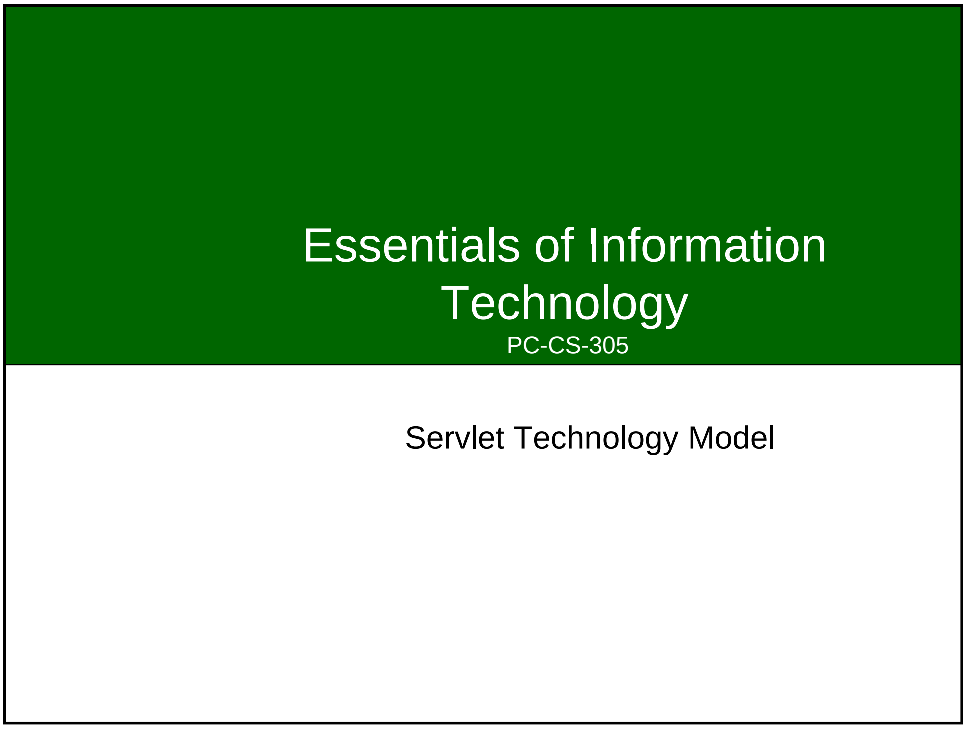
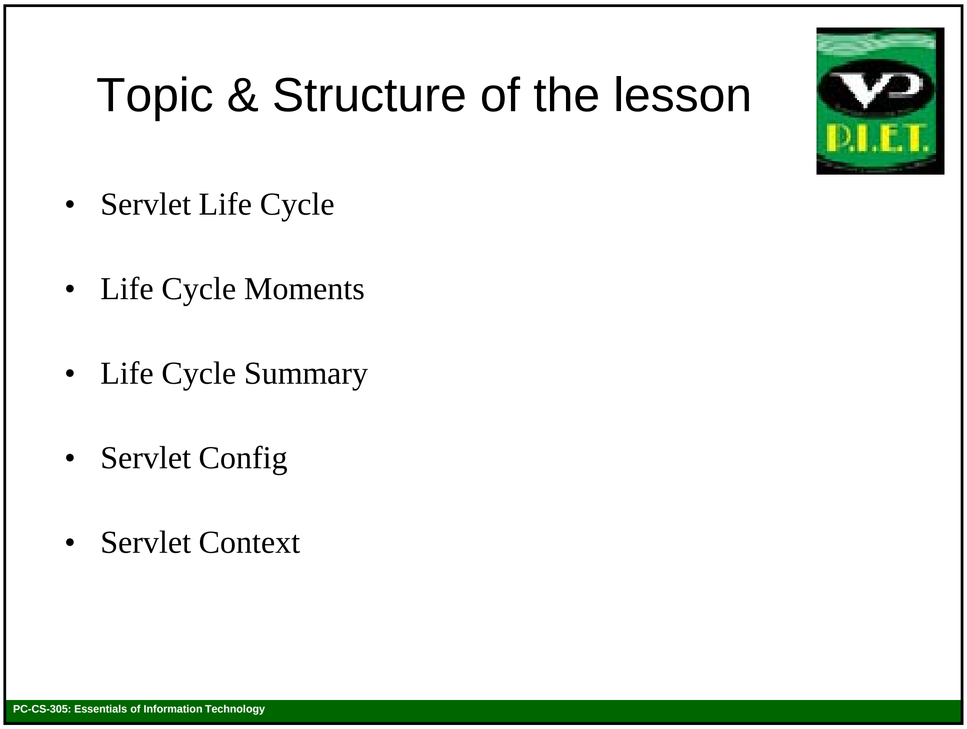


Essentials of Information Technology

PC-CS-305

Servlet Technology Model



Topic & Structure of the lesson

- Servlet Life Cycle
- Life Cycle Moments
- Life Cycle Summary
- Servlet Config
- Servlet Context

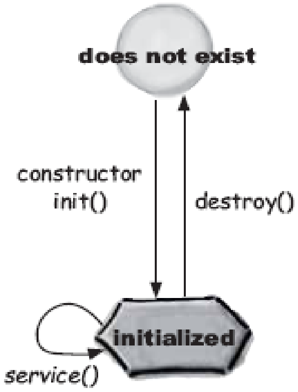
Topic & Structure of the lesson

- HTTP Methods
- Sending the Response
- Redirecting the Request
- Using Request Dispatch

Servlet Life Cycle

- The servlet life cycle is simple ,there is only one main state – initialized.
- If the servlet is not initialized,then its either
  - being initialized (running the constructor , init() method).
  - being destroyed(running its destroy() method)
  - or it simply does not exist.

Servlet Life Cycle



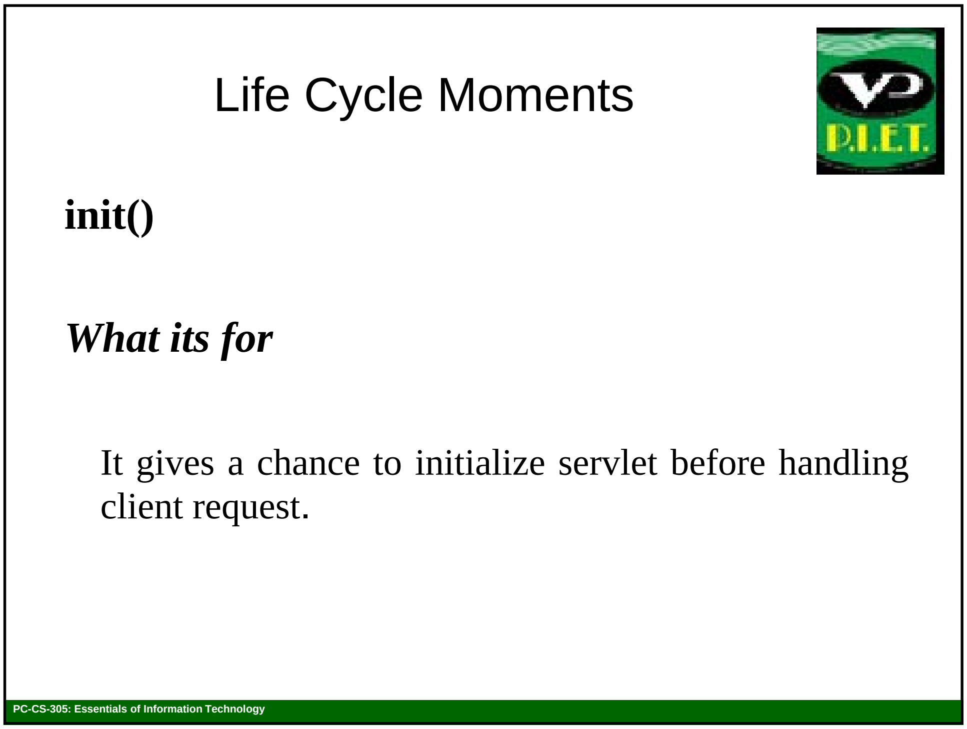
Life Cycle Moments

init()

When called

The container calls init() (GenericServlet) on servlet instance after the servlet instance is created but before servlet can service any client request.

This method gives a chance to initialize servlet before handling any client request.

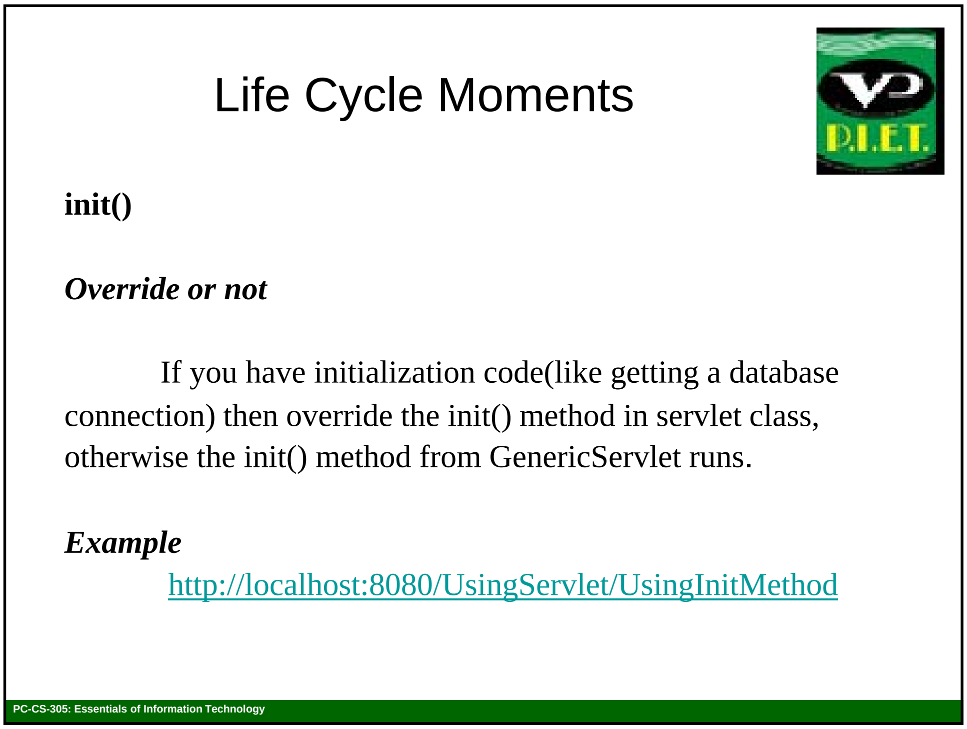


Life Cycle Moments

init()

What its for

It gives a chance to initialize servlet before handling client request.



Life Cycle Moments

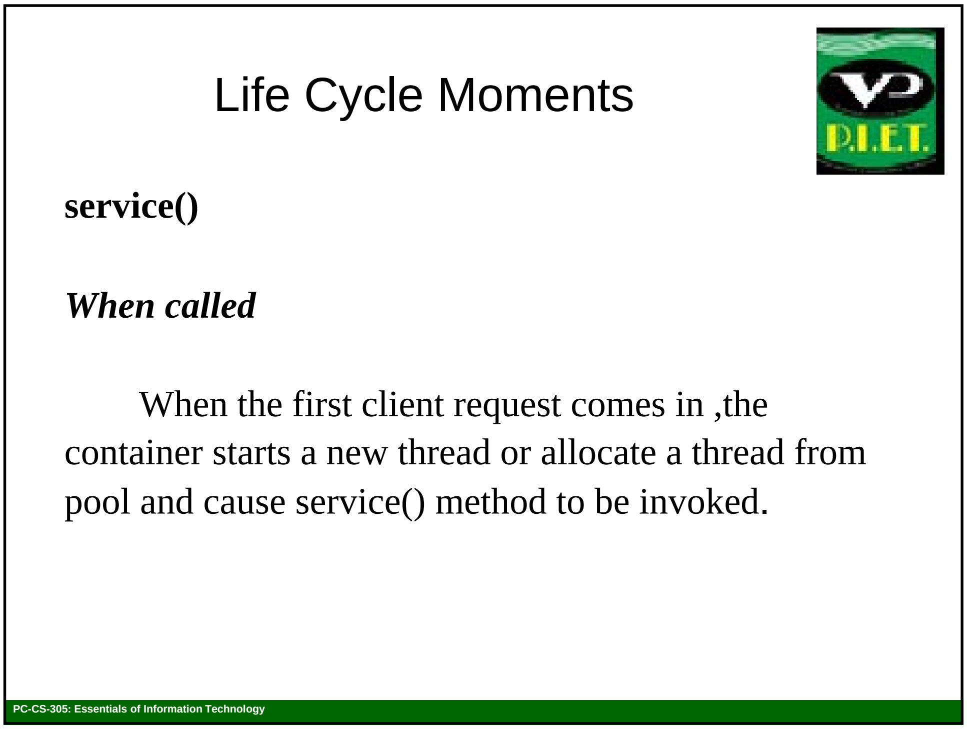
init()

Override or not

If you have initialization code(like getting a database connection) then override the init() method in servlet class, otherwise the init() method from GenericServlet runs.

Example

http://localhost:8080/UsingServlet/UsingInitMethod

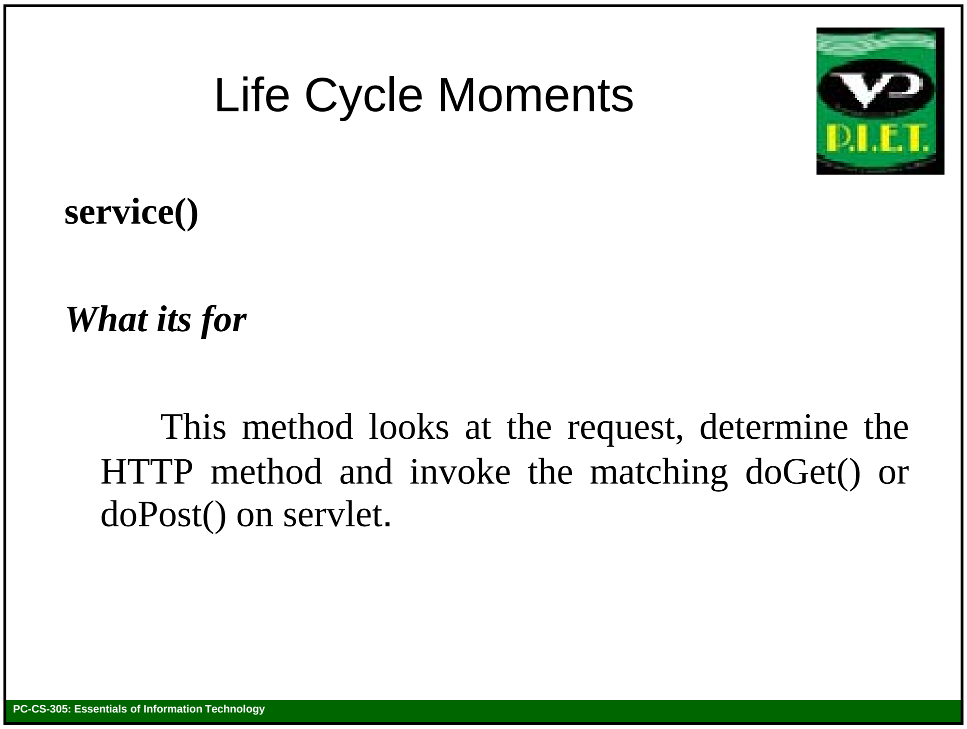


Life Cycle Moments

service()

When called

When the first client request comes in ,the container starts a new thread or allocate a thread from pool and cause service() method to be invoked.

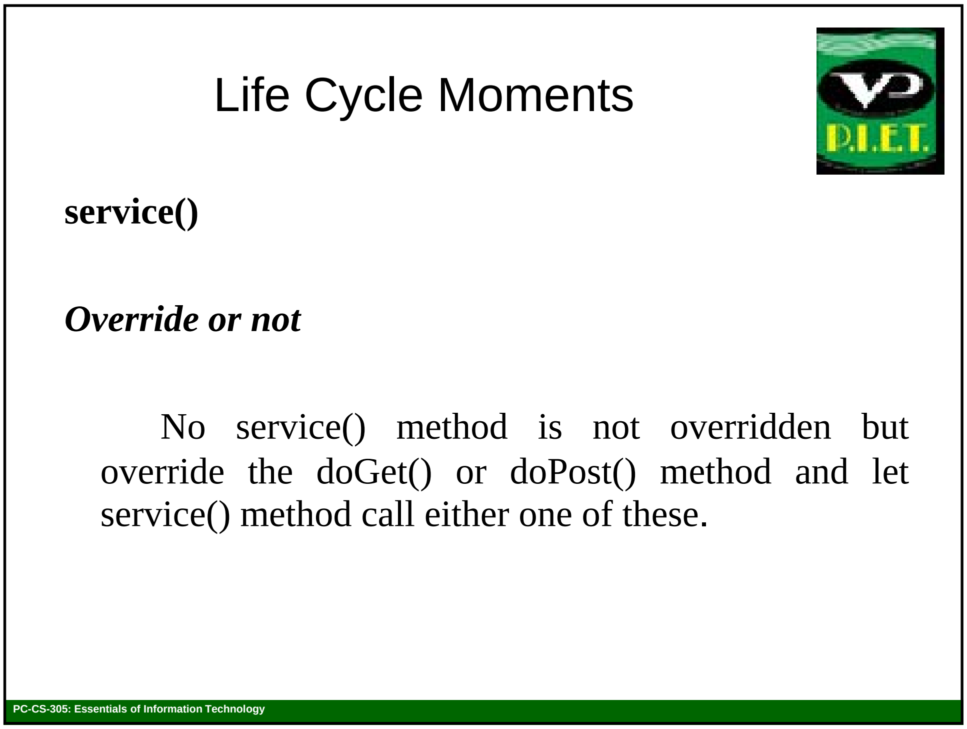


Life Cycle Moments

service()

What its for

This method looks at the request, determine the HTTP method and invoke the matching doGet() or doPost() on servlet.

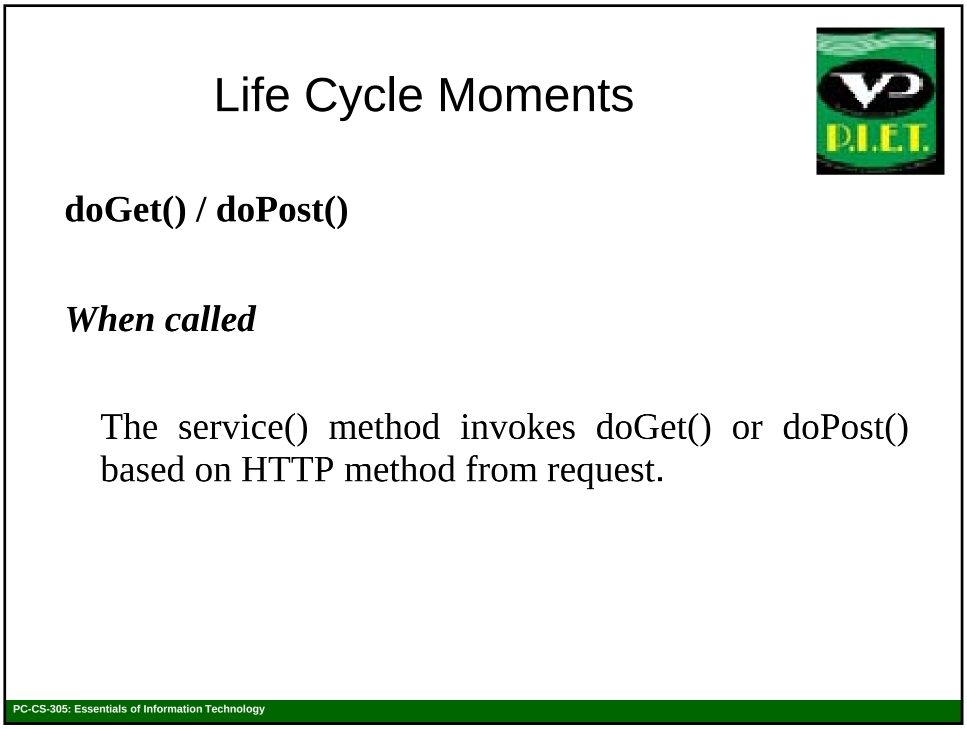


Life Cycle Moments

service()

Override or not

No service() method is not overridden but override the doGet() or doPost() method and let service() method call either one of these.



Life Cycle Moments

doGet() / doPost()

When called

The service() method invokes doGet() or doPost() based on HTTP method from request.

Life Cycle Moments

doGet() / doPost()

What its for

This method is responsible for whatever the web app is supposed to be doing.

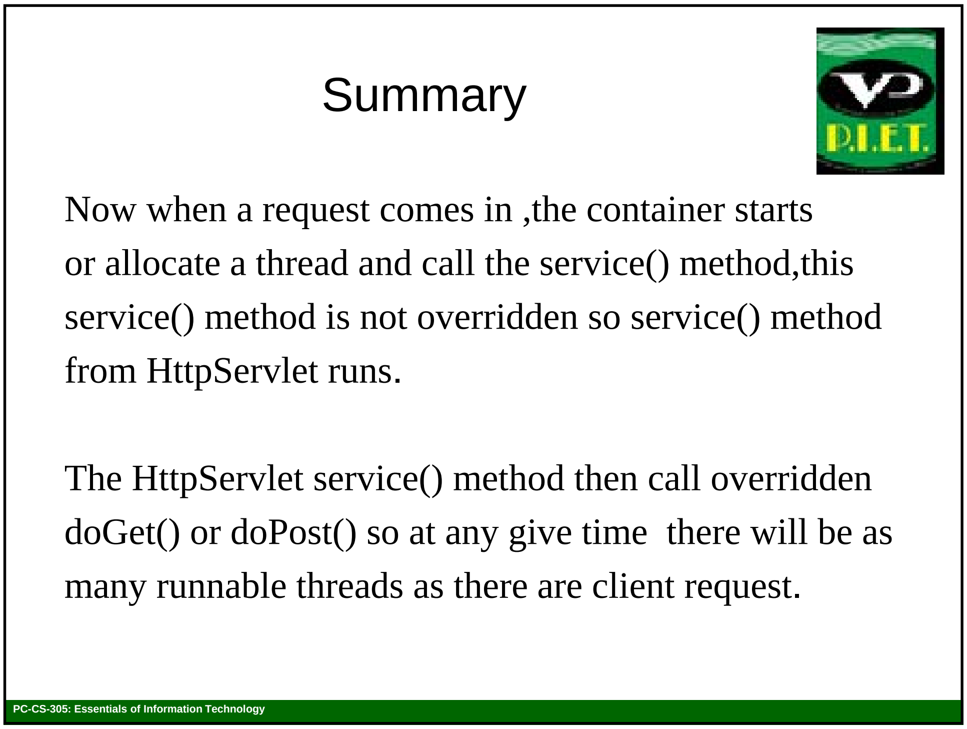
Override or not

Overridden always atleast one of them

Summary

Container calls init() method on the servlet instance before servlet can service() any client request

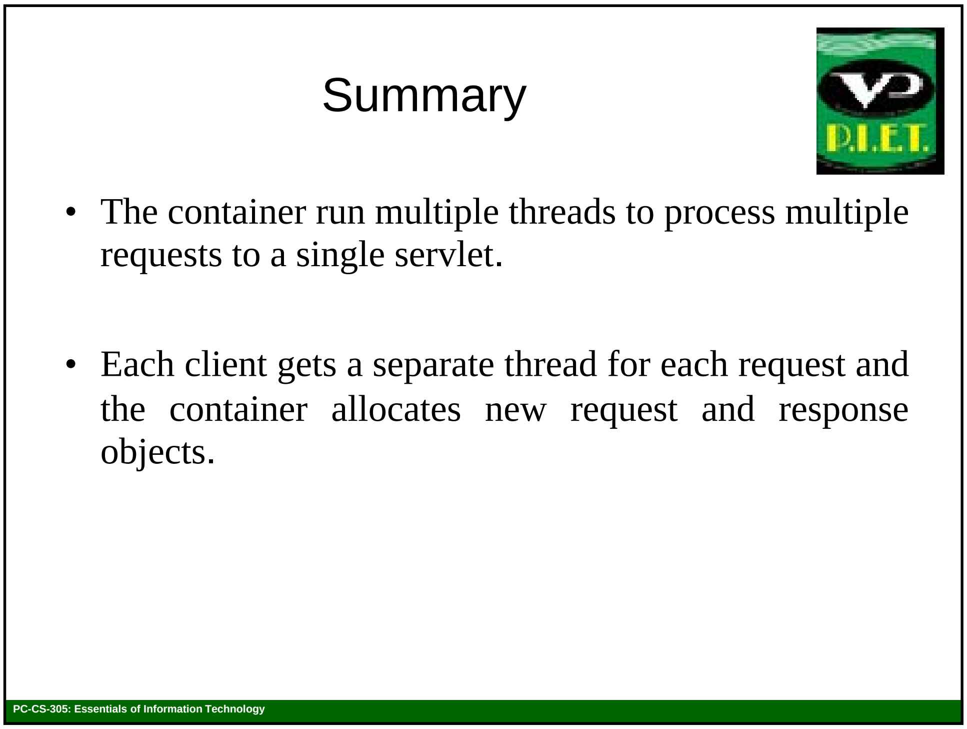
If there is some initialization code then override the init() method in the servlet class, otherwise the init() method from GenericServlet runs.



Summary

Now when a request comes in ,the container starts or allocate a thread and call the service() method,this service() method is not overridden so service() method from HttpServlet runs.

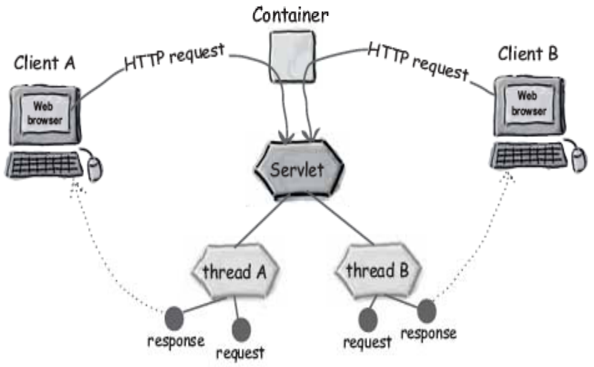
The HttpServlet service() method then call overridden doGet() or doPost() so at any give time there will be as many runnable threads as there are client request.



Summary

- The container run multiple threads to process multiple requests to a single servlet.
- Each client gets a separate thread for each request and the container allocates new request and response objects.

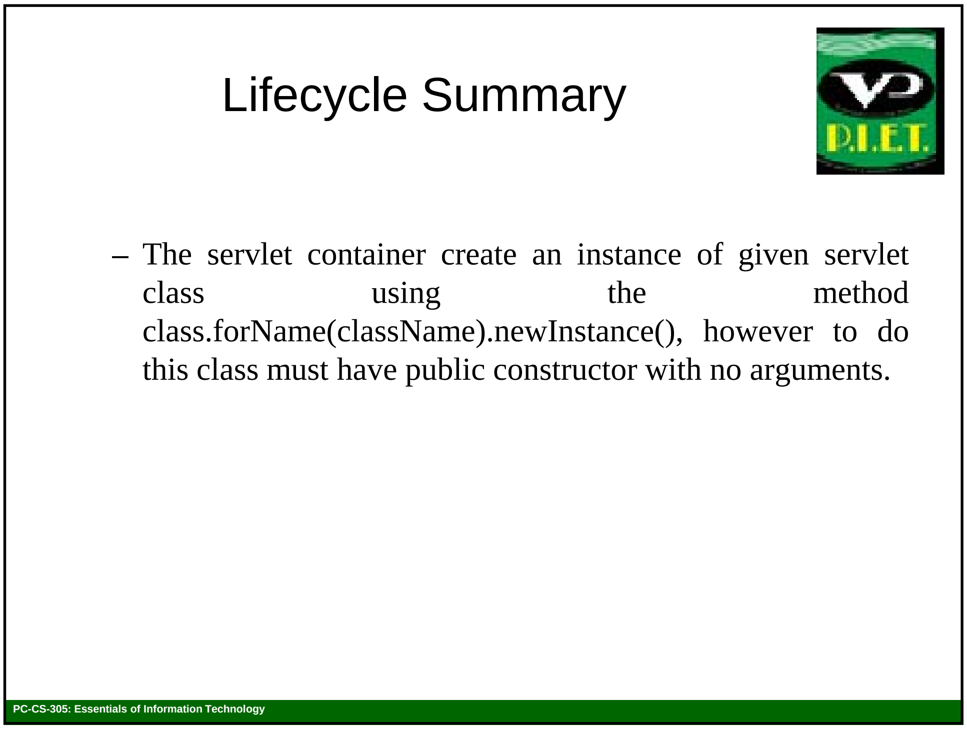
Summary



Lifecycle Summary

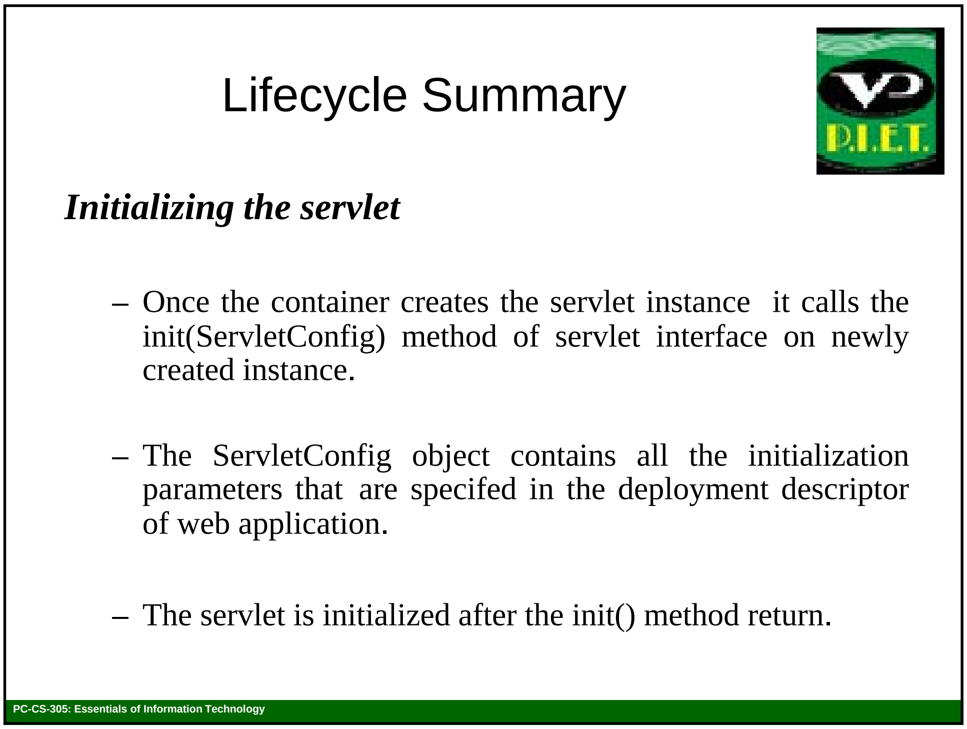
Loading and instantiating a servlet

- Servlet lifecycle start when the container startup,it look for the deployed webapp and look for set of configuration files also called deployment descriptor.
- This deployment descriptor file web.xml include an entry for each of the servlet it uses.An entry specifies the name of the servlet and Java class name of servlet.



Lifecycle Summary

- The servlet container create an instance of given servlet class using the method class.forName(className).newInstance(), however to do this class must have public constructor with no arguments.



Lifecycle Summary

Initializing the servlet

- Once the container creates the servlet instance it calls the init(ServletConfig) method of servlet interface on newly created instance.
- The ServletConfig object contains all the initialization parameters that are specifed in the deployment descriptor of web application.
- The servlet is initialized after the init() method return.

Lifecycle Summary

Preinitializing a servlet

- Container donot initalize the servlet as soon as it startup.It initialize the servlet when it receive a request for that servlet for the first time.This is called lazy loading.
- Although it improves the startup time of container but the client will have a poor response time when he send the first request.
- Define the <load-on-startup> element in the DD to make the servlet container load and initialize the servlet as soon as it startup.

Lifecycle Summary

- This process of loading a servlet before any request comes in is called preloading or preinitializing a servlet.

Servicing client request

- After instance is properly initialized its ready to service the client request,now when the container receive request it will dispatch them to servlet instance calling

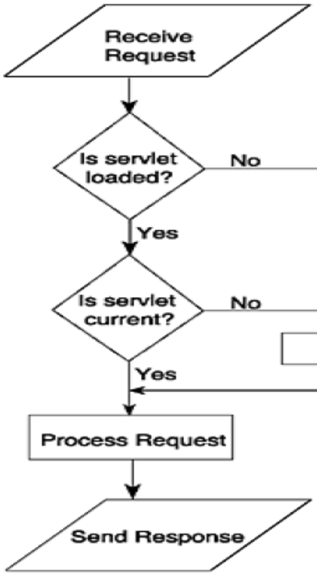
  Servlet.service(ServletRequest,ServletResponse)

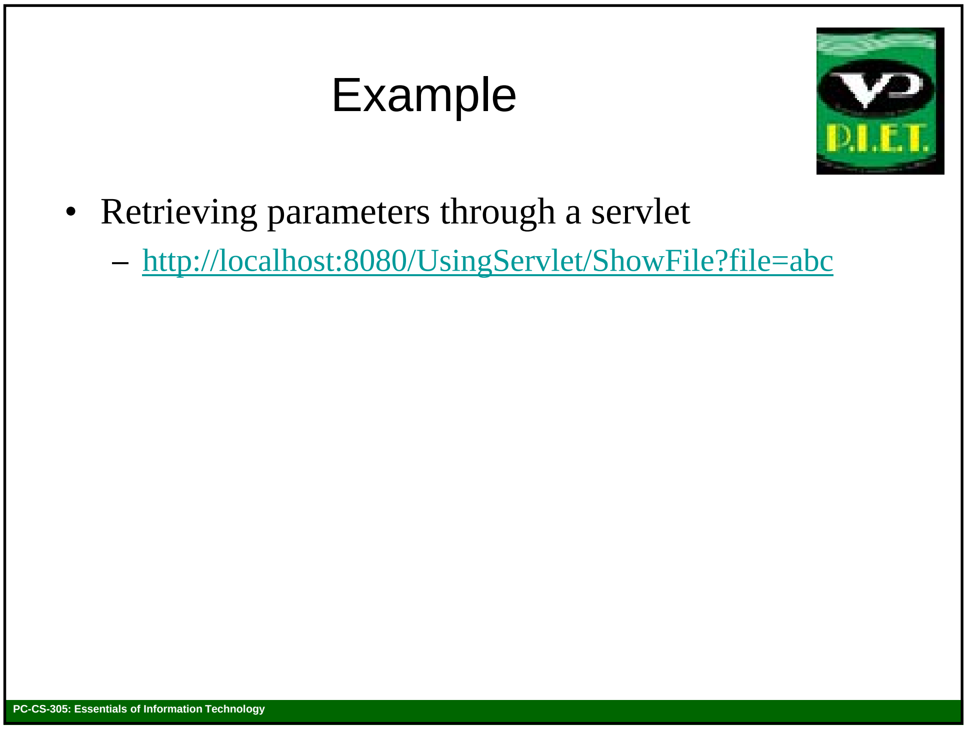
Lifecycle Summary

Destroying a servlet

If container decide it no longer need the servlet instance it call destroy method on servlet instance.Once this method is called the instance will be out of service and the container will never call the service() method on this instance.Before calling the destroy method the servlet container wait for the remaining thread executing the service() method to finish.

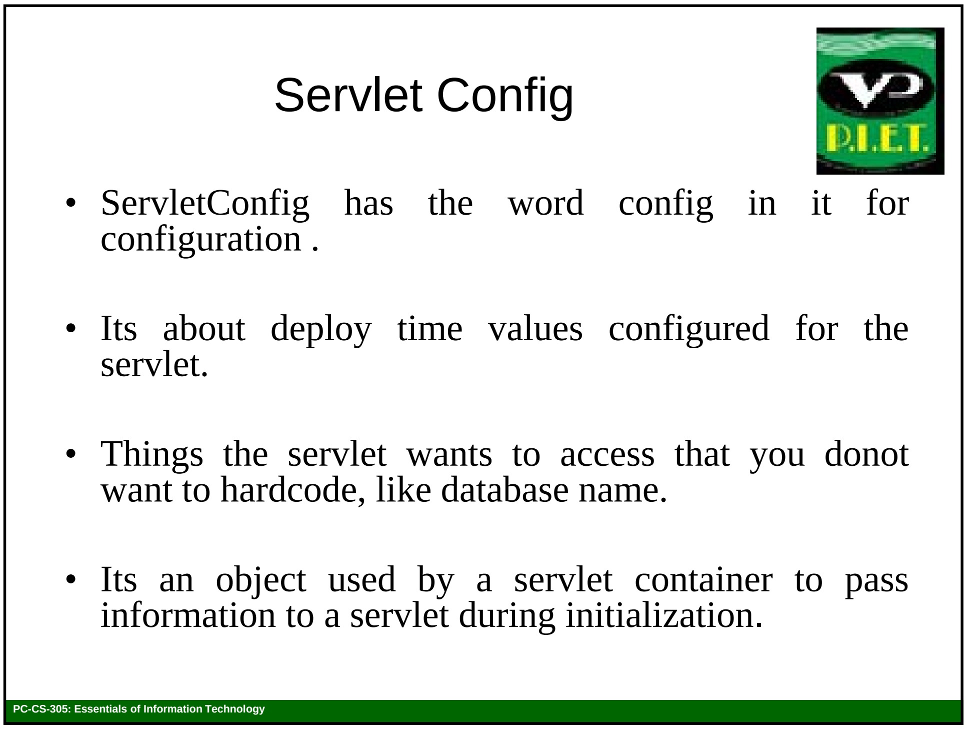
How Servlet Works





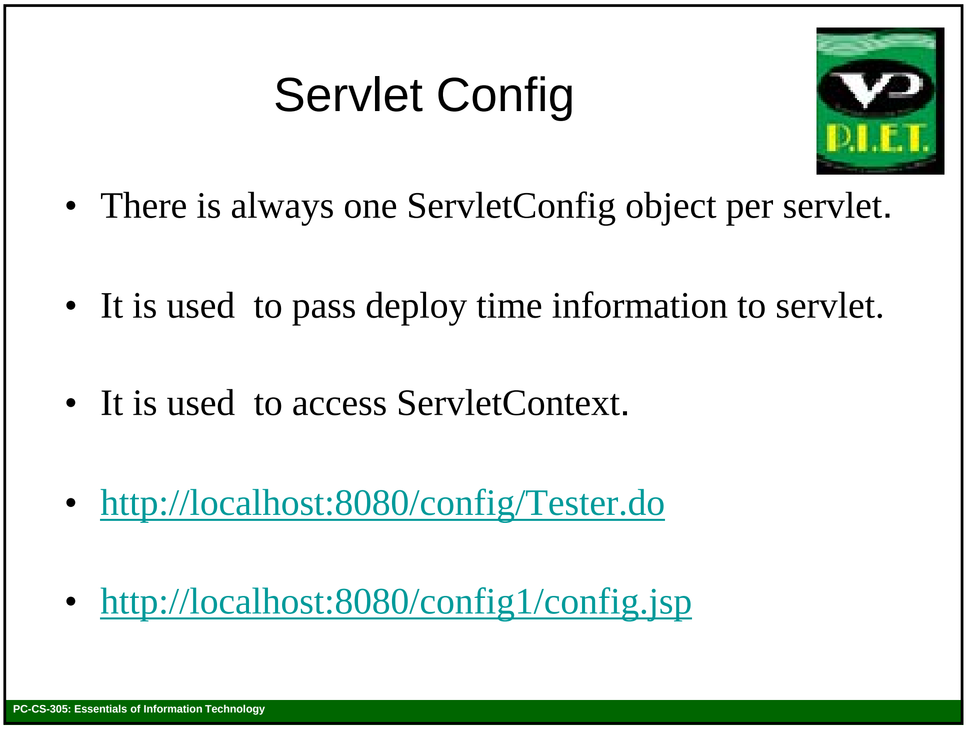
Example

- Retrieving parameters through a servlet
  - http://localhost:8080/UsingServlet/ShowFile?file=abc



Servlet Config

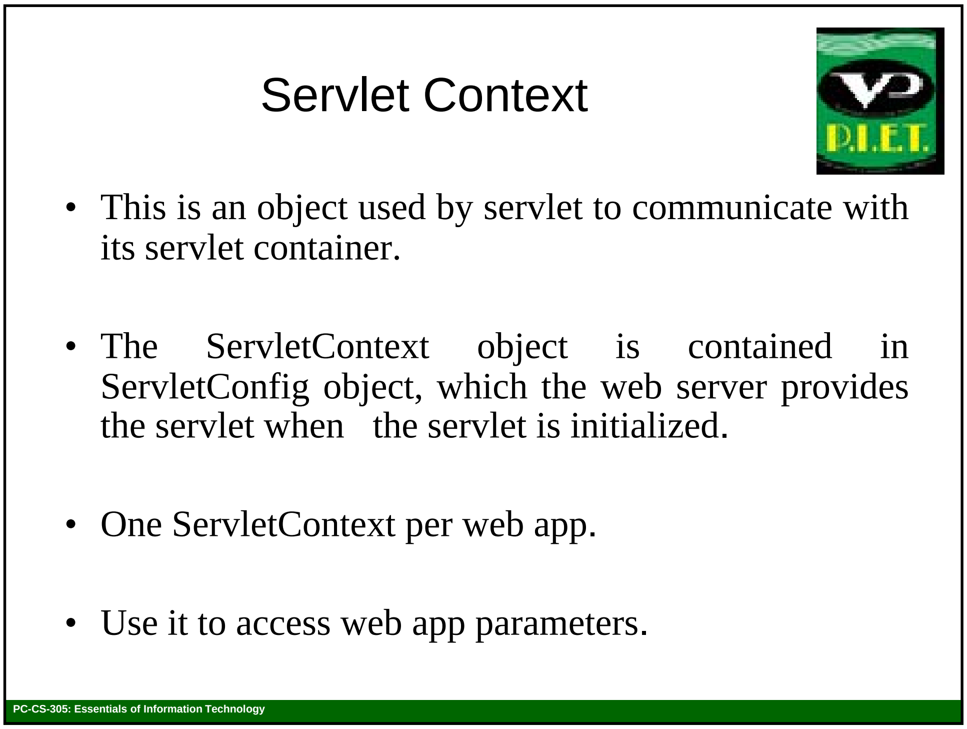
- ServletConfig has the word config in it for configuration .
- Its about deploy time values configured for the servlet.
- Things the servlet wants to access that you donot want to hardcode, like database name.
- Its an object used by a servlet container to pass information to a servlet during initialization.



Servlet Config

- There is always one ServletConfig object per servlet.
- It is used to pass deploy time information to servlet.
- It is used to access ServletContext.
- http://localhost:8080/config/Tester.do
- http://localhost:8080/config1/config.jsp

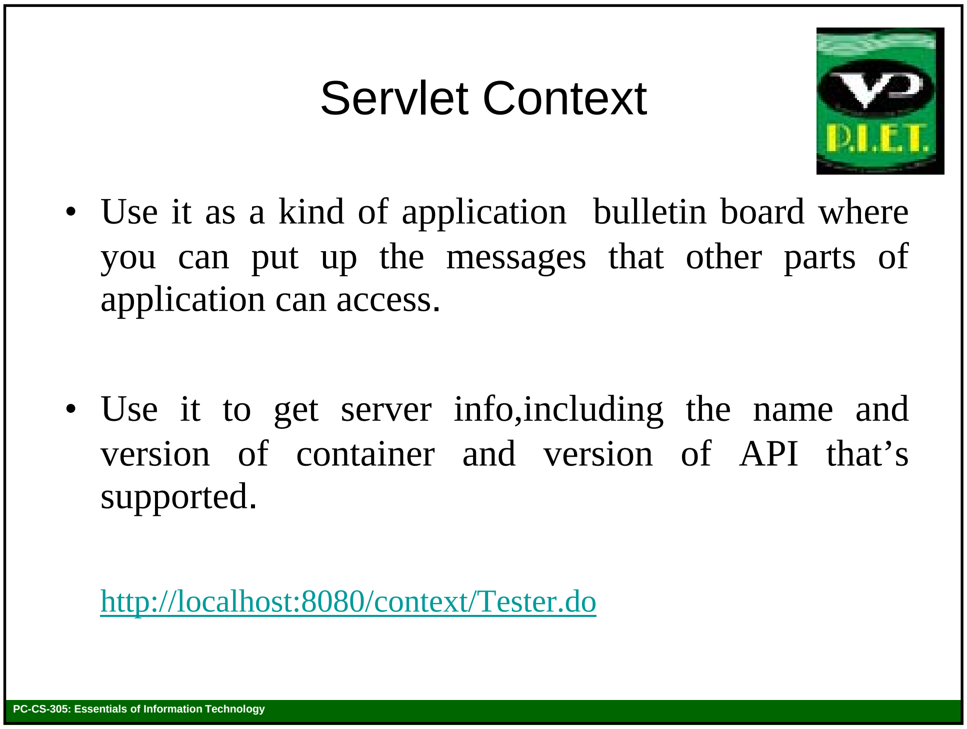
PC-CS-305: Essentials of Information Technology



Servlet Context

- This is an object used by servlet to communicate with its servlet container.
- The ServletContext object is contained in ServletConfig object, which the web server provides the servlet when the servlet is initialized.
- One ServletContext per web app.
- Use it to access web app parameters.

PC-CS-305: Essentials of Information Technology

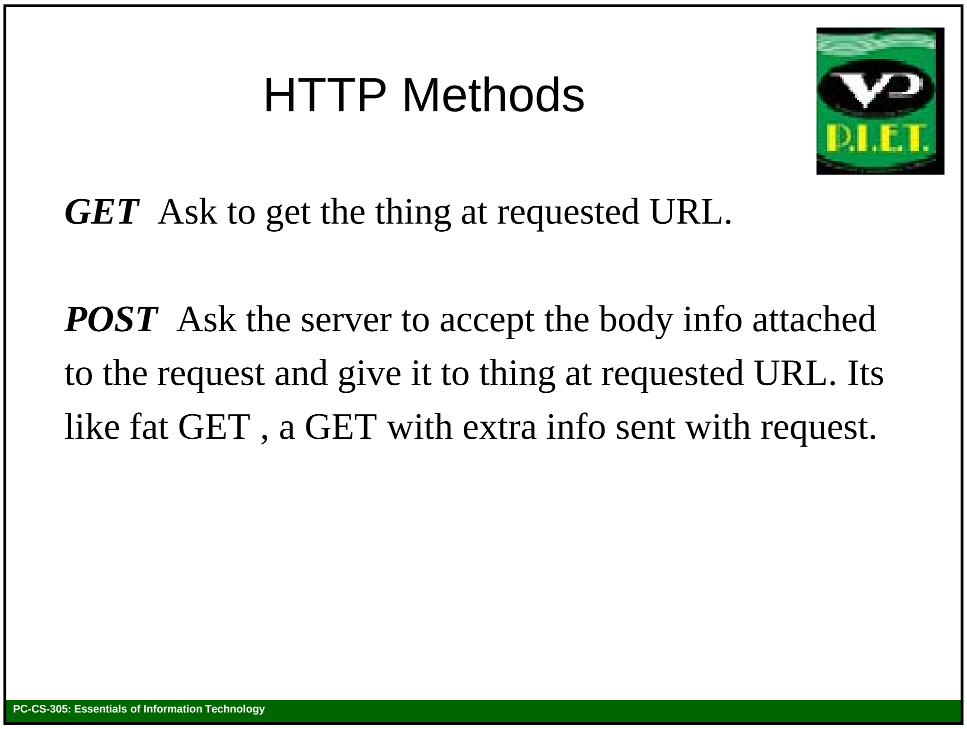


Servlet Context

- Use it as a kind of application bulletin board where you can put up the messages that other parts of application can access.
- Use it to get server info,including the name and version of container and version of API that’s supported.

http://localhost:8080/context/Tester.do

PC-CS-305: Essentials of Information Technology

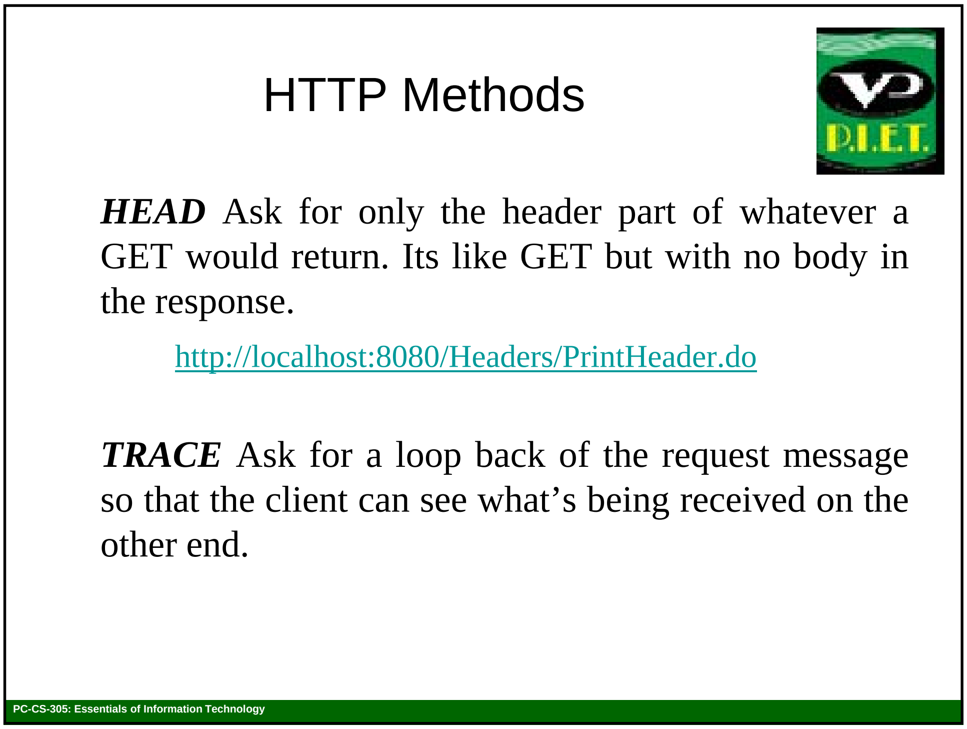


HTTP Methods

GET Ask to get the thing at requested URL.

POST Ask the server to accept the body info attached to the request and give it to thing at requested URL. Its like fat GET , a GET with extra info sent with request.

PC-CS-305: Essentials of Information Technology

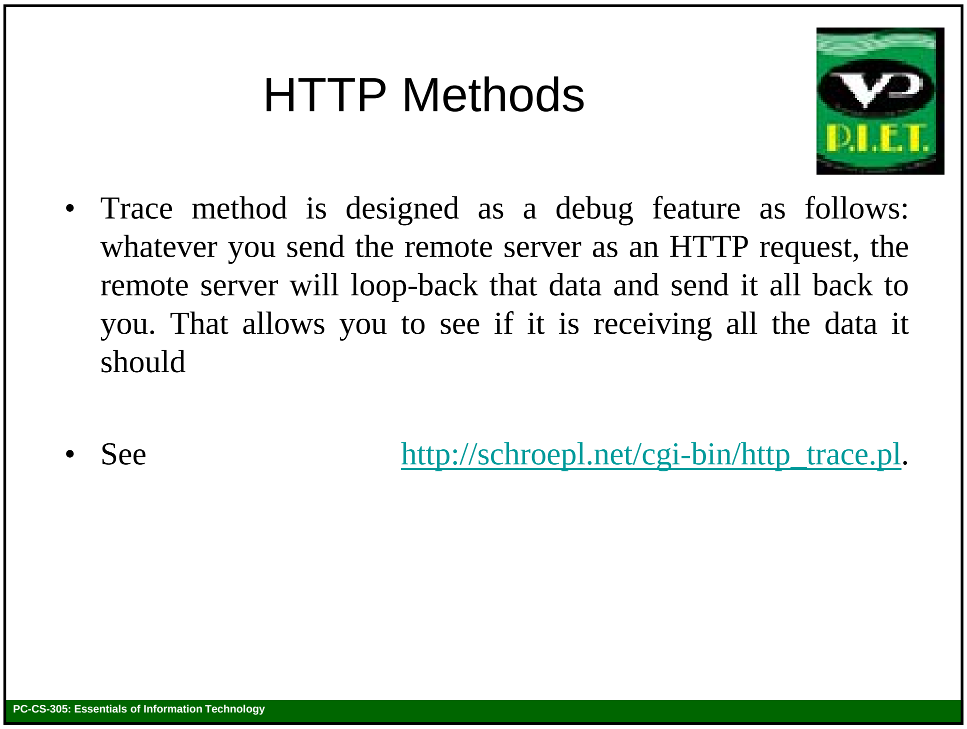


HTTP Methods

HEAD Ask for only the header part of whatever a GET would return. Its like GET but with no body in the response.

http://localhost:8080/Headers/PrintHeader.do

TRACE Ask for a loop back of the request message so that the client can see what's being received on the other end.



HTTP Methods

- Trace method is designed as a debug feature as follows: whatever you send the remote server as an HTTP request, the remote server will loop-back that data and send it all back to you. That allows you to see if it is receiving all the data it should
- See http://schroepl.net/cgi-bin/http_trace.pl.

HTTP Methods

PUT say to put the enclosed info at requested URL.

DELETE say to delete the thing at requested URL.

OPTIONS Ask for a list of HTTP methods to which the thing at the requested URL can respond.

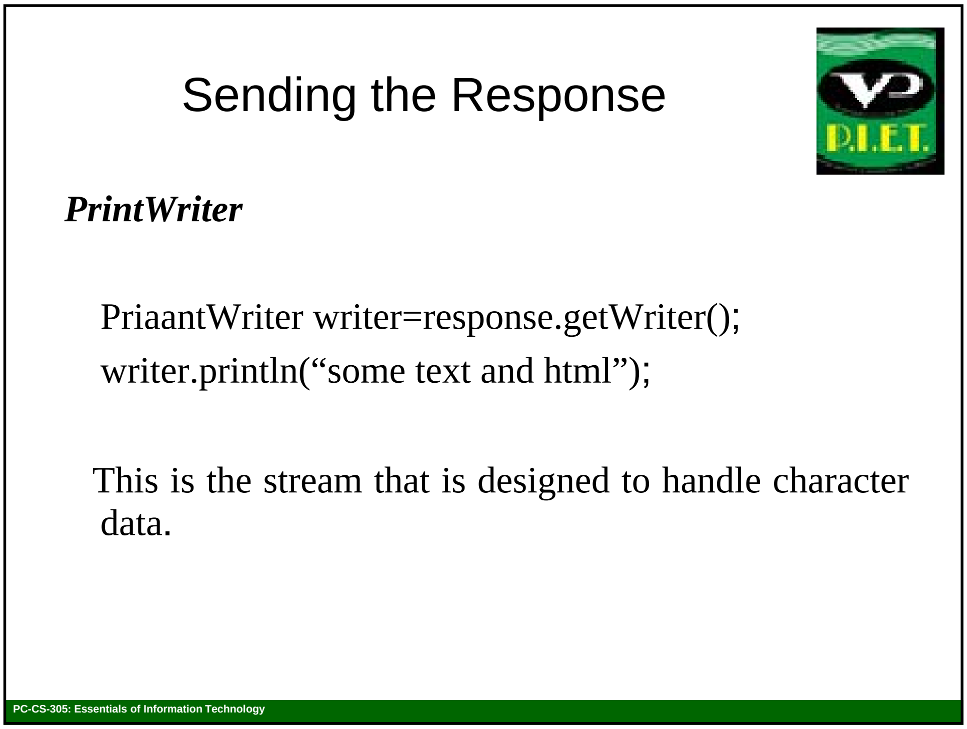
CONNECT say to connect for the purpose of tunneling.

Sending the Response

Output in two formats(character or bytes)

1. ServletOutputStream for bytes

2. PrintWriter for character data.

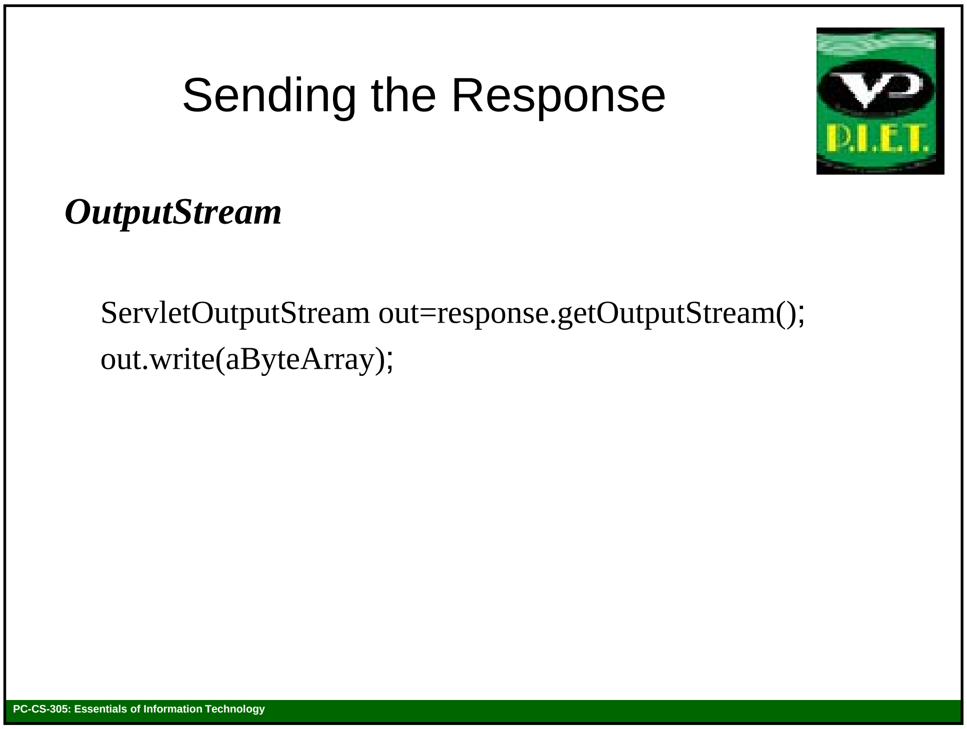


Sending the Response

PrintWriter

```
PriaantWriter writer=response.getWriter();
writer.println(“some text and html”);
```

This is the stream that is designed to handle character data.



Sending the Response

OutputStream

```
ServletOutputStream out=response.getOutputStream();
out.write(aByteArray);
```

Redirect or Dispatch the Request

To have something else handle the response for request

Redirect the request to completely different URL

Dispatch the request to other component in webapp

Redirect makes the client do the work while request dispatcher makes something else on server do the work.

Redirect=client

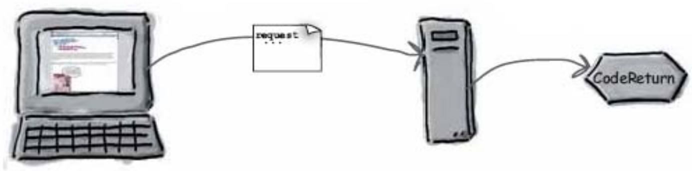
Request dispatch =server

Redirecting a Request

1.Client types a URL in browser bar.

2.The request goes to the server/Container.

3.The servlet decides that request should go to a completely different URL.



Redirecting a Request

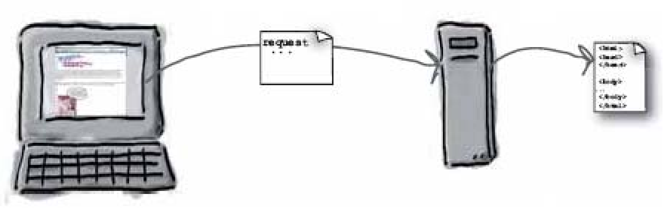
4.The browser gets the response, sees the "301" status code and looks for a location header.

5.The HTTP response has a status code "301" and a location header with a URL as the value.

CodeReturn
response

Redirecting a Request

6.The servlet calls sendRedirect(aString) on the response.



Redirecting a Request

7.The browser makes a new request using the URL that was the value of "Location" header in the previous response. The user might notice that the URL in the browser bar changed

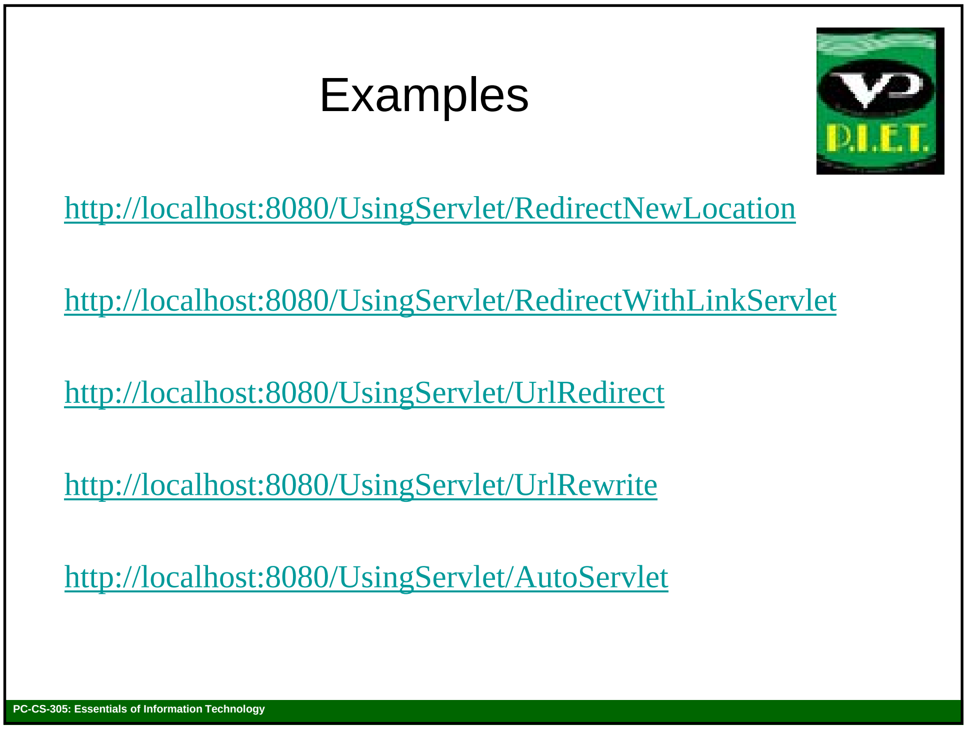
8.There is nothing unique about the request.

9.The server gets the thing at the requested URL.

Redirecting a Request

10.The browser renders the new page.

11.The HTTP response is just like any other response but is not coming from location client typed in.



Examples

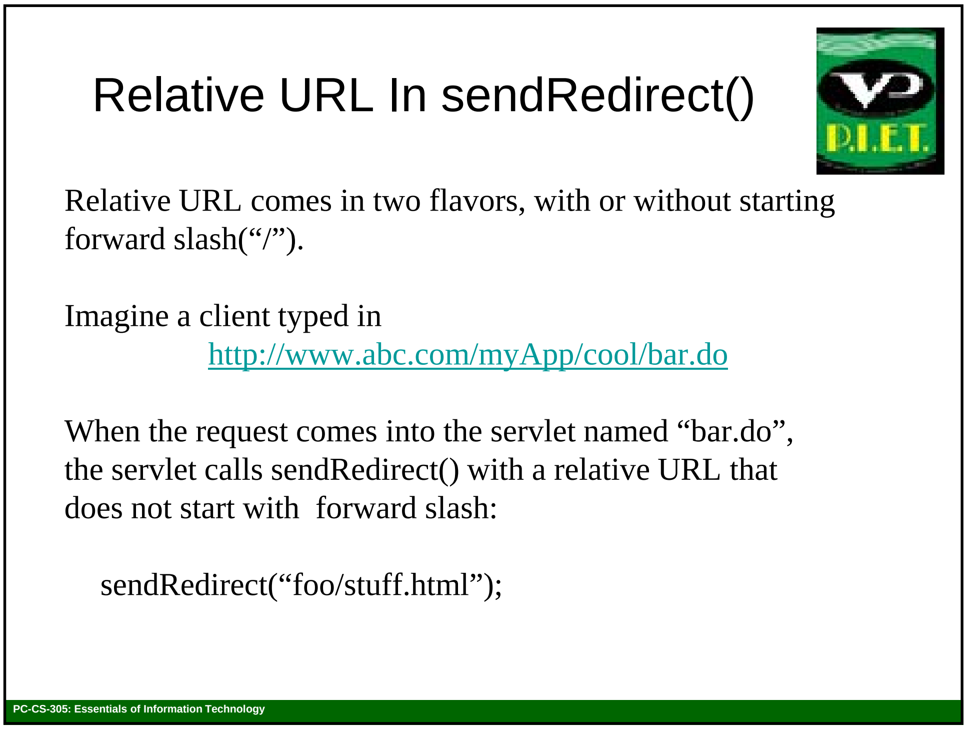
http://localhost:8080/UsingServlet/RedirectNewLocation

http://localhost:8080/UsingServlet/RedirectWithLinkServlet

http://localhost:8080/UsingServlet/UrlRedirect

http://localhost:8080/UsingServlet/UrlRewrite

http://localhost:8080/UsingServlet/AutoServlet



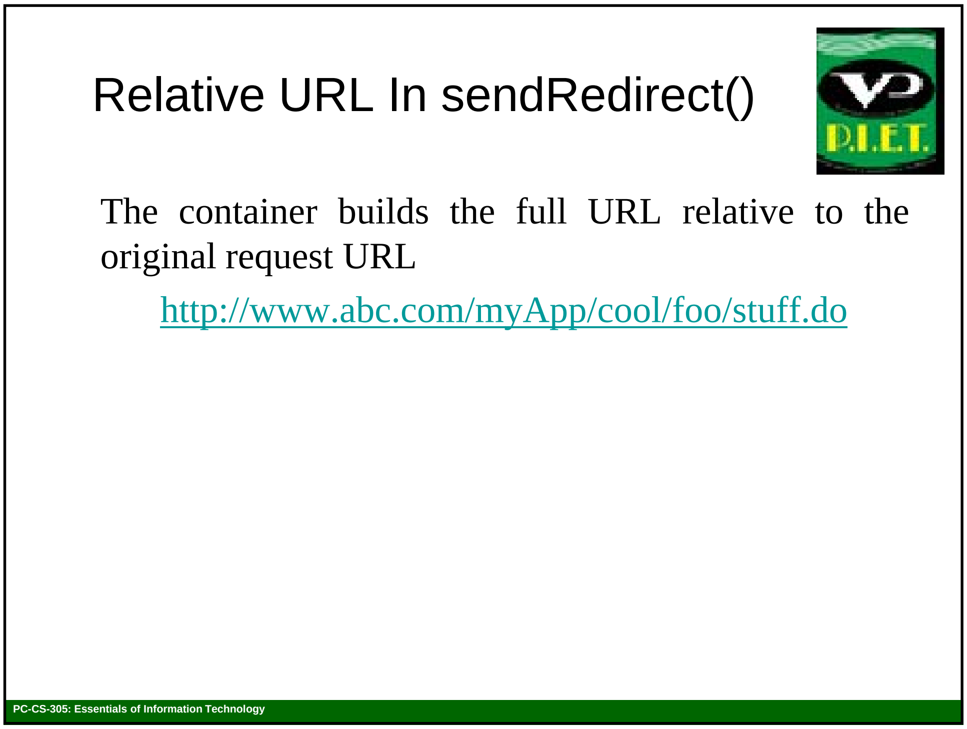
Relative URL In sendRedirect()

Relative URL comes in two flavors, with or without starting forward slash(“/”).

Imagine a client typed in
http://www.abc.com/myApp/cool/bar.do

When the request comes into the servlet named “bar.do”, the servlet calls sendRedirect() with a relative URL that does not start with forward slash:

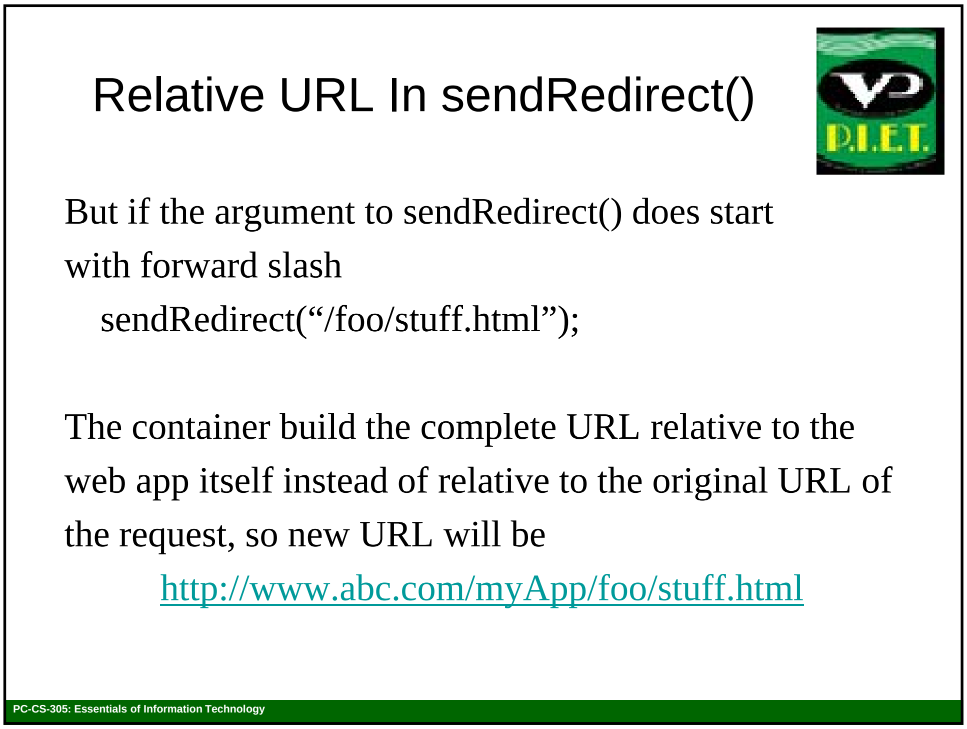
```
sendRedirect(“foo/stuff.html”);
```



Relative URL In sendRedirect()

The container builds the full URL relative to the original request URL

http://www.abc.com/myApp/cool/foo/stuff.do



Relative URL In sendRedirect()

But if the argument to sendRedirect() does start with forward slash

sendRedirect("/foo/stuff.html");

The container build the complete URL relative to the web app itself instead of relative to the original URL of the request, so new URL will be

http://www.abc.com/myApp/foo/stuff.html

Note

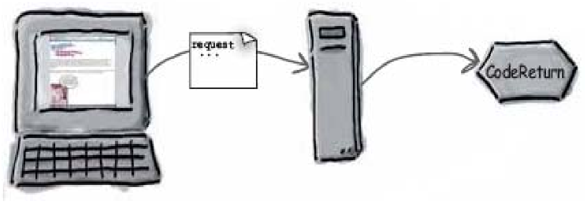
IllegalStateException is thrown if you try to invoke sendRedirect() after writing to the response.

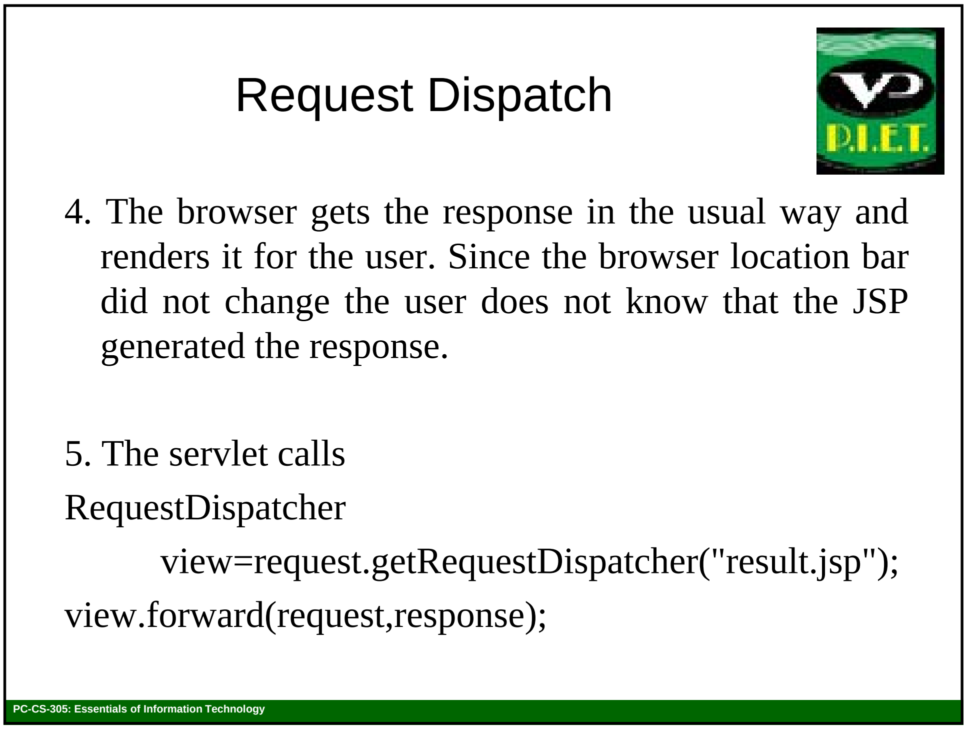
Request Dispatch

1.The user type a URL into the browser bar.

2.The request goes to the server/Container.

3.The servlet decides that the request should go to another part of the web app.





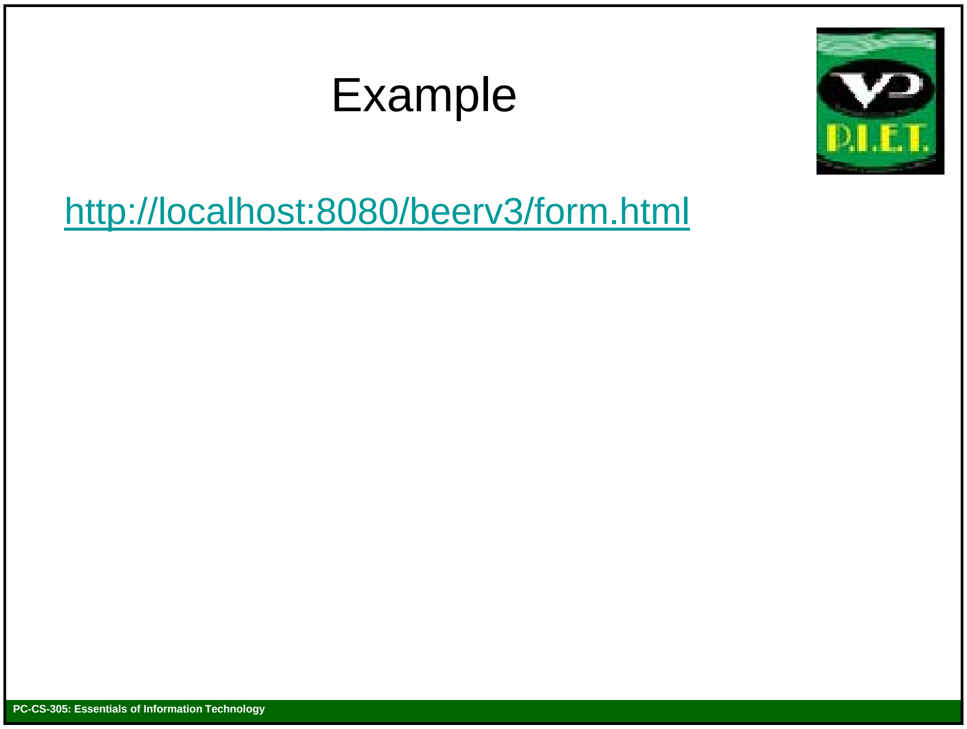
Request Dispatch

4. The browser gets the response in the usual way and renders it for the user. Since the browser location bar did not change the user does not know that the JSP generated the response.

5. The servlet calls

```
RequestDispatcher
        view=request.getRequestDispatcher("result.jsp");
view.forward(request,response);
```

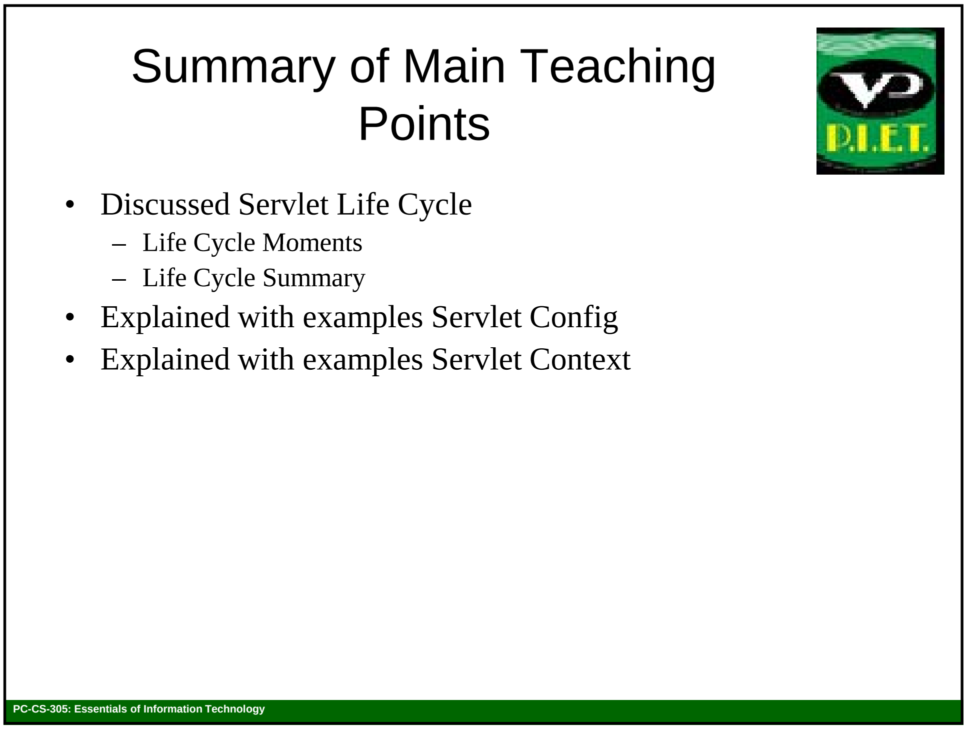
PC-CS-305: Essentials of Information Technology



Example

http://localhost:8080/beerv3/form.html

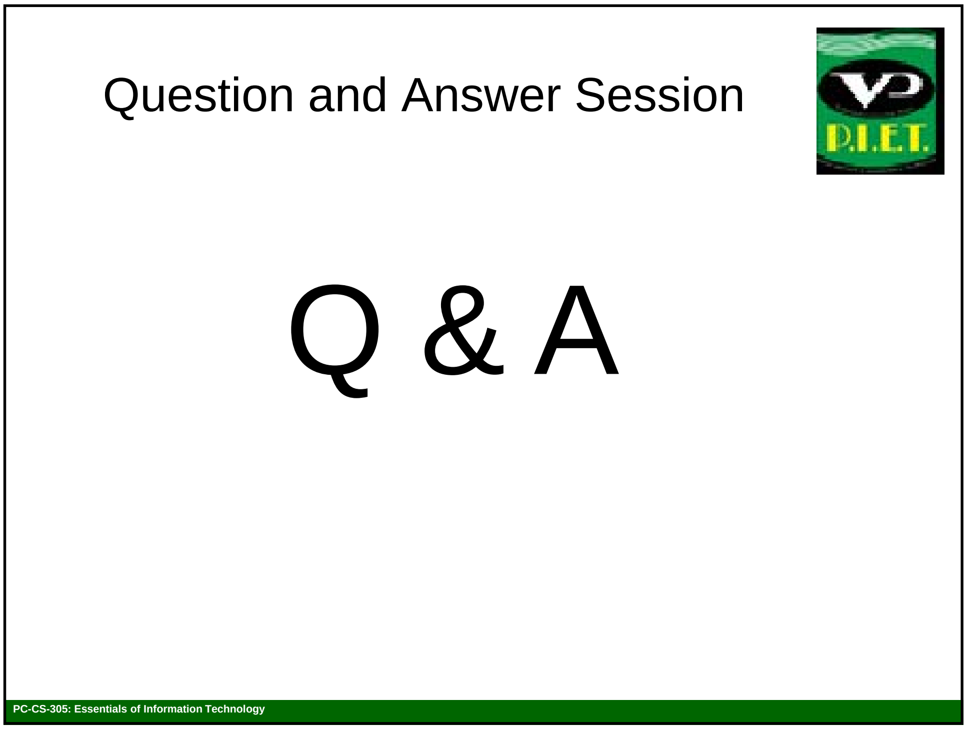
PC-CS-305: Essentials of Information Technology



Summary of Main Teaching Points

- Discussed Servlet Life Cycle
  - Life Cycle Moments
  - Life Cycle Summary
- Explained with examples Servlet Config
- Explained with examples Servlet Context

PC-CS-305: Essentials of Information Technology



Question and Answer Session

Q & A

PC-CS-305: Essentials of Information Technology