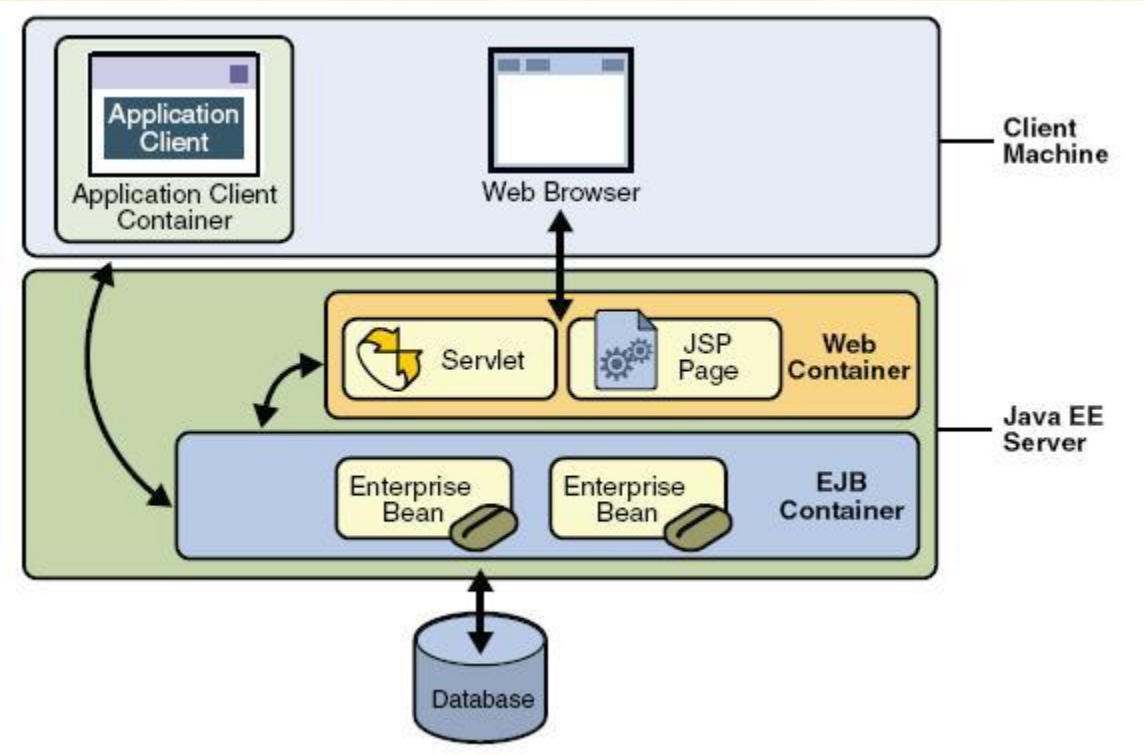
INTRODUCTION TO SERVLETS AND WEB CONTAINERS

Actions in Accord with All the Laws of Nature

© 2016 Maharishi University of Management, All Rights Reserved.

Web container and servlet architecture



A servlet is a Java class that extends the capabilities of servers that host applications access by means of a request-response programming model.

Overview of Web Dynamics

Interaction between a browser (client) and a web server:

- Client requests a resource (file, picture, etc)
- Server returns the resource, or declares it's unavailable

Steps:

1. User clicks a link or button in browser
2. Browser formats request and sends to server
3. Server interprets request and attempts to locate resource
4. Server formats a response and sends to browser
5. Browser renders response into a display for user

Web Dynamics (continued)

Clients and servers communicate using the HTTP protocol

Browsers know how to transform HTML markup into an HTTP request. They also know how to extract HTML from an HTTP response and render it as a displayable HTML page

Servers know how to translate an HTTP request into an action of locating a resource. They also know how to produce an HTTP response that contains HTML and gives information about the requested resource.

HTTP

A request/response protocol that uses TCP/IP to send and receive messages.

IP routes packets of a message from one host to another; TCP is responsible for arranging these packets into the original message at the destination

HTTP protocol is *stateless*; it does not remember any data that was sent or received in previous conversations

HTTP Requests

POST Request – A request for a resource that includes data sent from an HTML form. There is no limit to the amount of data that can be sent in the request.

GET Request – A request to get a resource specified in the URL. Although some form data can be sent also, it is very limited and very insecure

Other types of requests that are rarely used in web apps: PUT, DELETE, OPTIONS, HEAD, TRACE, CONNECT

Anatomy of an HTTP Request

An HTTP Request consists of the following:

- A request line, for example
  GET /images/logo.png HTTP/1.1,
  which requests a resource called /images/logo.png from the server. GET may include a query string e.g. ?name=Joe&job=programmer
- Request headers, such as Accept-Language: en
- An empty line.
- An optional message body (not used in GET requests but contains form data in a POST)

HTTP Response

A response message consists of the following:

- A Status-Line (for example
  HTTP/1.1 200 OK
  which indicates that the client's request succeeded)
- Response headers, such as
  Content-Type: text/html

  May include a “set-cookie” command to maintain a session (more later)
- An empty line
- An optional message body (which often contains the HTML code to be displayed)

Example

Conversation between an HTTP client and an HTTP server running on www.example.com, port 80.

Request:

GET /index.html HTTP/1.1
Host: www.example.com

Response:

HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT ETag: "3f80f-1b6-3e1cb03b"
Content-Type: text/html;
charset=UTF-8
Content-Length: 131
Accept-Ranges: bytes
Connection: close

<html>
<body> Hello World, this is a very simple HTML document. </body>
</html>

What Do Web Servers Serve?

Without making use of some kind of helper application, a web server serves *static content* – files, images, pdfs, videos, exactly as they are on the server machine. Responses cannot be customized based on input data passed in by the client or modified at the time they are delivered.

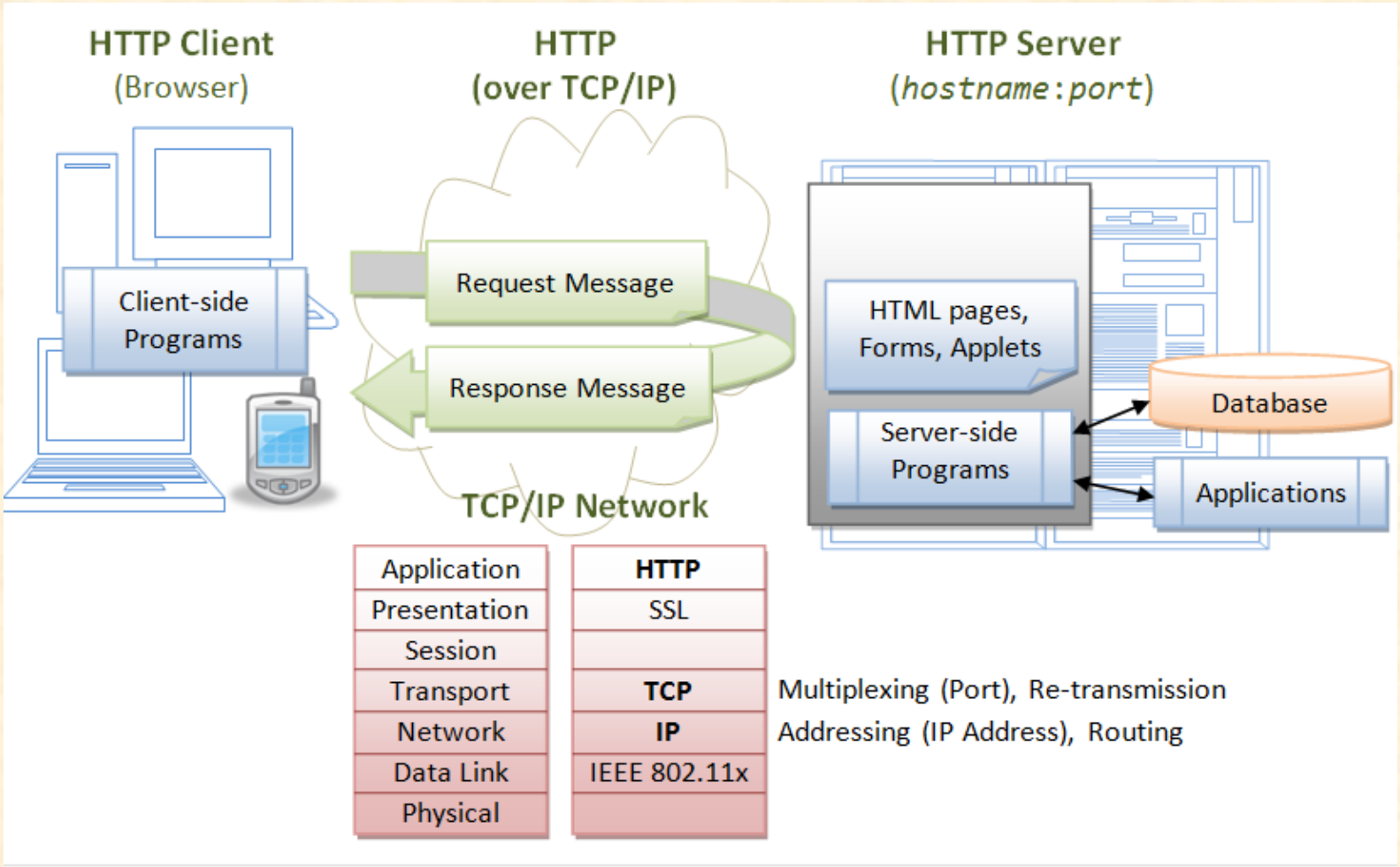
As a consequence:

1. A web server on its own can't handle behavior like log-in, online shopping, data lookups, and computations related to input data -- over the web -- require more than a web server.
2. A web server on its own can't save data to the server.

Servlets: Add Dynamic Content

- A servlet is server-side java code that can handle http requests and return dynamic content
- Servlets are managed by a *servlet engine* or *container*. Each request that comes in results in the spawning of a new thread that runs a servlet (eliminating the cost of creating a new process every time).

Servlet Architecture



Simple Servlet

```
public class SimplestServlet extends HttpServlet {

  protected void doGet(HttpServletRequest request, HttpServletResponse response)
                    throws ServletException, IOException
  {
    PrintWriter out = response.getWriter();
    out.print("<html><head><title>Test</title></head><body>");
    out.print("<form method='post'>");
    out.print("<p>Please click the button</p>");
    out.print("<input type='submit' value='Click me'/>");
    out.print("</form>");
    out.print("</body></html>");
  }

  protected void doPost(HttpServletRequest request, HttpServletResponse response)
                              throws ServletException, IOException
  {
    PrintWriter out = response.getWriter();
    out.print("<html><head><title>Test</title></head><body>");
    out.print("<p>Postback received</p>");
    out.print("</body></html>");
  }
}
```

web.xml

```
<web-app …>
    <servlet>
        <servlet-name>SimplestServlet</servlet-name>
        <servlet-class>mum.cs545.SimplestServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>SimplestServlet</servlet-name>
        <url-pattern>/SimplestServlet</url-pattern>
    </servlet-mapping>
    <session-config>
        …
    </session-config>
</web-app>
```

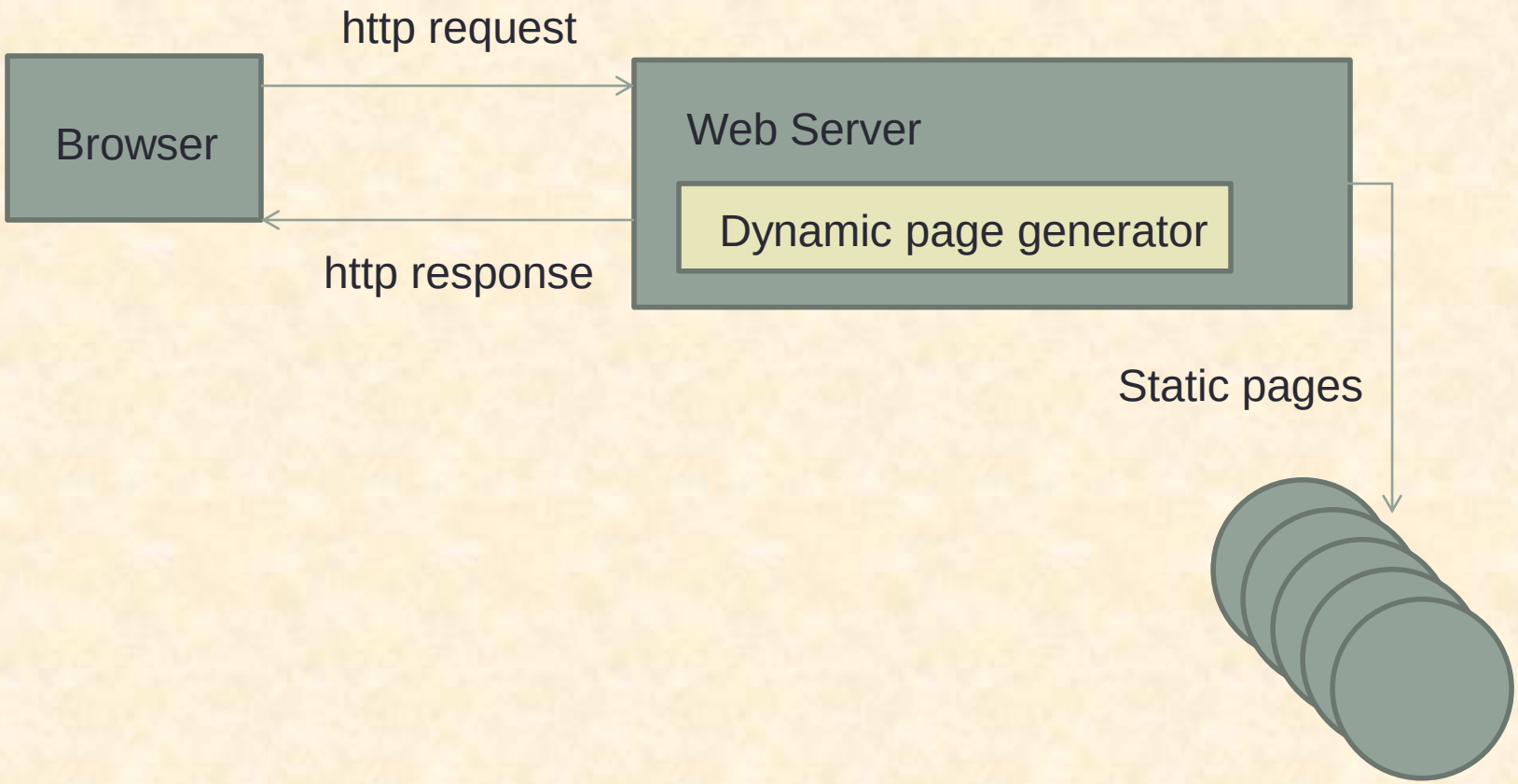
Main Point 1

Servlets are the basis of dynamic web applications. Servlets process information from a request object and generate new information in a response object.

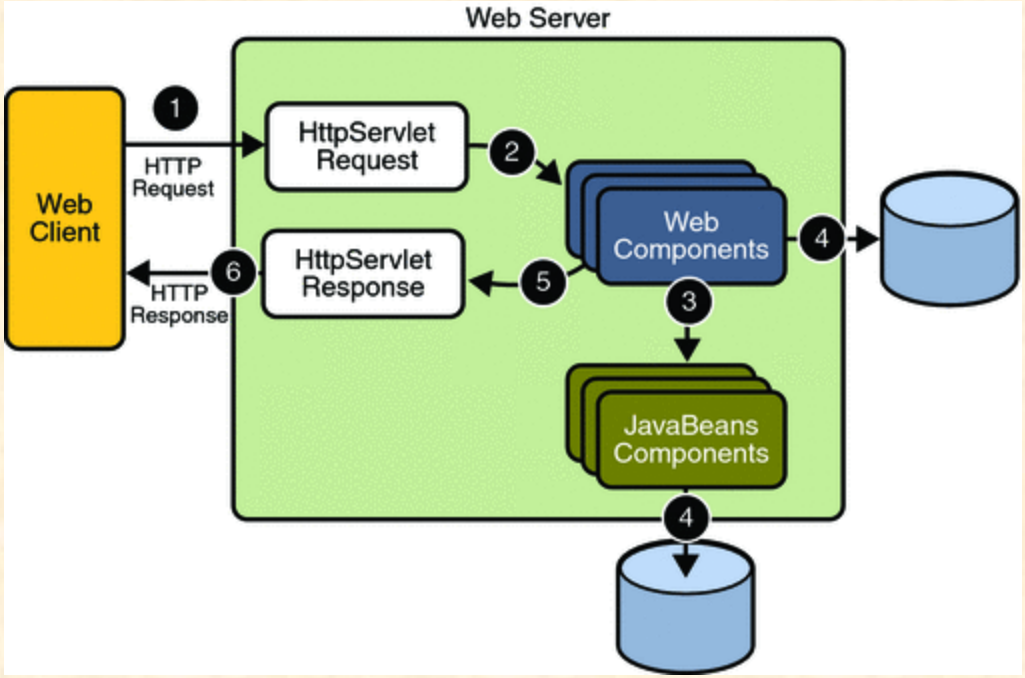
Science of Consciousness:

Analogously, humans also operate by giving responses to requests. Response are more likely to be correct and appropriate to the extent that one has broad awareness to consider all relevant information and fine focus to understand the request.

Dynamic versus static pages



Web server with Servlet container



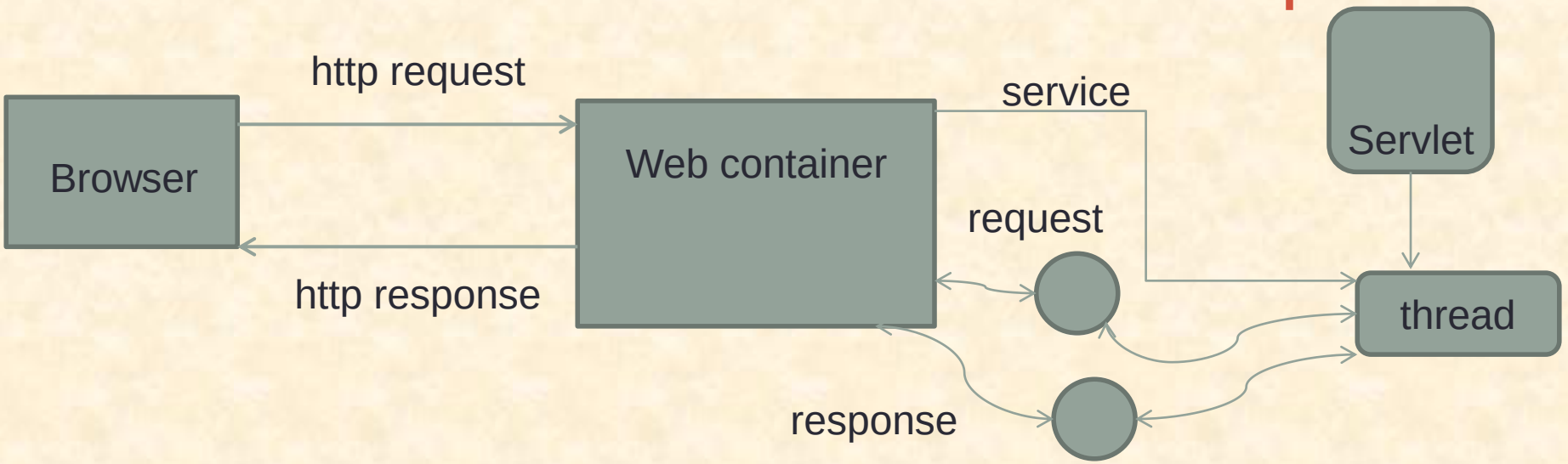
Containers provide fundamental support

- network communications
  - communicating with web server
- lifecycle management
  - no “main” method in a servlet, …
- concurrency
- state management
  - session and context attributes
- security
- Support for JSP (JSF, JPA, JTA, EJB, …)

The Container

- Servers that support servlets have as a helper app a *servlet container.*
- When a request comes to the web server, if the server sees the request is for a servlet, it passes the request data to the servlet container.
- The servlet container locates the servlet, creates request and response objects and passes them to the servlet, and returns to the web server the response stream that the servlet produces.
- The web server sends the response back to the client browser to be rendered.

How container handles an HTTP request

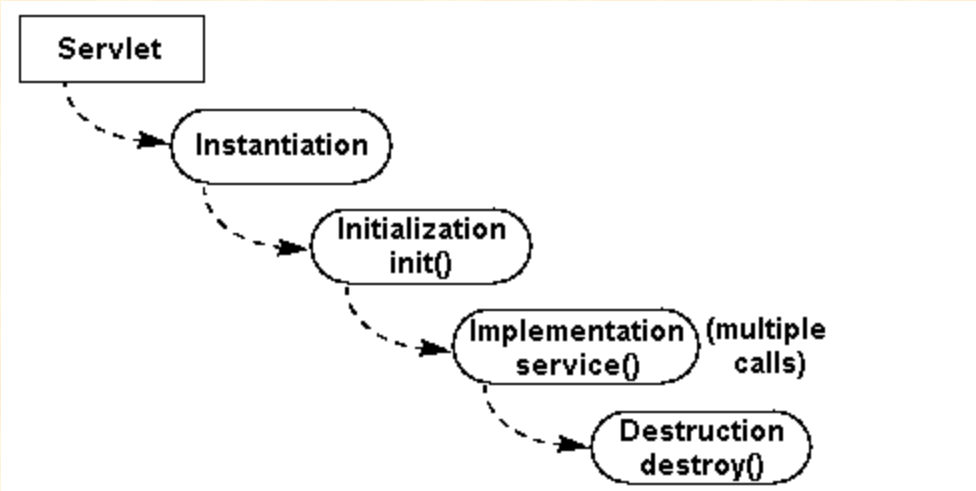


- Container receives new request for a servlet
- Creates HttpServletRequest and HttpServletResponse objects
- Calls service method on HttpServlet object in thread
- When thread completes, converts response object into HTTP response message

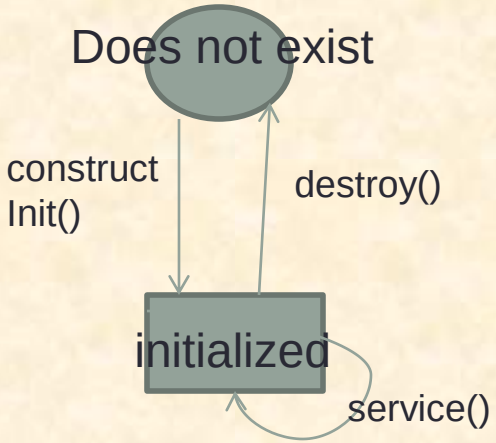
Main Point 2

Web containers provide essential support services for servlets. **Science of Consciousness**: Experience of the unified field of pure consciousness provides essential support services for broad awareness and fine focus in the individual.

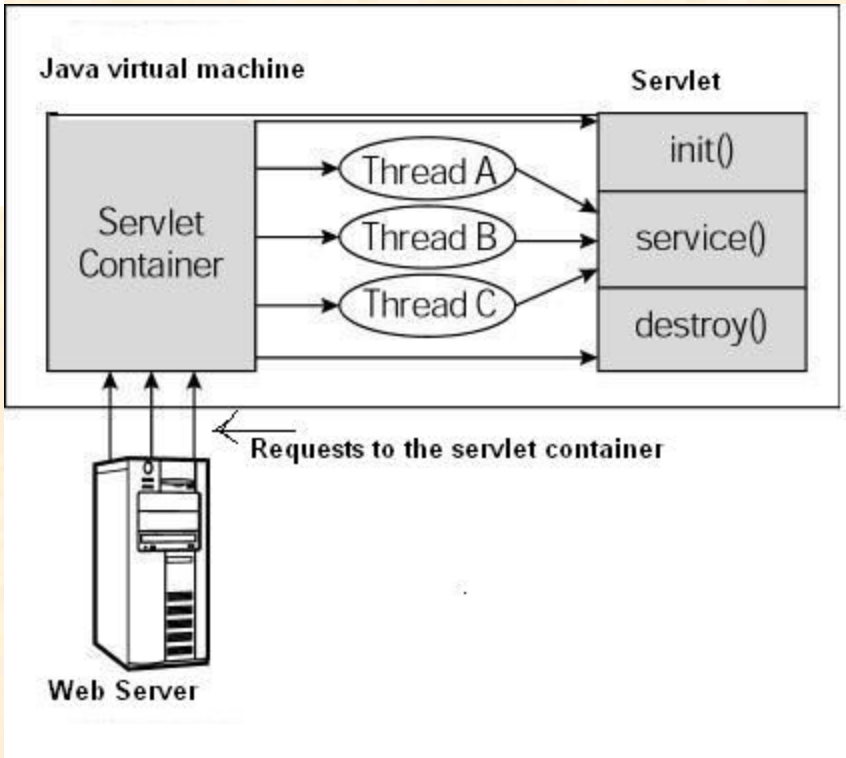
Servlet life cycle



. Load
. Create
. Init
. Service
. Destroy



- 1 instance of servlet
- new thread created for every request
  - Then service() called on the thread
- All threads share instance variables
- Each thread has own stack for local variables
- Compare with mylets



Servlet life cycle

1. Load servlet class (Class c = Class.forname(...))
2. Instantiate servlet (c.newInstance());
3. init() called only once in the servlet's life.
   Must complete before Container can call service().
4. service() (called for each request, each request runs in a separate thread)
5. destroy() (called only once)

Main Point 3

Web containers manage the life cycle of servlets. **Science of Consciousness**: The unified field manages the entire universe, and by experiencing this field of awareness on a regular basis we spontaneously act more and more in accord with all the laws of nature.

Specifying a Servlet

Servlets are given names that are used by different parts of a web app:

1. Each servlet is a Java class and so has a fully qualified class name. Example: mum.Hello
2. Another name for the servlet is used in an HTML form to tell the Web Server that a servlet is being requested and to tell the Container *which* servlet is needed. Example: hello. This is the name used by client in the form (action = “hello”) to refer to servlet.
3. A third name is the *internal name* that the container uses to keep track of the servlets it is managing. This is the value specified as the *servlet-name* in the web.xml configuration file.

Three Names of a Servlet

```
<servlet>
    <servlet-name>hello</servlet-name>
    <servlet-class>mum.Hello</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>hello</servlet-name>
    <url-pattern>/hello</url-pattern>
</servlet-mapping>
```

- servlet-name is the internal name of the servlet
- serlvet-class is the Java name of the class
- url-pattern is the way the servlet is specified in the HTML form
  <form action="*hello*" *method="get"*>