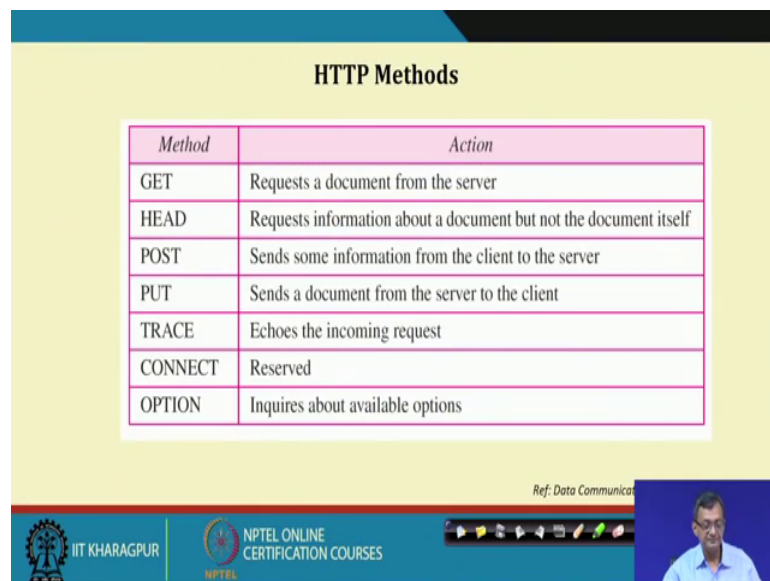
So, HTTP Request Response Messages, so what we have? This is again very a standard format. We have a Request line, a set of Headers, a blank line and the body of the thing. It may be possible that the body may not be always present in the in the message. In the in case of a response message, again it has a status line that based on the request, it is the status line say the header lines, blank line and the body of the message right. Again there can be, there may be possibilities message. So, this is the basic block of the thing but it what are the different type of request, responses or what are the different type of common status reports etcetera, that we will see slowly.

(Refer Slide Time: 19:05)



So, if you send the request and status line, so what we have that request type, the URL where the request is there and the HTTP version where which is being used. Similarly, for the status thing, we have that HTTP version, status code and the status phrase of the thing. So, particular status code like say 200 is for successful type of status code and in like that. So, it comes to the thing like if you had 404 or 4 steps of error, those are primarily error status.

(Refer Slide Time: 19:48)



Method	Action
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
POST	Sends some information from the client to the server
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request
CONNECT	Reserved
OPTION	Inquires about available options

So, there are different HTTP methods. One is say get method which request a document from the server, there is a head method requests information about a document but not the document itself right. Post method sends some information from the client to the server, put sends a document from the server to the client, trace echoes incoming request, connect is reserved method and option that inquires about the available options.

So, there are different set of methods right, we are discussing some. So, primarily more popular are Get, Head, Post, Put, these are the most popular widely used, not popular I should say other side also, there have since etcetera.

(Refer Slide Time: 20:41)

HTTP Status Codes		
Code	Phrase	Description
Informational		
100	Continue	The initial part of the request has been received, and the client may continue with its request.
101	Switching	The server is complying with a client request to switch protocols defined in the upgrade header.
Success		
200	OK	The request is successful.
201	Created	A new URL is created.
202	Accepted	The request is accepted, but it is not immediately acted upon.
204	No content	There is no content in the body.

Similarly, if you look at the HTTP status code 100 is a code of continue type of things that the initial part of the request has been received, the client may continue with the request, this is the status code. 101 is the switching the server is complying with a client request to switch protocols defined in the upgrade header, there are status code of two access series that is 200, that is OK, the request is successful; 201 that is created; 202 the request is accepted but it is immediately not immediately acted on and 204 no content; that means, there is no content in the body, right.

(Refer Slide Time: 21:29)

HTTP Status Codes (contd...)		
Code	Phrase	Description
Redirection		
301	Moved permanently	The requested URL is no longer used by the server.
302	Moved temporarily	The requested URL has moved temporarily.
304	Not modified	The document has not been modified.
Client Error		
400	Bad request	There is a syntax error in the request.
401	Unauthorized	The request lacks proper authorization.
403	Forbidden	Service is denied.
404	Not found	The document is not found.
405	Method not allowed	The method is not supported in this URL.
406	Not acceptable	The format requested is not acceptable.
Server Error		
500	Internal server error	There is an error, such as a crash, at the server site.
501	Not implemented	The action requested cannot be performed.
503	Service unavailable	The service is temporarily unavailable, but may be requested in the future.

So, there is that say that there is no content in the body. And there are more status report that is three access series; primarily moved, permanently or moved temporarily or not modified this type of things, four access series are primarily. So, three access series are redirection. So, here what we have seen, 1 xx series is informal, informational, it gives information, it is more for informational purpose; 2 xx series is mostly reporting success; 3 xx is redirection, so it is any redirection of the page or change or somewhat; 4 xx is the client error or the whatever the client sites error that bad request, unauthorized request, forbidden, not found, methods not allow allowed, not acceptable and these are state of request which are at the client error whereas, server errors are 5 xxx series that internal server error, not implemented, service unavailable. So, these are all server site error.

So, there are client site errors, server site error. So, everything has a status code and if you minutely check that whenever it keeps you whether accept the, you mostly see this sort of error by that you can decipher that; one is 404 is the most what we say mostly most seen error type of thing that the document not found type sort of error.

(Refer Slide Time: 22:59)

The slide is titled "HTTP Header". It illustrates the structure of an HTTP header as "Header name" followed by a colon, a space, and then the "Header value". Below this, a table lists several common headers and their descriptions.

Header	Description
Cache-control	Specifies information about caching
Connection	Shows whether the connection should be closed or not
Date	Shows the current date
MIME-version	Shows the MIME version used
Upgrade	Specifies the preferred communication protocol

The slide also features logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES at the bottom, along with a video player interface and a small inset video of a speaker.

So, if we look at the HTTP header, header name colon a space and the header value. So, header are different type of header like Cache control specify informational information about the cache caching; Connection shows whether the connections would be closed or not; Date, current date; MIME upgrade and so and so.

(Refer Slide Time: 23:23)

Request Headers	
Header	Description
Accept	Shows the medium format the client can accept
Accept-charset	Shows the character set the client can handle
Accept-encoding	Shows the encoding scheme the client can handle
Accept-language	Shows the language the client can accept
Authorization	Shows what permissions the client has
From	Shows the e-mail address of the user
Host	Shows the host and port number of the server
If-modified-since	Sends the document if newer than specified date
If-match	Sends the document only if it matches given tag
If-non-match	Sends the document only if it does not match given tag
If-range	Sends only the portion of the document that is missing
If-unmodified-since	Sends the document if not changed since specified date
Referrer	Specifies the URL of the linked document
User-agent	Identifies the client program

And there are this, this are mostly request type of header like this state of request type of request header. So, what we have seen request header and the response or status header. So, these are the state of the request header Accept, Accept-charset, Accept-encoding and so and so forth. So, these are more; in detail if you want to look at those who are interested look need to be looked into the any standard book or RFC maybe a good place a look at.

(Refer Slide Time: 23:55)

Response Headers	
Header	Description
Accept-range	Shows if server accepts the range requested by client
Age	Shows the age of the document
Public	Shows the supported list of methods
Retry-after	Specifies the date after which the server is available
Server	Shows the server name and version number

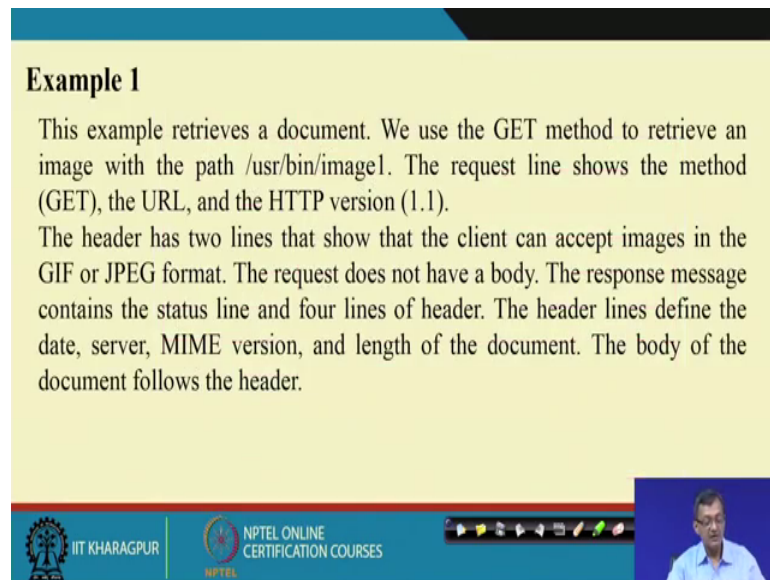
So, there are response header like Age shows the age of the document, Public shows the supported list of methods, Retry after specify the date after which the server is available, Server shows the name and server version number like that. So, these are different response headers.

(Refer Slide Time: 24:15)

HTTP: Entity headers	
Header	Description
Allow	Lists valid methods that can be used with a URL
Content-encoding	Specifies the encoding scheme
Content-language	Specifies the language
Content-length	Shows the length of the document
Content-range	Specifies the range of the document
Content-type	Specifies the medium type
Etag	Gives an entity tag
Expires	Gives the date and time when contents may change
Last-modified	Gives the date and time of the last change
Location	Specifies the location of the created or moved document

So, HTTP Entity headers, so allow say allow the list of valid methods, content encoding, so these are more related to the entity right. So, that is content length, content range, content type, when it expires last, modified date, location specifies the location of the created or modified document. So, these are the different entity headers.

(Refer Slide Time: 24:46)



Example 1

This example retrieves a document. We use the GET method to retrieve an image with the path `/usr/bin/image1`. The request line shows the method (GET), the URL, and the HTTP version (1.1).

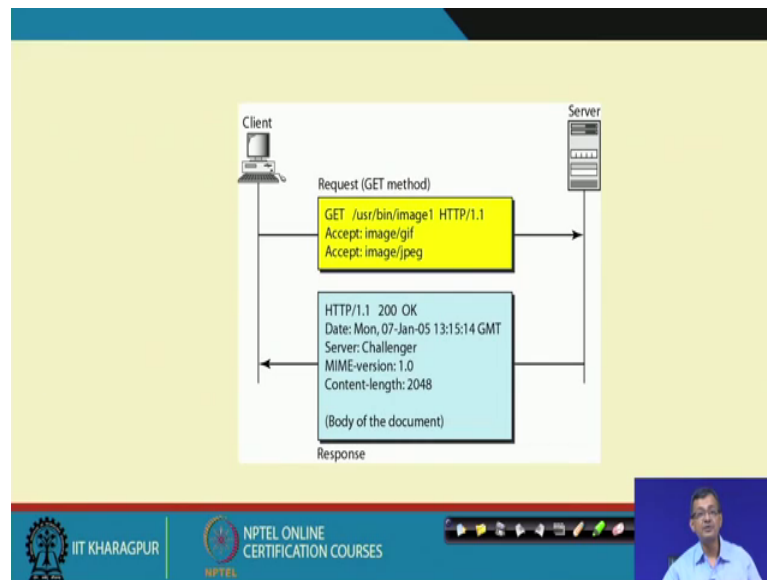
The header has two lines that show that the client can accept images in the GIF or JPEG format. The request does not have a body. The response message contains the status line and four lines of header. The header lines define the date, server, MIME version, and length of the document. The body of the document follows the header.

The slide features a blue header bar, a yellow main content area, and a blue footer bar. The footer bar contains the IIT Kharagpur logo, the NPTEL Online Certification Courses logo, a navigation bar with icons, and a small video inset of a speaker in the bottom right corner.

So, what we see, it is a rich collection or rich set of commands or control is there. So, it that is why it is able to handle a variety of say media or the variety of data or the variety of information across the across the www. And we need to also keep in mind that there are data are stored in a heterogeneous version in across the network and every data has their own payload or data load which takes time to the thing. HTTP has it is own say timing issues like how long, suppose I request a particular page, how long I should wait the page will come right. So, that is a that there is a timeout thing also is there.

So, keeping in mind so, in the overall management of this type of distributed and loosely coupled system, you need to have lot of flags and what we say so called headers and other information's to handle that. The example here, a example again from that book that example retrieves a document, we get a get method to retrieve an image with a path slash usr slash bin slash image 1. The request line shows the method get and thing.

(Refer Slide Time: 26:20)



So, if we come to the example, so this is the request to get method HTTP version 1 accept it is looking for a image and accept this image type of things. If you look at the header thing, so accept, so the medium format the client can accept, right. So, that it shows that what are the format it can accept right, it is a image of gif type or image of jpeg type and the at the other end, it gives a status of 2 200 that if you remember that the success header, then the date server is what sort of server, what sort of MIME version is there how much content length is there and the body of the document right. So, this is the way request comes. So, if you if you expand the HTTP, it will be looking like that.

(Refer Slide Time: 27:06)

Example 2

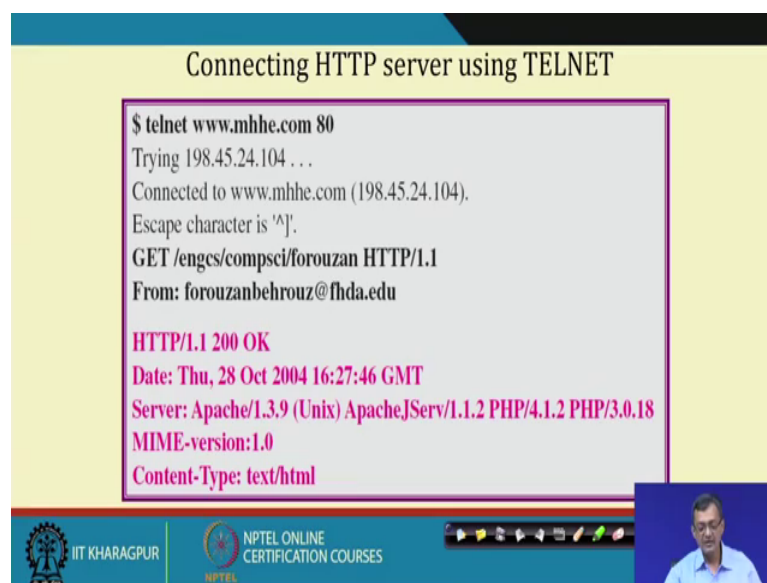
In this example, the client wants to send data to the server. Use the POST method. The request line shows the method (POST), URL, and HTTP version (1.1). There are four lines of headers. The request body contains the input information. The response message contains the status line and four lines of headers. The created document, which is a CGI document, is included as the body

The slide is part of an NPTEL presentation from IIT Kharagpur.

Similarly, in another example, so this is a request and this example it is a post, right where in this example, a client want to send a data to a server. Use post method, the request line shows the POST, URL and HTTP version 1. There are four lines of the header. The request body contains the information and so and so, what we see? So, it is a posting something against the image type, some data and the code either j for jpeg and it is content is some content length is there; on the other side, the server replies with 200 and it is the type of things it is able to handle it is right, right.

So, it is a response message contains the status line and the four lines of the header, the created document which is a CGI document that is if you remember it is a Common Gate Way Interface document and is included as the in the body of the thing, right. So, it comes as a request message as is as encoded as a CGI document for the body of the message.

(Refer Slide Time: 28:12)



```
$ telnet www.mhhe.com 80
Trying 198.45.24.104 ...
Connected to www.mhhe.com (198.45.24.104).
Escape character is '^['.
GET /engcs/compsci/forouzan HTTP/1.1
From: forouzanbehrouz@fhda.edu

HTTP/1.1 200 OK
Date: Thu, 28 Oct 2004 16:27:46 GMT
Server: Apache/1.3.9 (Unix) ApacheJServ/1.1.2 PHP/4.1.2 PHP/3.0.18
MIME-version:1.0
Content-Type: text/html
```

Now, just to look at that connecting HTTP server using I can have a TELNET also, right. TELNET will see subsequently, it is primarily used to connect to some other remote login type of situation but I can have TELNET, the name of the things it is that I can have TELNET www mhhe dot com dot at, right.

So, it access series it, it will hit and try to retrieve provided that if it is not blocked that the particular TELNET function etcetera but it that will retrieve back. So, what we mean to say that underlining, I can use some other protocol at that particular port and if the

HTTP server or the server which is listening at the port is responding to the request, it will respond right. In this case, it gets a I a after doing that I give a get message of a something, this is actually from that data communication of the Forouzan book.

So, that is the examples on there. And the server in also it responded, we 200 and so and so forth, right. So, what we see here that it is it is the server which is listening at port 80 in this case. It could have been in other case also, accept that message and respond it back to the thing and it say, I am sending an HTTP sort of request through a other protocol that is a TELNET protocol, right. So, this is feasible right.

So, what we see a HTTP is a hyper text transfer protocol where a server side that at the server is listening to a particular port and the client or client or the clients knows that which where is the IP and the port of the things. Whenever, it wants to connect to the thing, connect to that particular port and get the result and based on the things that can be different type of status right, it can be it mostly it is successful, otherwise if the error is there. So, there should be a error flagging and so and so forth.

And get displayed on the things; another point if you see, it is a it is, it can handle different type of media. It can handle text, it can handle your different version on the thing, it can handle image, it can handle video, it can handle voice. So, it is a some sort of a, it can handle any hyper media type of thing but as the as the communication or the formatting is in a particular generic format, so it is able to that at the client or the browser able to accept it.

So, that is the beauty of this HTTP which can access across the across the over the internet. So, with this, let us conclude here and will be discussing little bit on HTML and other TELNET in the subsequent lecture.

Thank you.