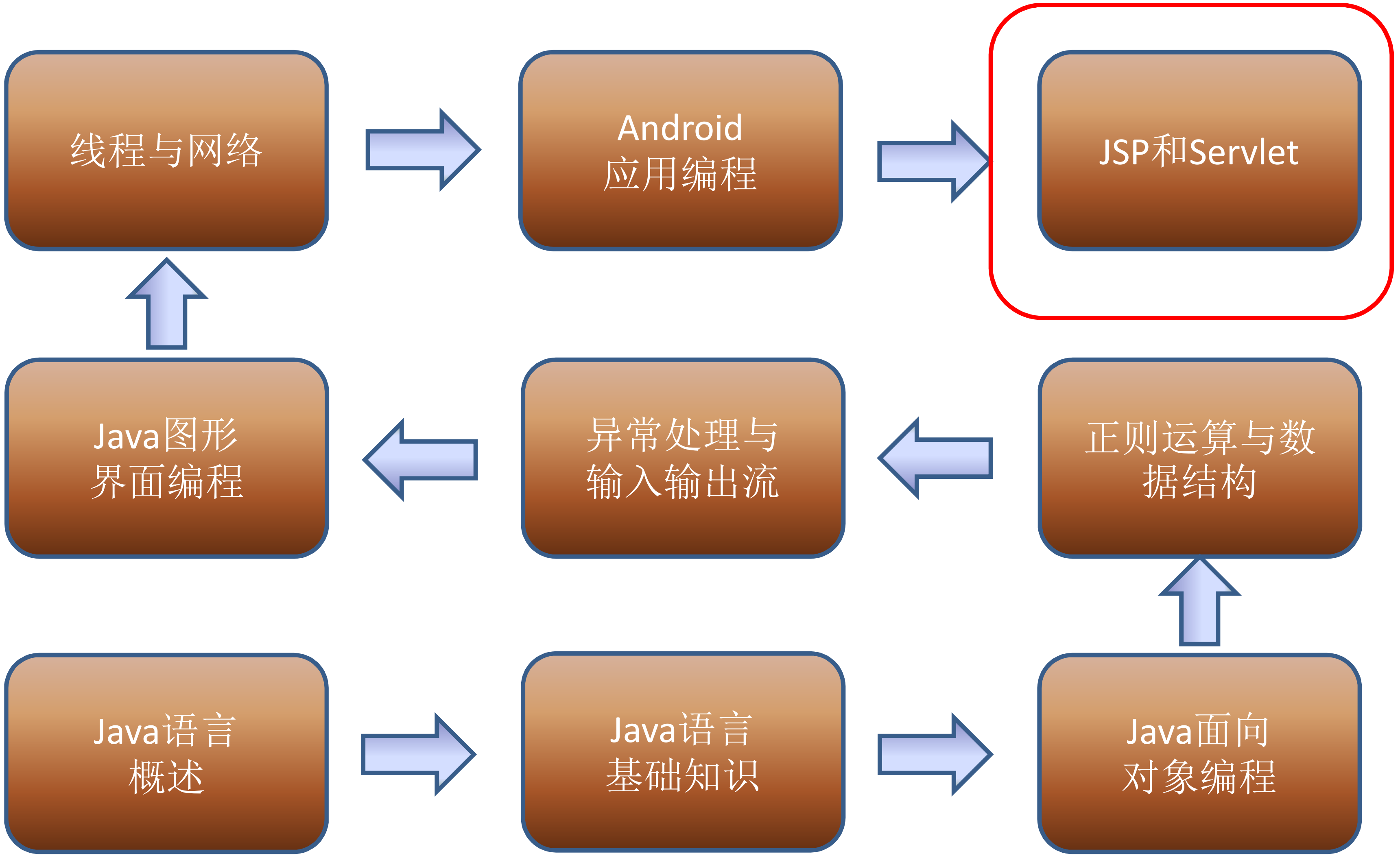
JSP和Servlet
Web Programming with Java

课程内容安排


线程与网络
Android 应用编程
JSP和Servlet
Java图形 界面编程
异常处理与 输入输出流
正则运算与数 据结构
Java语言 概述
Java语言 基础知识
Java面向 对象编程

课前思考

- Who is the inventor of Web?
- How to write a static HTML file?
- How to write a HTML file dynamically?
- What are the operations of HTTP?
- How to generate HTML with Java?
- How to embed Java code into HTML?

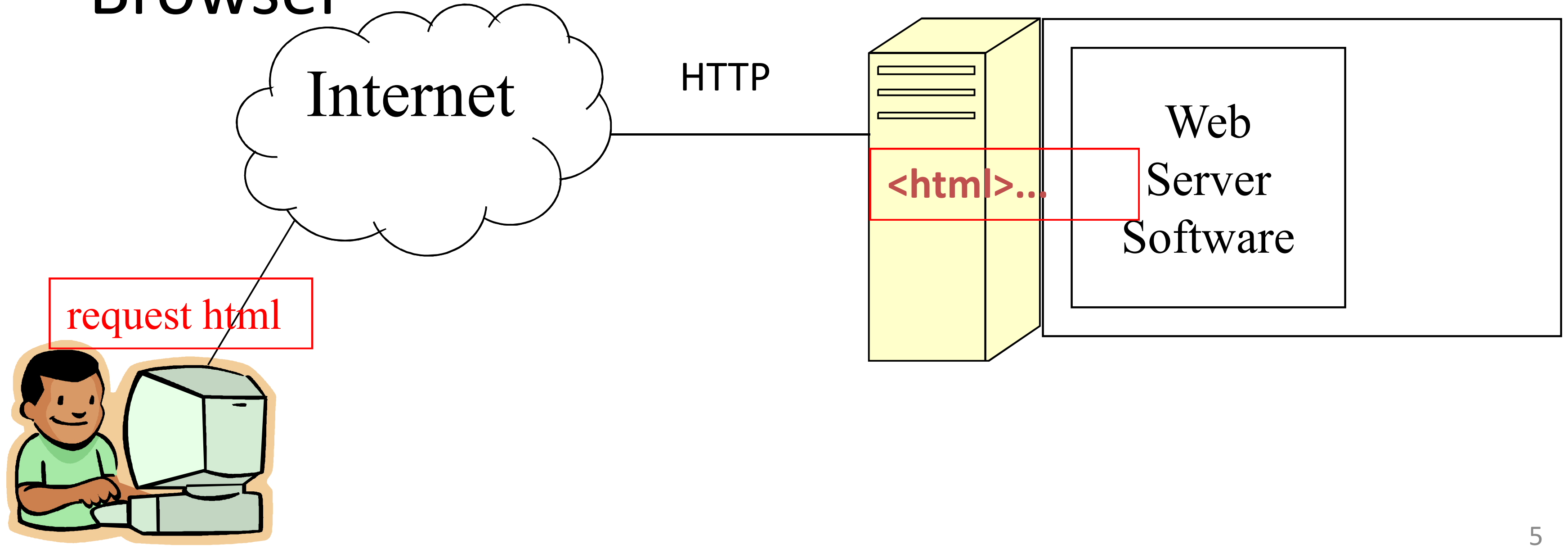
Java and Web

James Gosling
Sun Microsystems
(Google now)
Inventor of Java in 1995

Tim Berners-Lee
MIT & W3C
Inventor of WWW in 1989

Web

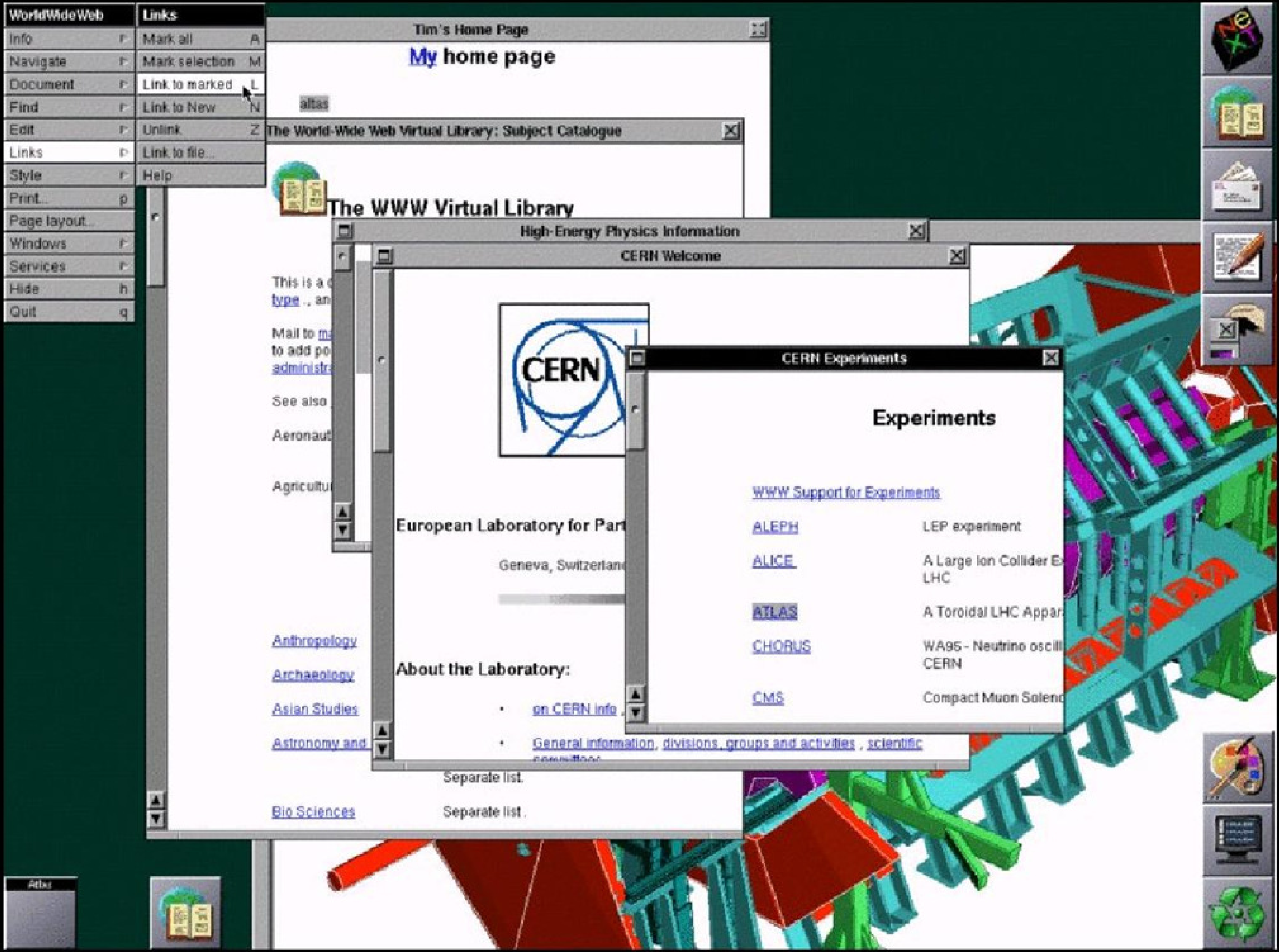
- HTML
- HTTP
- Web Server
- Browser



First web server (Tim Berners-Lee's NeXTcube) – 1990

"This machine is a server. DO NOT POWER IT DOWN!!"

WorldWideWeb - 1990



My home page

altas

The WWW Virtual Library

This is a
type ., an

Mail to m
to add po
administr

See also

Aeronaut

Agricultu

Anthropology

Archaeology

Asian Studies

Astronomy and Separate list.

Bio Sciences Separate list .

European Laboratory for Part

Geneva, Switzerlan

About the Laboratory:

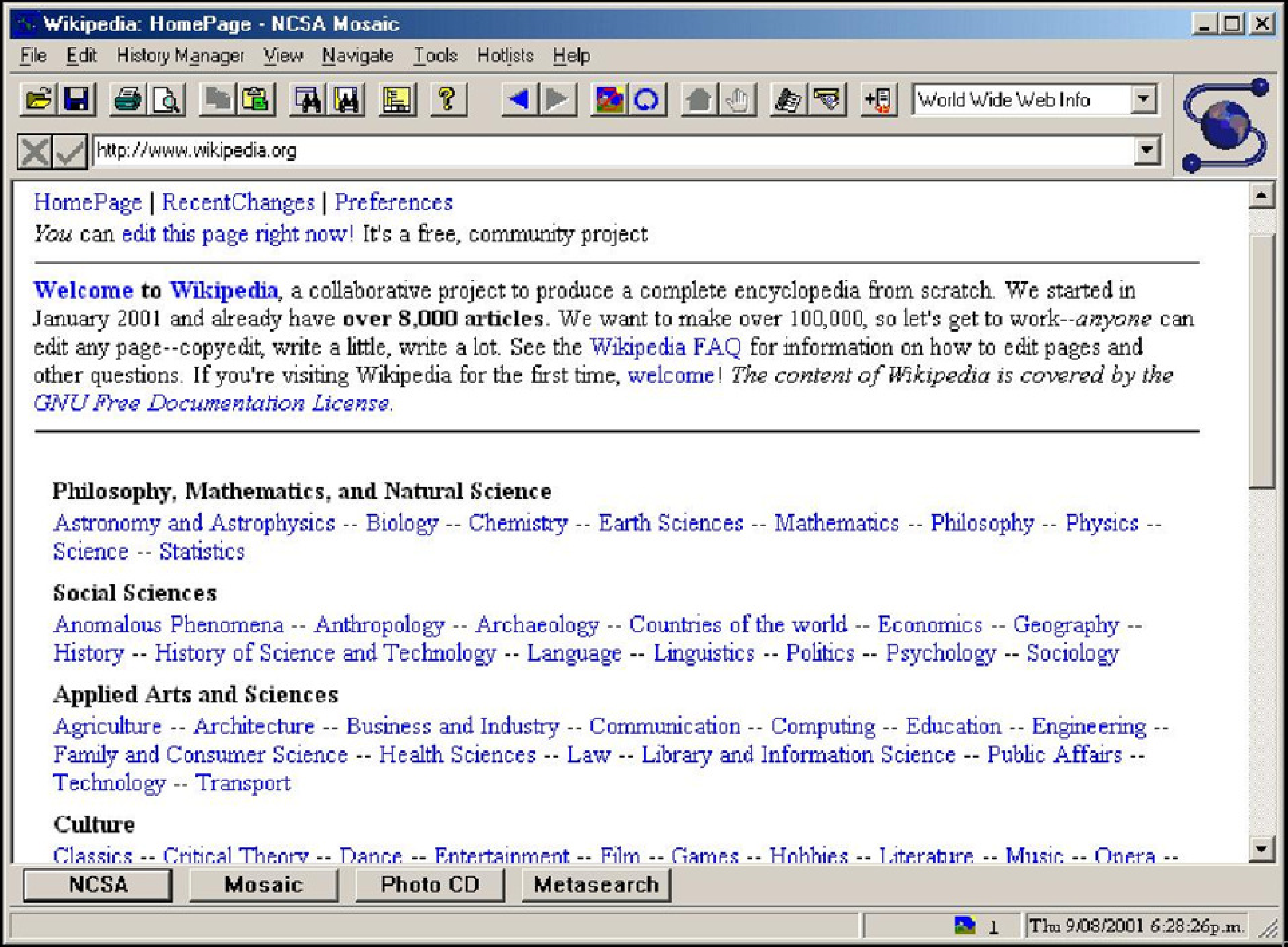
- on CERN info
- General information, divisions, groups and activities , scientific committees

Experiments

WWW Support for Experiments

ALEPH	LEP experiment
ALICE	A Large Ion Collider E LHC
ATLAS	A Toroidal LHC Appar
CHORUS	WA95 - Neutrino oscill CERN
CMS	Compact Muon Solen

Mosaic - 1993


Wikipedia: HomePage - NCSA Mosaic
File Edit History Manager View Navigate Tools Hotlists Help
World Wide Web Info
http://www.wikipedia.org
NCSA
Mosaic
Photo CD
Metasearch
Thu 9/08/2001 6:28:26p.m.

HomePage | RecentChanges | Preferences
You can edit this page right now! It's a free, community project

Welcome to Wikipedia, a collaborative project to produce a complete encyclopedia from scratch. We started in January 2001 and already have **over 8,000 articles**. We want to make over 100,000, so let's get to work--*anyone* can edit any page--copyedit, write a little, write a lot. See the Wikipedia FAQ for information on how to edit pages and other questions. If you're visiting Wikipedia for the first time, welcome! *The content of Wikipedia is covered by the GNU Free Documentation License.*

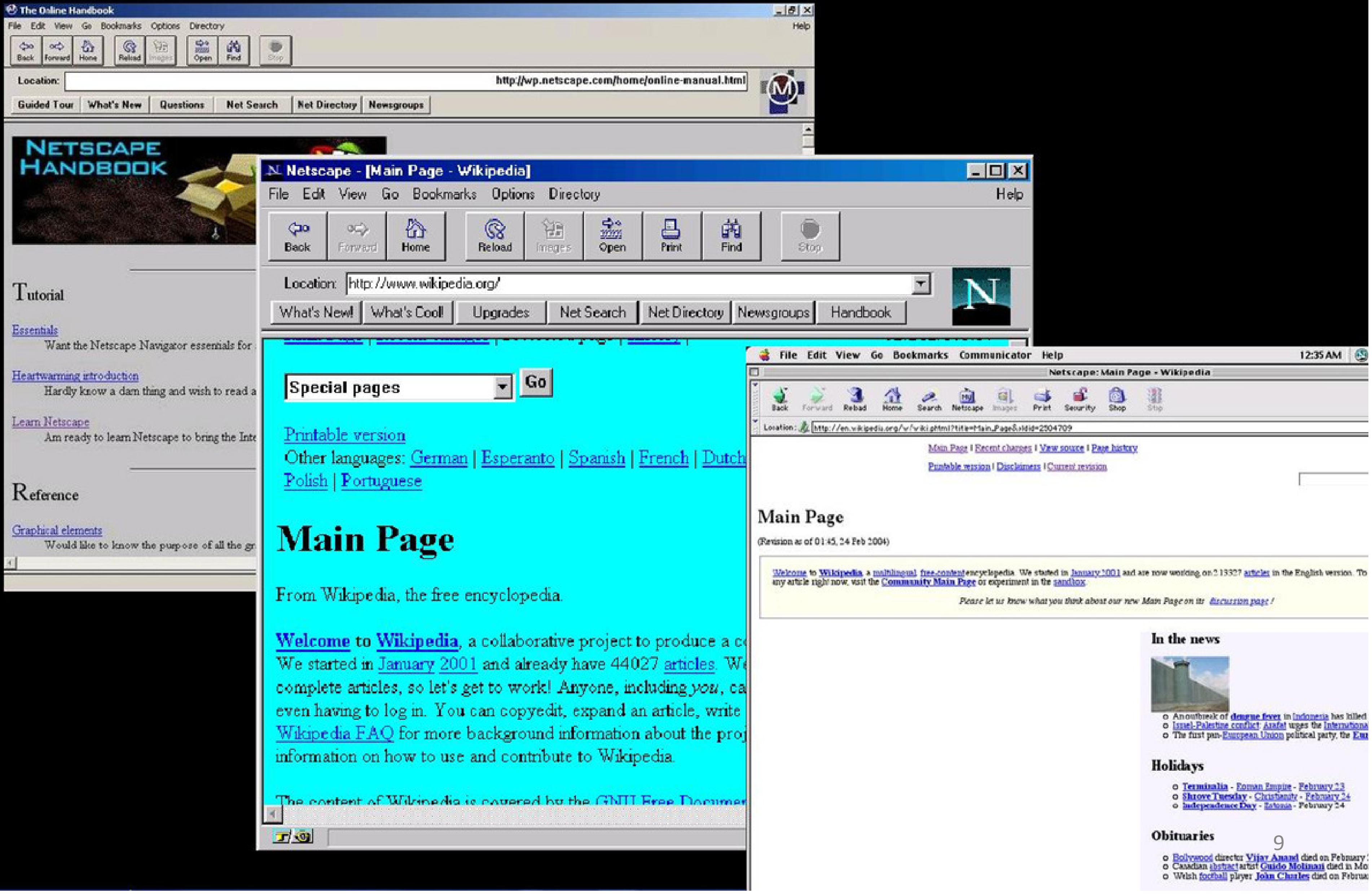
Philosophy, Mathematics, and Natural Science
Astronomy and Astrophysics -- Biology -- Chemistry -- Earth Sciences -- Mathematics -- Philosophy -- Physics -- Science -- Statistics

Social Sciences
Anomalous Phenomena -- Anthropology -- Archaeology -- Countries of the world -- Economics -- Geography -- History -- History of Science and Technology -- Language -- Linguistics -- Politics -- Psychology -- Sociology

Applied Arts and Sciences
Agriculture -- Architecture -- Business and Industry -- Communication -- Computing -- Education -- Engineering -- Family and Consumer Science -- Health Sciences -- Law -- Library and Information Science -- Public Affairs -- Technology -- Transport

Culture
Classics -- Critical Theory -- Dance -- Entertainment -- Film -- Games -- Hobbies -- Literature -- Music -- Opera --

Netscape Navigator – 1994 - 1998


The Online Handbook
File Edit View Go Bookmarks Options Directory Help
Back Forward Home Reload Images Open Find Stop
Location: http://wp.netscape.com/home/online-manual.html
Guided Tour What's New Questions Net Search Net Directory Newsgroups
NETSCAPE HANDBOOK

Tutorial

Essentials
Want the Netscape Navigator essentials for

Heartwarming introduction
Hardly know a darn thing and wish to read a

Learn Netscape
Am ready to learn Netscape to bring the Inte

Reference

Graphical elements
Would like to know the purpose of all the gr

Netscape - [Main Page - Wikipedia]
File Edit View Go Bookmarks Options Directory Help
Back Forward Home Reload Images Open Print Find Stop
Location: http://www.wikipedia.org/
What's New! What's Cool! Upgrades Net Search Net Directory Newsgroups Handbook
Special pages
Go

Printable version
Other languages: German | Esperanto | Spanish | French | Dutch
Polish | Portuguese

Main Page

From Wikipedia, the free encyclopedia.

Welcome to Wikipedia, a collaborative project to produce a c
We started in January 2001 and already have 44027 articles. We
complete articles, so let's get to work! Anyone, including *you*, ca
even having to log in. You can copyedit, expand an article, write
Wikipedia FAQ for more background information about the proj
information on how to use and contribute to Wikipedia.

The content of Wikipedia is covered by the GNU Free Documen

File Edit View Go Bookmarks Communicator Help 12:35 AM
Netscape: Main Page - Wikipedia
Back Forward Reload Home Search Netscape Images Print Security Shop Stop
Location: http://en.wikipedia.org/w/wiki.phtml?title=Main_Page&oldid=2504709

Main Page | Recent changes | View source | Page history
Printable version | Disclaimers | Current revision

Main Page

(Revision as of 01:45, 24 Feb 2004)

Welcome to Wikipedia, a multilingual free-content encyclopedia. We started in January 2001 and are now working on 213327 articles in the English version. To
any article right now, visit the Community Main Page or experiment in the sandbox.

Please let us know what you think about our new Main Page on its discussion page!

In the news

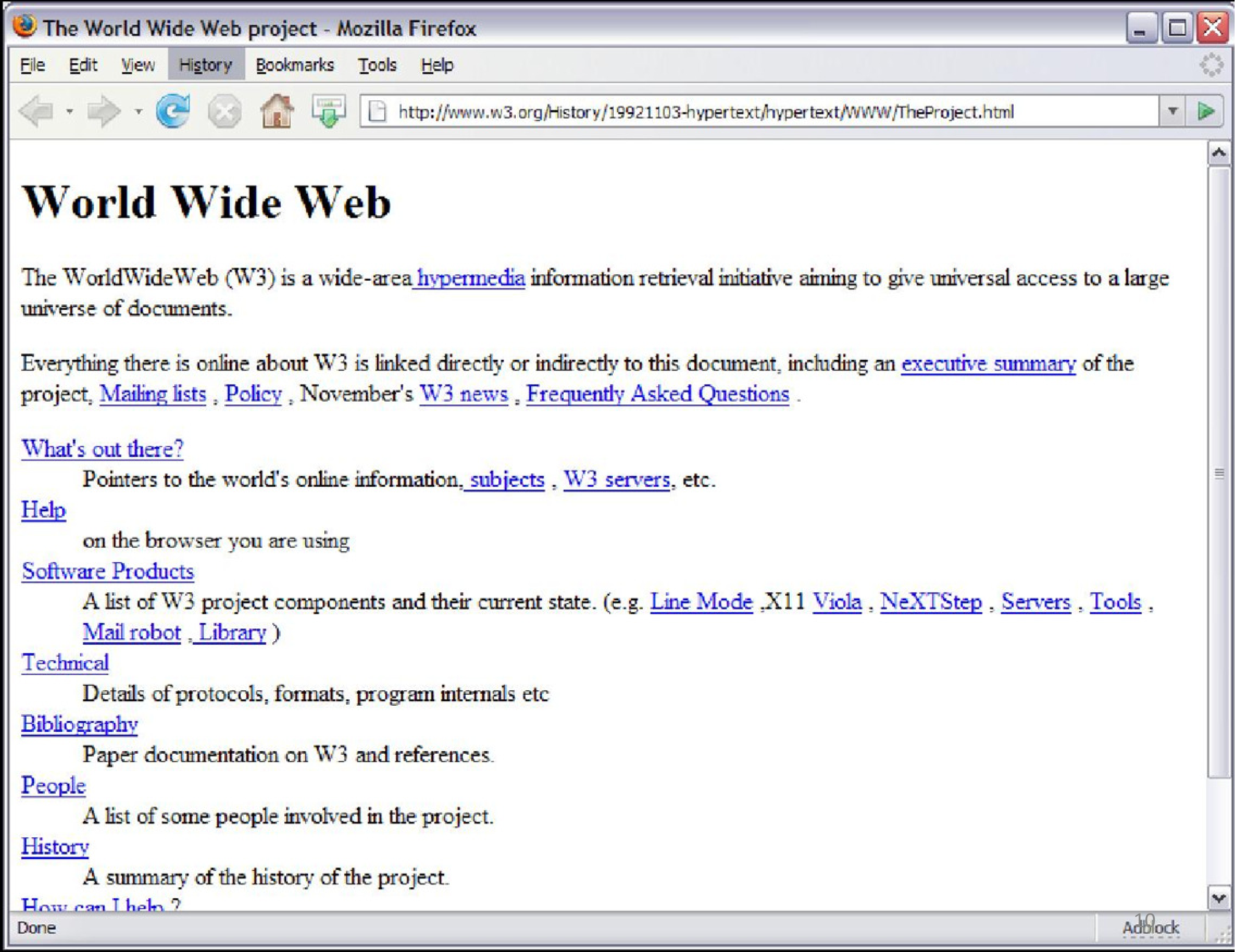
- An outbreak of dengue fever in Indonesia has killed
- Israel-Palestine conflict: Arafat urges the Internationa
- The first pan-European Union political party, the Eu

Holidays

- Terminalia - Roman Empire - February 23
- Shrove Tuesday - Christianity - February 24
- Independence Day - Estonia - February 24

Obituaries

- Bollywood director Vijay Anand died on February
- Canadian abstract artist Guido Molinari died in Mo
- Welsh football player John Charles died on Februa


The World Wide Web project - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.w3.org/History/19921103-hypertext/hypertext/WWW/TheProject.html

World Wide Web

The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary of the project, Mailing lists , Policy , November's W3 news , Frequently Asked Questions .

What's out there?
: Pointers to the world's online information, subjects , W3 servers, etc.

Help
: on the browser you are using

Software Products
: A list of W3 project components and their current state. (e.g. Line Mode ,X11 Viola , NeXTStep , Servers , Tools , Mail robot , Library)

Technical
: Details of protocols, formats, program internals etc

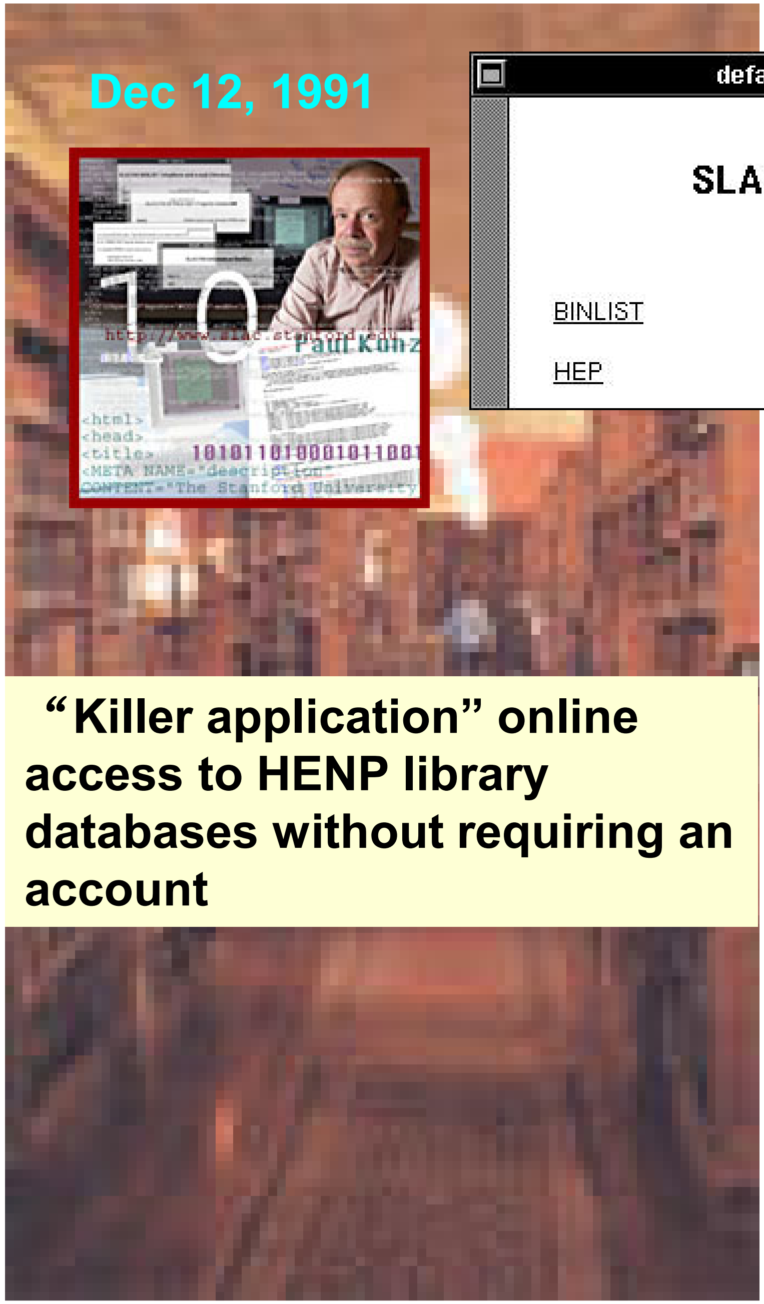
Bibliography
: Paper documentation on W3 and references.

People
: A list of some people involved in the project.

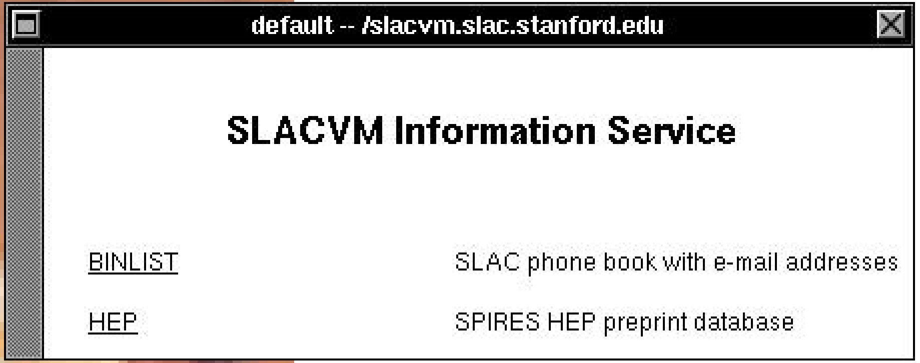
History
: A summary of the history of the project.

How can I help ?

First US Web Site



"Killer application" online access to HENP library databases without requiring an account



default -- /slacvm.slac.stanford.edu

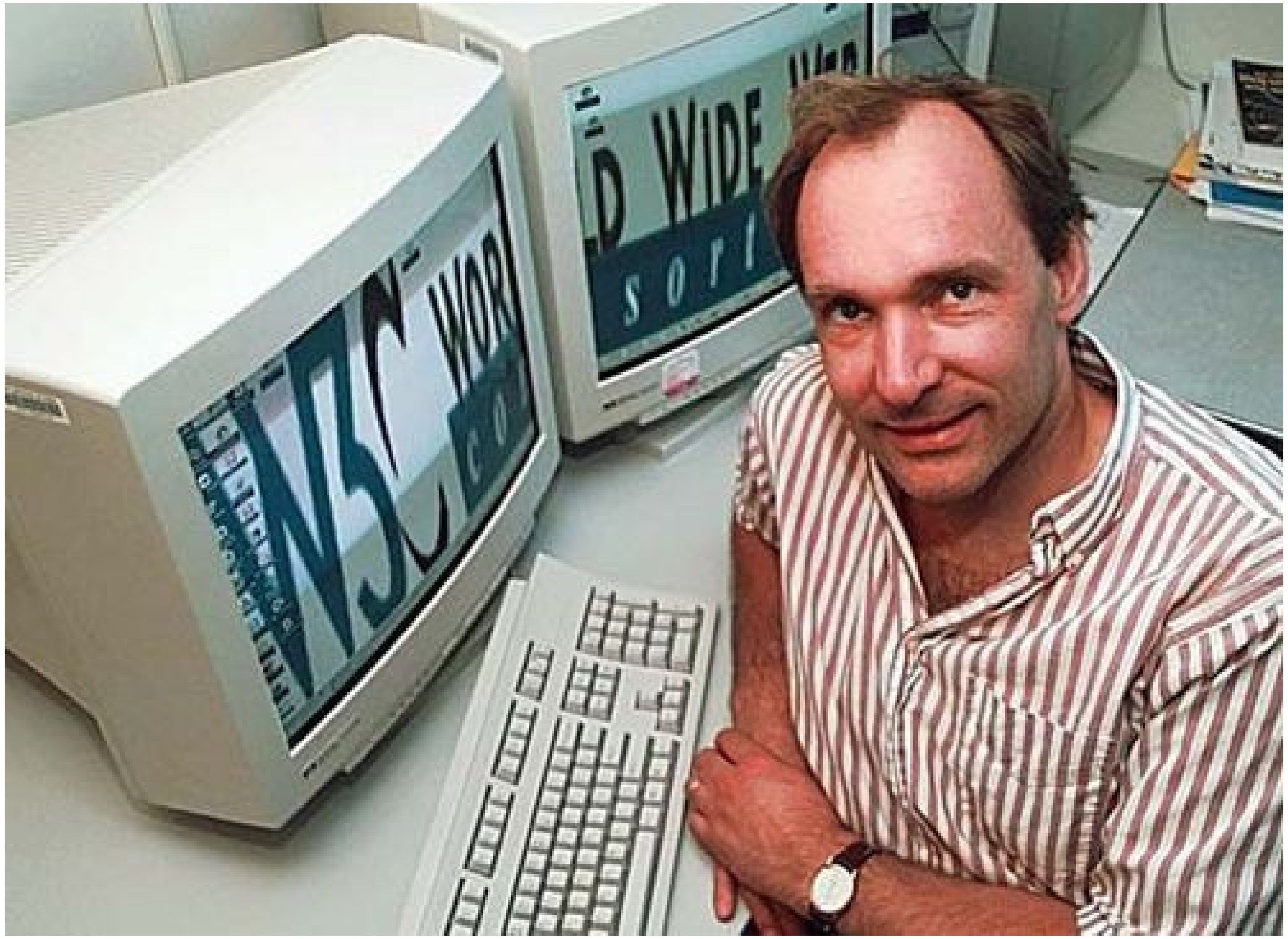
SLACVM Information Service

BINLIST SLAC phone book with e-mail addresses

HEP SPIRES HEP preprint database

Early web page

蒂姆·伯纳斯一李建立的第一个网站(也是世界上第一个网站)是http: //info. cern. ch/，它于1991年8月6日上网。蒂 姆·伯纳斯·李爵士最杰出的成就，是免费把万维网的构想推广到全世界，让万维网科技获得迅速的发展，深深改变了人类的生活面貌。



伦敦2012奥运会开幕式上，开幕式总导演丹尼·博伊尔特别为表扬蒂姆·伯纳斯·李爵士的功绩，设计了激动人心的一幕：蒂姆·伯纳斯·李爵士在“伦敦碗”场馆中央用电脑键盘敲出了一句话：This Is For Everyone(为了每一个人)。

“This is For Everyone”一语双关，既表达了伦敦奥运的全民参与性，也反映了“WWW”全球互联网的免费特质。

Introduction to HTML

HTTP

•HTTP stands for HyperText Transport Protocol, and is the network protocol used over the web. It runs over TCP/IP.

•HTTP uses a request/response model. Client sends an HTTP request, then the server gives back the HTTP response that the browser will display (depending on the content type of the answer).

•If the response is an HTML file, then the HTML file is added to the HTTP response body.

•An HTTP response includes : status code, the content-type (MIME type-Multipurpose Internet Mail Extensions), and actual content of the response.

•A MIME type tells the browser what kind of data the response is holding.

HTTP METHODS

METHOD	DESCRIPTION
HEAD	Asks for the response identical to the one that would correspond to a GET request, but without the response body. This is useful for retrieving meta-information written in response headers, without having to transport the entire content.
GET	means retrieve whatever data is identified by the URI
POST	Submits data to be processed (e.g., from an HTML form) to the identified resource. The data is included in the body of the request.
DELETE	Deletes the specified resource.
TRACE	Echoes back the received request, so that a client can see what intermediate servers are adding or changing in the request.
OPTIONS	Returns the HTTP methods that the server supports for specified URL. This can be used to check the functionality of a web server by requesting '*' instead of a specific resource.
CONNECT	Converts the request connection to a transparent TCP/IP tunnel, usually to facilitate SSL-encrypted communication (HTTPS) through an unencrypted HTTP proxy.[3]
PUT	Uploads a representation of the specified resource.

HTTP METHODS

HTTP servers are required to implement at least the GET and HEAD methods.

Safe methods

Some methods (for example, HEAD, GET, OPTIONS and TRACE) are defined as *safe*, which means they are intended only for information retrieval and should not change the state of the server.

Idempotent methods

Some http methods are defined to be idempotent, meaning that multiple identical requests should have the same effect as a single request. Methods PUT, DELETE, GET, HEAD, OPTIONS and TRACE, being prescribed as safe, should also be idempotent. HTTP is a stateless protocol.

By contrast, the POST method is not necessarily idempotent, and therefore sending an identical POST request multiple times may further affect state or cause further side effects.(like updating inserting records in database multiple times).

DIFFERENCE BETWEEN GET AND POST

GET:

GET parameters are passed on the URL line, after a **?**. The URL size is limited.

The parameters appear in the browser URL field, hence showing any password to the world...

GET is supposed to be used to GET things (only retrieval, no modification on the server).

GET processing must be idempotent.

A click on a hyperlink always sends a GET request.

GET is the default method used by HTML forms. Use the method=”POST” to change it.

POST:

POST has a body.

In POST requests, the parameters are passed in the body of the request, after the header. There is no size-limit. Parameters are not visible from the user.

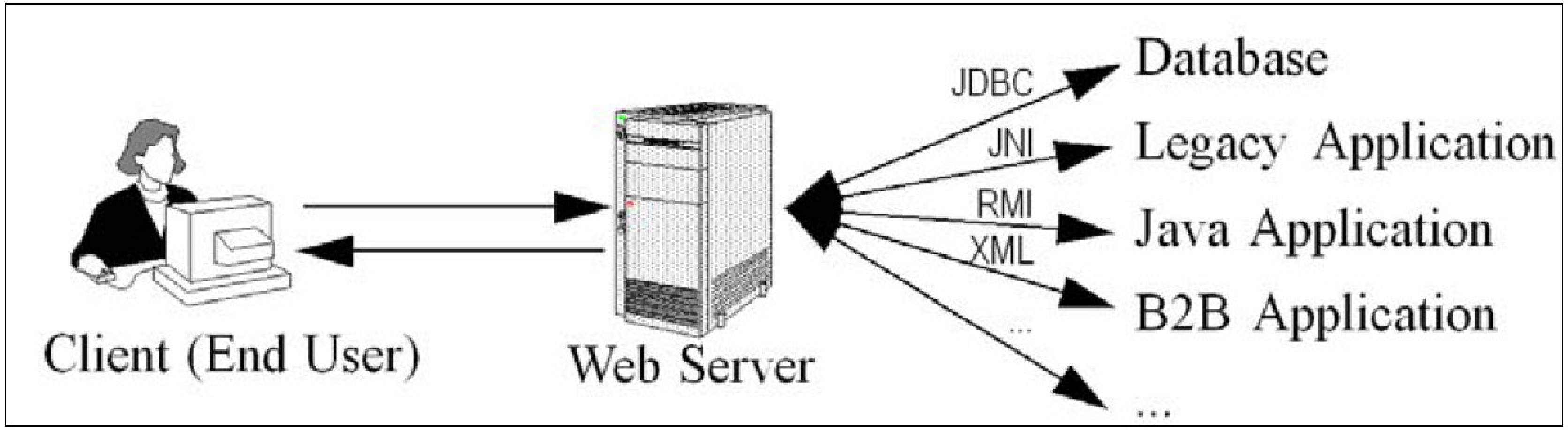
POST is supposed to be used to send data to be processed.

POST may not be idempotent and is **the only one**.

IDEMPOTENT : Can do the same thing over and over again, with no ***negative*** side effect.

Local interactive and remote interactive

- Local interactive—Java Script
- Remote interactive—JSP/Servlet



WEB CONTAINER

A Web application runs within a Web container of a Web server. The Web container provides the runtime environment through components that provide naming context and life cycle management. Some Web servers may also provide additional services such as security and concurrency control.

Tomcat

Web applications are composed of web components and other data such as HTML pages. Web components can be servlets, JSP pages created with the JavaServer Pages™ technology, web filters, and web event listeners. These components typically execute in a web server and may respond to HTTP requests from web clients. Servlets, JSP pages, and filters may be used to generate HTML pages that are an application's user interface.

Why build Web Pages Dynamically?

- **The Web page is based on data submitted by the user**
  - E.g., results page from search engines and order confirmation pages at on-line stores
- **The Web page is derived from data that changes frequently**
  - E.g., a weather report or news headlines page
- **The Web page uses information from databases or other server-side sources**
  - E.g., an e-commerce site could use a servlet to build a Web page that lists the current price and availability of each item that is for sale.

Servlet

- A **servlet** is a Java programming language class used to extend the capabilities of servers that host applications accessed via a request-response programming model. Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by Web servers. Thus, it can be thought of as a Java Applet that runs on a server instead of a browser.

Servlet (Con')

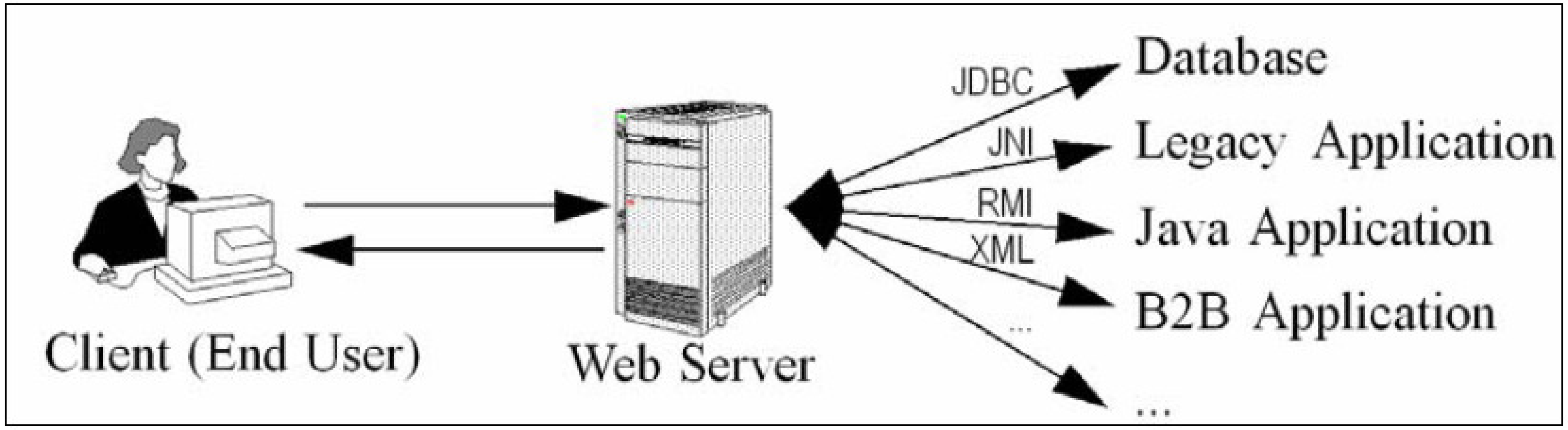
- A **Servlet** is a Java class in Java EE that conforms to the **Java Servlet API**, a protocol by which a Java class may respond to requests. They are not tied to a specific client-server protocol, but are most often used with the HTTP protocol. Thus, a software developer may use a servlet to add dynamic content to a Web server using the Java platform. The generated content is commonly HTML, but may be other data such as XML. Servlets are the Java counterpart to non-Java dynamic Web content technologies such as CGI and ASP.NET. Servlets can maintain state in session variables across many server transactions by using HTTP cookies, or URL rewriting.

HelloServlet

```
import java.io.IOException;
public class HelloServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException,
        IOException {
            PrintWriter out=response.getWirter();
            out.println("<html>\n"+
                "<head><title>hello</title></head>" +
                "<h1>high</h1>\n"+
                "</body></html>");
    }
}
```

A Servlet's Job

- Read implicit data sent by client (request headers)
- Generate the results
- Send the explicit data back to client (HTML)
- Send the implicit data to client (status codes and response headers)



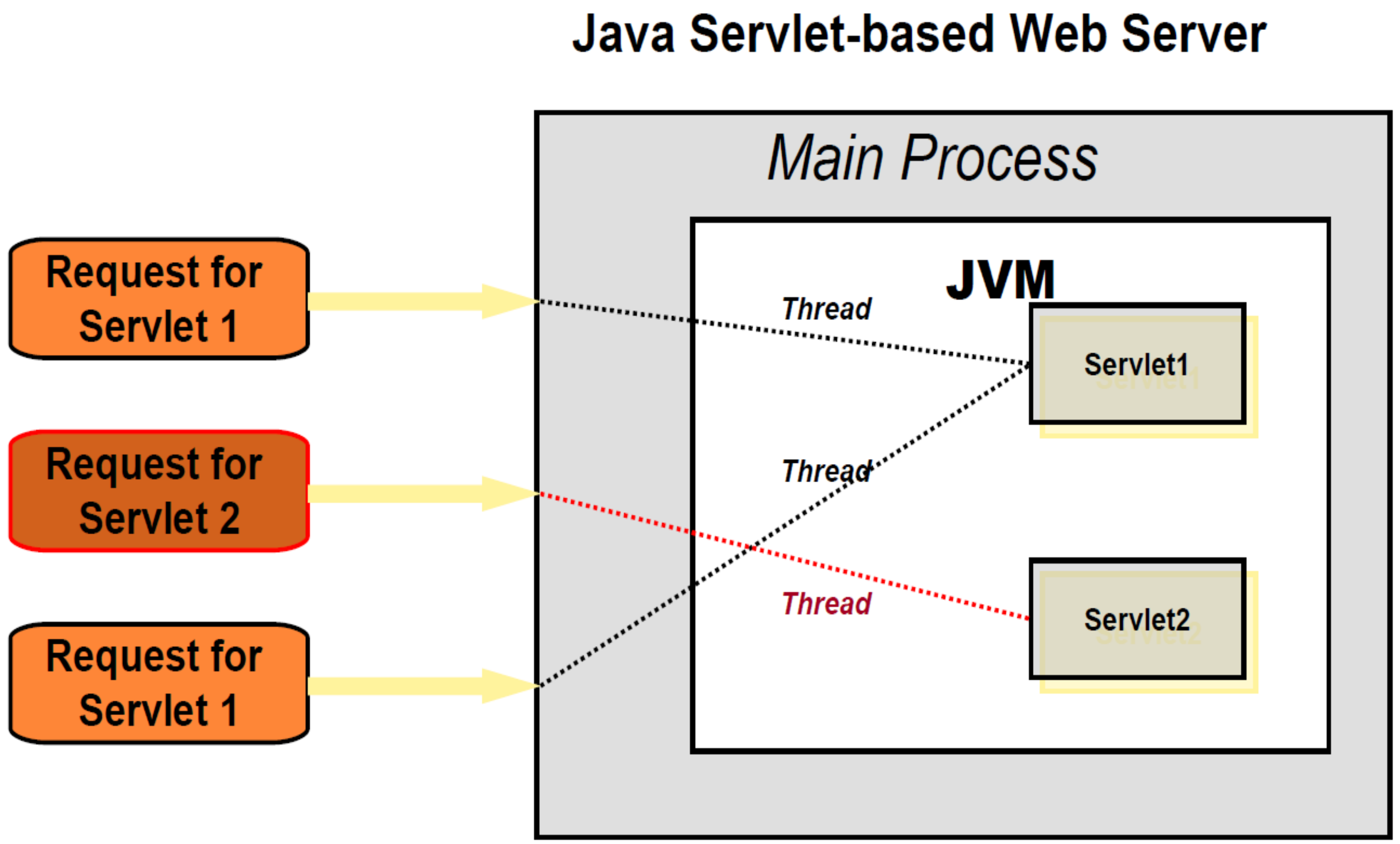
Servlet开发过程

- 继承HttpServlet， 重写doGet(), doPost(), doPut(), doDelete()等方法
- 修改部署web.xml， 重新启动tomcat

修改web.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="WebApp_ID" version="2.5">
 <display-name>FirstWebApp</display-name>
<servlet>
   <description></description>
   <display-name>HelloServlet</display-name>
   <servlet-name>HelloServlet</servlet-name>
   <servlet-class>keg.web</servlet-class>
 </servlet>
 <servlet-mapping>
   <servlet-name>HelloServlet</servlet-name>
   <url-pattern>/HelloServlet</url-pattern>
 </servlet-mapping>
</web-app>
```

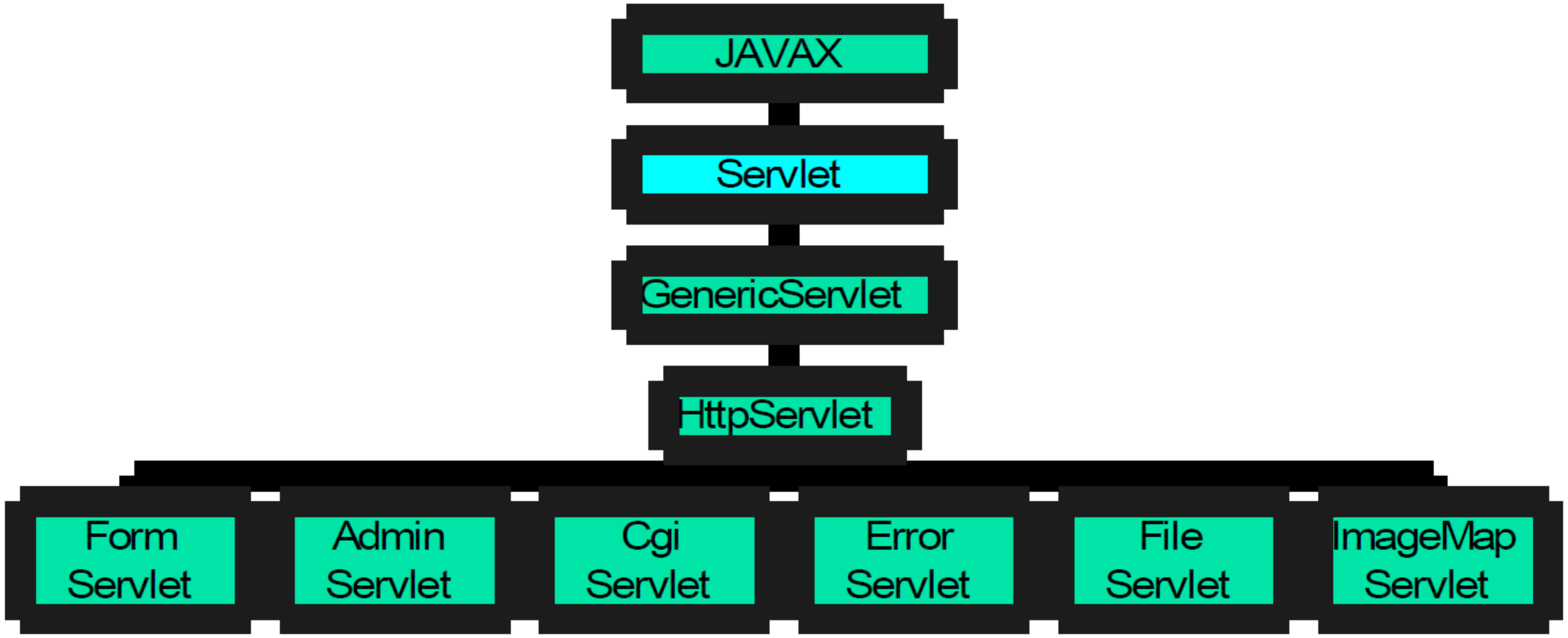
Servlet life cycle



Sevrlet Basics

- Java servlets are the first standard extension to Java, including two packages:
  - javax.servlet
  - javax.servlet.http

Servlet package framework



Servlet Interface

Servlet interface

{

void **init(ServletConfig sc) throws ServletException;**

void **service(ServletRequest req, ServletResponse res)** throws ServletException, IOException;

void **destroy();**

}

Structure of a servlet

- A servlet does not have a main() method.
- Certain methods of a servlet are invoked by the server in the process of handling requests.
- Each time the server dispatches a request to a servlet, it invokes the servlet's service() method.

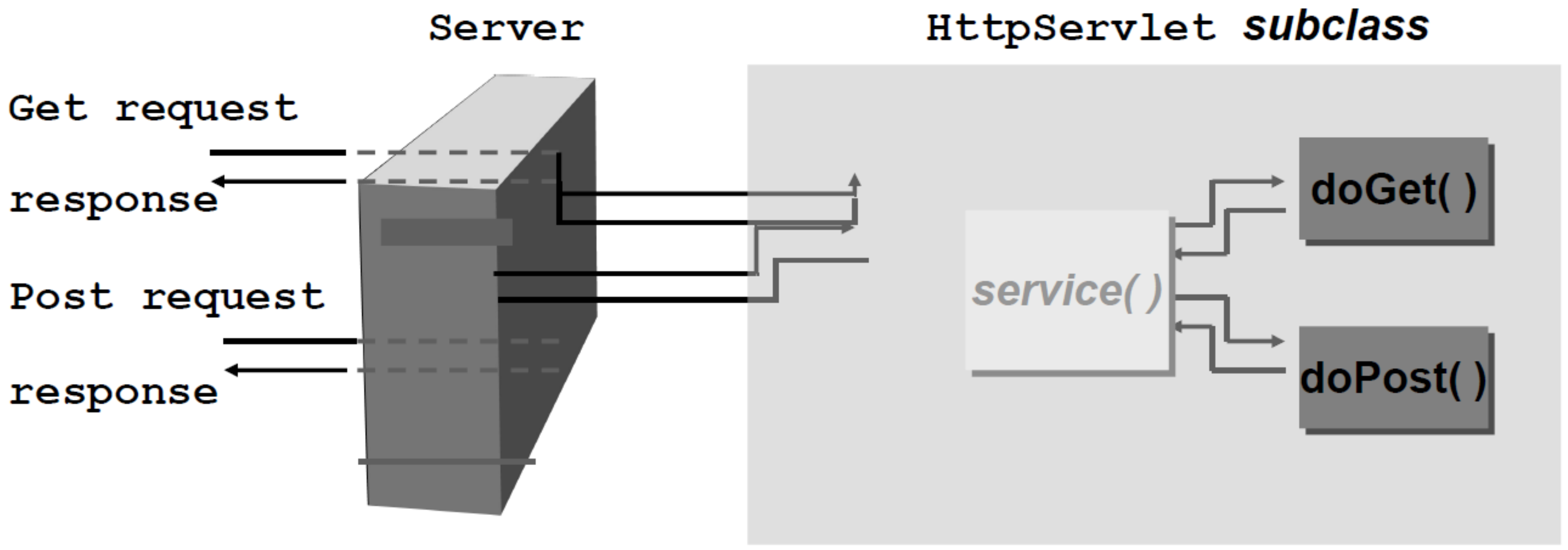
Request / response

- The service() method accepts two parameters:
  - a **request** object: tells the servlet about the request
  - a **response** object: the response object is used to return a response

HttpServlet class

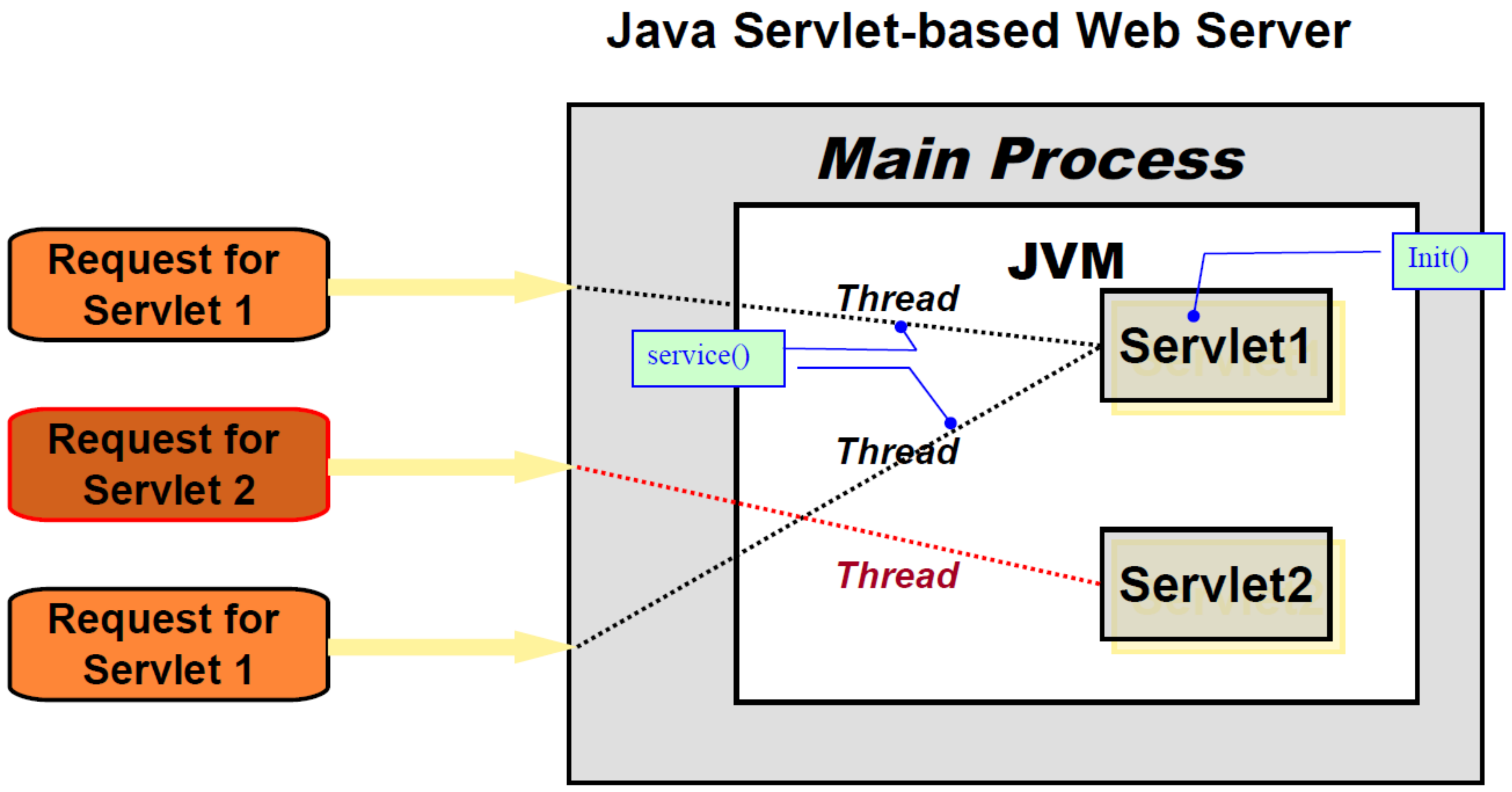
- **Servlet** is a simple Java class which must implement javax.servlet.Servlet interface.
- **GenericServlet** class provides a default implementation of this interface so that we don't have to implement every method of it.
- **HttpServlet** class extends GenericServlet to provide an HTTP protocol specific implementation of Servlet interface.

An Http Servlet handling get and post request



Any subclass overrides the doGet()/doPost method(s), not the service() method

Servlet Life Cycle



HttpServlet methods (1/2)

- init()
  - Called only once during the initialization of the Servlet.
- destroy()
  - Called only once when Servlet instance is about to be destroyed.
- service()
  - Do not override this method!!.
- doGet(), doPost(), doPut(), doDelete(), doOptions(), doTrace()
  - These methods are called according to the type of HTTP request received. Override them to generate your own response.
- log()
  - Writes messages to the Servlet's log files.

HttpServlet methods (2/2)

- getLastModified()
  - Override this method to return your Servlet's last modified date.
- getServletInfo()
  - Override this method to provide a String of general info about your Servlet such author, version, copyright etc.
- getServletName()
  - Override this method to return name of the Servlet.
- getInitParameter(), getInitParameterNames()
  - Return value(s) of initialization parameter(s)
- getServletConfig()
  - Returns a reference to ServletConfig object.
- getServletContext()
  - Returns reference to ServletContext object.

Do not override...

- service()
- getServletConfig()
- getServletContext()

  – The three methods which stated above should not be overridden as their default implementation is more than enough to suffice.

Servlet input processing

- protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException

- A servlet may access the HTTP request information trough the request object
- See: http://docs.oracle.com/javaee/6/api/
  - HttpServletRequest
  - ServletRequest

Request parameters

- **String getParameter(String name)**
  - Returns the value of a request parameter as a String, or null if the parameter does not exist.
- **Map getParameterMap()**
  - Returns a java.util.Map of the parameters of this request.
- **Enumeration getParameterNames()**
  - Returns an Enumeration of String objects containing the names of the parameters contained in this request.
- **String[] getParameterValues(String name)**
  - Returns an array of String objects containing all of the values the given request parameter has, or null if the parameter does not exist.

HTTP headers information (1/2)

- **String getHeader(String name)**
  - Returns the value of the specified request header as a String.
- **Int getIntHeader(String name)**
  - Returns the value of the specified request header as an int.
- **Enumeration getHeaderNames()**
  - Returns an enumeration of all the header names this request contains.
- **Enumeration getHeaders(String name)**
  - Returns all the values of the specified request header as an Enumeration of String objects.

HTTP headers information (2/2)

- **long getDateHeader(String name)**
  - Returns the value of the specified request header as a long value that represents a Date object.
- **String getMethod()**
  - Returns the name of the HTTP method with which this request was made, for example, GET, POST, or PUT.
- **String getQueryString()**
  - Returns the query string that is contained in the request URL after the path.
- **String getRemoteUser()**
  - Returns the login of the user making this request, if the user has been authenticated, or null if the user has not been authenticated.

会话(session)管理

- 会话：有始有终的一系列动作/消息，用来在客户端与服务器之间保持状态的解决方案
- HTTP 协议本身是无状态的
- session机制是一种服务器端的机制
- Cookie在客户端协助进行会话管理，cookie的内容主要包括：名字，值，过期时间，路径和域。

HTTP session handling

- **Cookie[] getCookies()**
  - Returns an array containing all of the Cookie objects the client sent with this request.
- **String getRequestedSessionId()**
  - Returns the session ID specified by the client.
- **HttpSession getSession()**
  - Returns the current session associated with this request, or if the request does not have a session, creates one.
- **boolean isRequestedSessionIdValid()**
  - Checks whether the requested session ID is still valid.

Other interesting information

- **String getRemoteAddr()**
  - Returns the Internet Protocol (IP) address of the client or last proxy that sent the request.
- **String getRemoteHost()**
  - Returns the fully qualified name of the client or the last proxy that sent the request.
- **int getRemotePort()**
  - Returns the Internet Protocol (IP) source port of the client or last proxy that sent the request.
- **String getServerName()**
  - Returns the host name of the server to which the request was sent.
- **int getServerPort()**
  - Returns the port number to which the request was sent.

HttpSession class (1/3)

- **getAttribute(), getAttributeNames(), setAttribute(), removeAttribute()**
  - These methods are used to set, get and remove objects from a user session.
- **getId()**
  - Every session created by the server has a unique 'id’ associated with it in order to identify this session from other sessions. This method returns the 'id' of this session.
- **getCreationTime()**
  - Simple returns a long value indicating the date and time this session was created, meaning there by that you get the time this user first accessed your site.

HttpSession class (2/3)

- **getLastAccessedTime()**
  - Returns a long value indicating the last time user accessed any resource on this server.
- **getMaxInactiveInterval(), setMaxInactiveInterval()**
  - Return and set the maximum inactive interval in seconds for this session respectively. Every session has a maximum inactive interval during which if user doesn't make request to the server, the session is invalidated.
- **isNew()**
  - Returns a boolean value indicating if the session is new. Either it is the first page of the site user has hit so his session is new and has just been created or that user is not accepting cookies required for managing sessions so this value will then always return true.

HttpSession class (3/3)

- **invalidate()**
  - Simply invalidates a session. You can use this method on a 'logout' page allowing user to end his session. If after invalidation of his session user accesses some resource on the site then a new session will be created for it.

ServletConfig class

- ServletConfig object is used to pass information to the Servlet during its initialization
  - Inistialization information is stored in web.xml
- Servlet can obtain information regarding initialization parameters and their values
- using different methods of ServletConfig class.

ServletConfig methods

- **getInitParameter(String paramName)**
  - Returns value of the given parameter. If value of parameter could not be found in web.xml file then a null value is returned.
- **getInitParameterNames()**
  - Returns an Enumeration object containing all the names of initialization parameters provided for this Servlet.
- **getServletContext()**
  - Returns reference to the ServletContext object for this Servlet. It is similar to getServletContext() method provided by HttpServlet class.
- **getServletName()**
  - Returns name of the Servlet as provided in the web.xml file or if none is provided then returns complete class path to the Servlet.

JSP-JavaServer Pages

JSP-JavaServer Pages

- **JavaServer Pages** (**JSP**) is a Java technology that helps software developers serve dynamically generated web pages based on HTML, XML, or other document types. Released in 1999 as Sun's answer to ASP and PHP, JSP was designed to address the perception that the Java programming environment didn't provide developers with enough support for the Web.
- To deploy and run, a compatible web server with servlet container is required. The Java Servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems and the JCP (Java Community Process) must both be supported by the container.

Example of JSP

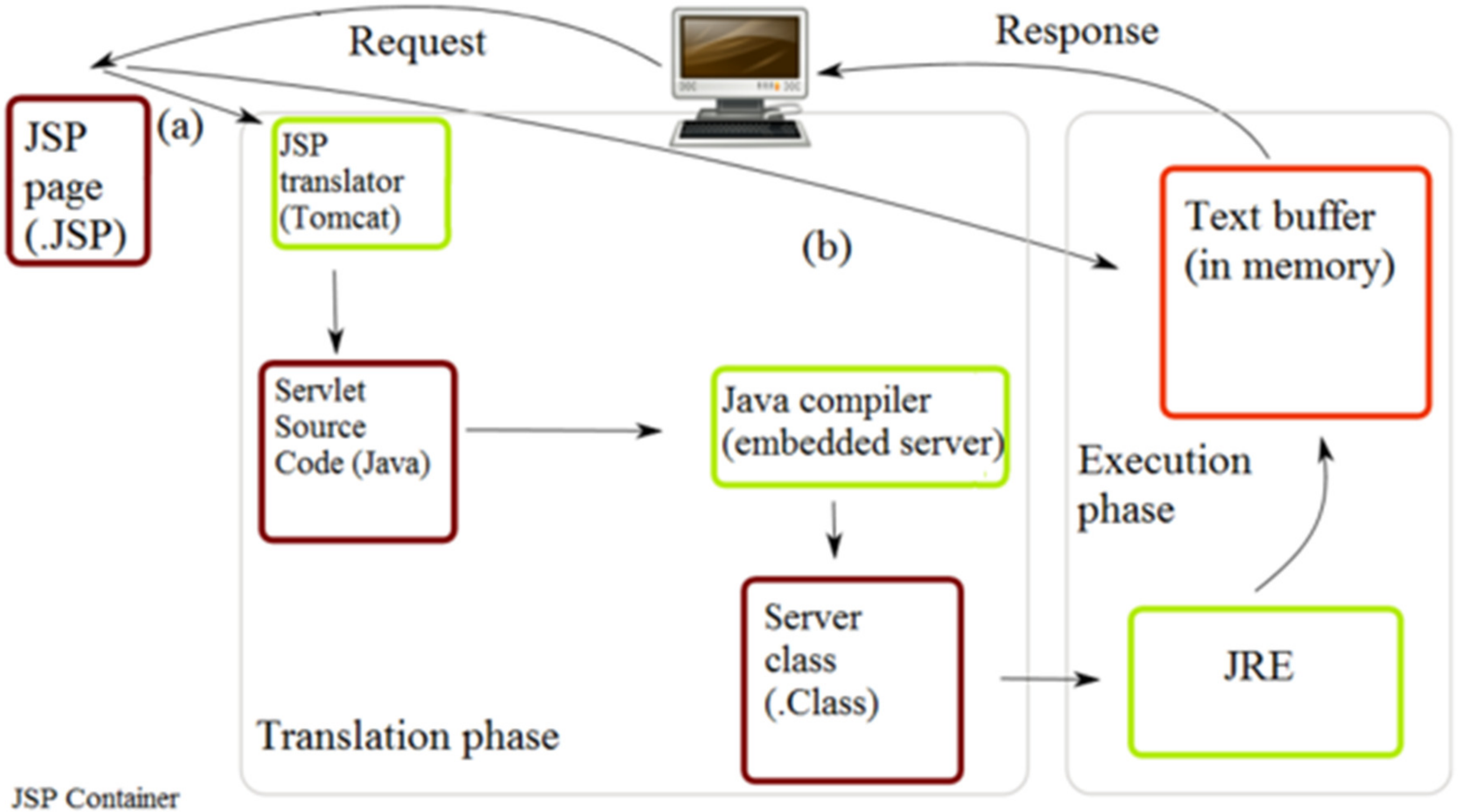
```
<%@ page errorPage="myerror.jsp" %>
<%@ page import="com.foo.bar" %>

<html>
<head>
<%! int serverInstanceVariable = 1;%>

<% int localStackBasedVariable = 1; %>
<table>
<tr><td><%= toStringOrBlank( "expanded inline data " + 1 ) %></td></tr>
```

Differences between Servlet and JSP

- Servlet embeds HTML into Java code (generated HTML)
- JSP embeds Java code into HTML
- JSP will be translated to Servlet for running.

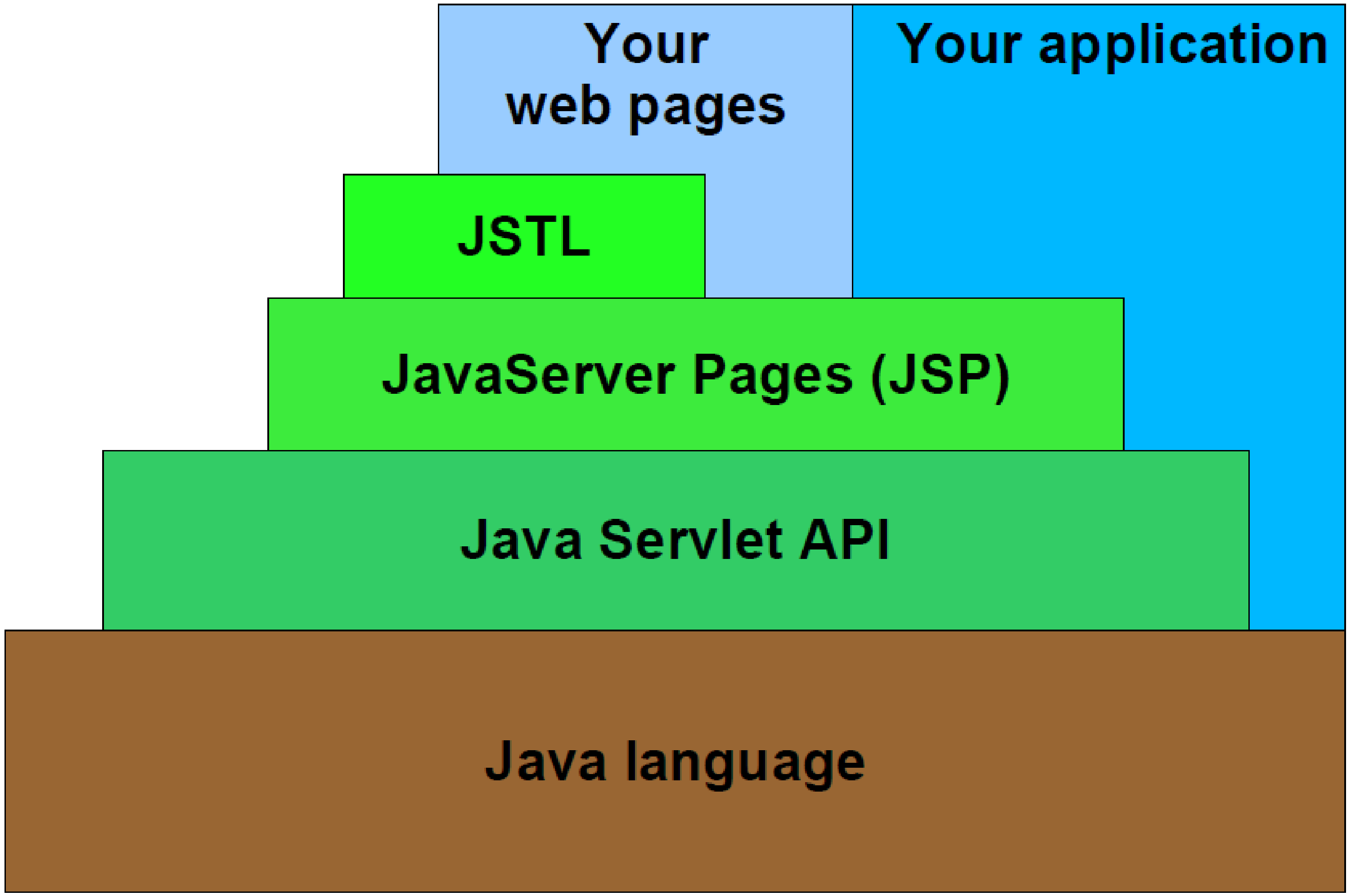


JSP Container
(a) Translation occurs at this point, if JSP has been changed or is new.
(b) If not, translation is skipped.

The Java EE presentation tier

- Servlets
  - Java classes that handle requests by producing responses (e.g., HTTP requests and responses)
- JavaServer Pages (JSP)
  - HTML-like pages with some dynamic content.
  - Translated into servlets automatically
- JSP Standard Tag Library (JSTL)
  - Set of standard components for JSP
  - Used inside JSP pages.

Organization of the platform



Why use JSP?

- Convenient:
  - We already know Java and HTML!
- Provides an extensive infrastructure for:
  - Tracking sessions.
  - Managing cookies.
  - Reading and sending HTML headers.
  - Parsing and decoding HTML form data.
- Efficient:
  - Every request for a JSP is handled by a simple Java thread.

Why use JSP?

- Portable
  - JSP follow a well standardized API.
  - The Java VM which is used to execute a JSP file is supported on many architectures and operating systems.
- Inexpensive
  - There are a number of free or inexpensive Web Servers that are good for commercial-quality websites.

What is JSP?

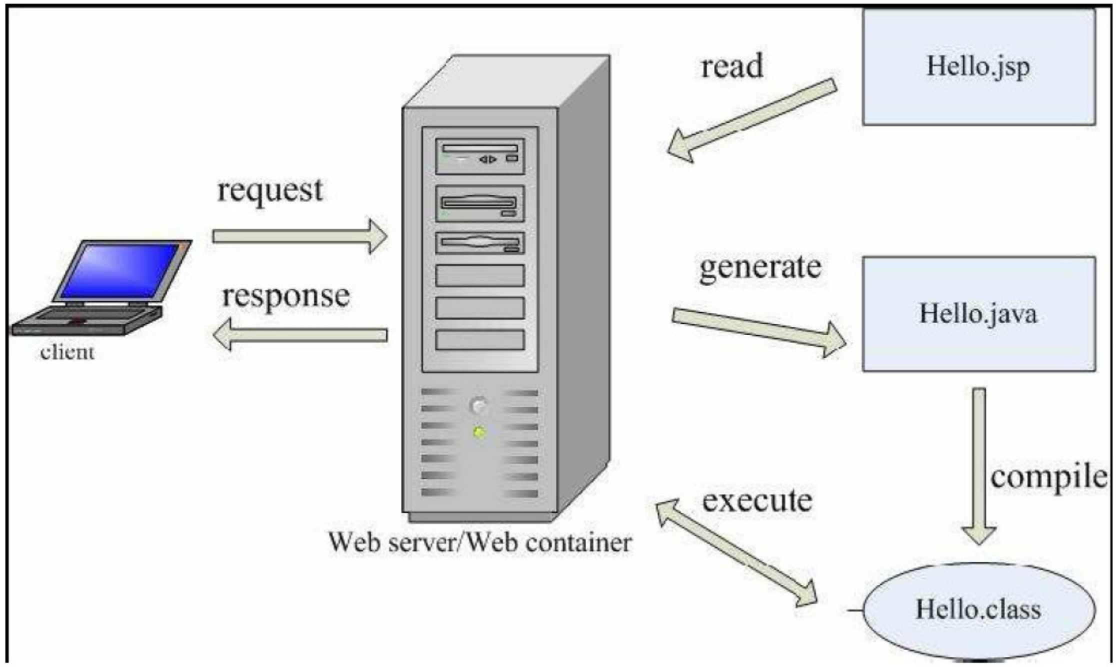
- Java based technology that simplifies the developing of dynamic web sites
- JSP pages are HTML pages with embedded code that allows to access data from Java code running on the server
- JSP provides separation of HTML presentation logic from the application logic.

JSP Technology

- JSP technology provides a way to combine the worlds of HTML and Java servlet programming.
- JSP specs are built on the Java Servlet API.
- JSP supports two different styles for adding dynamic content to web pages:
  - JSP pages can embed actual programming code (typically Java)
  - JSP supports a set of HTML-like tags that interact with Java objects on the server (without the need for raw Java code to appear in the page)

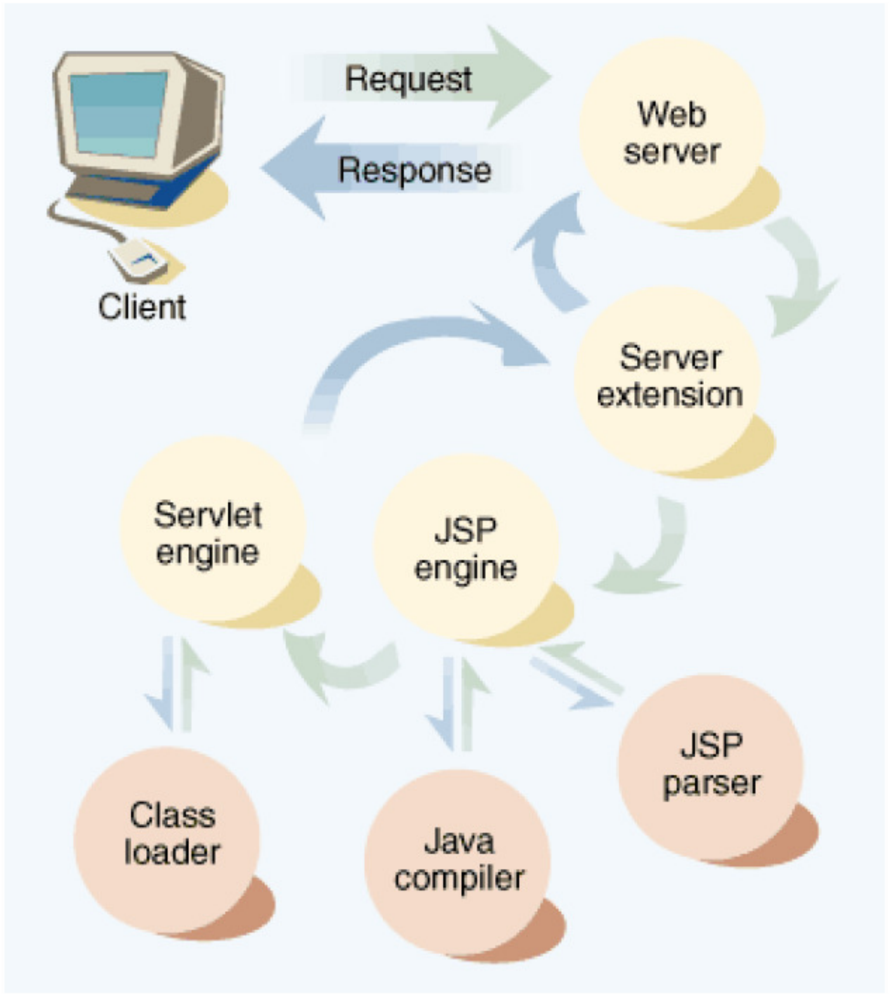
JSP Flow..

- JSP pages “live” within a container that manages its interaction:
- HTTP Protocol (request, response, header) Sessions



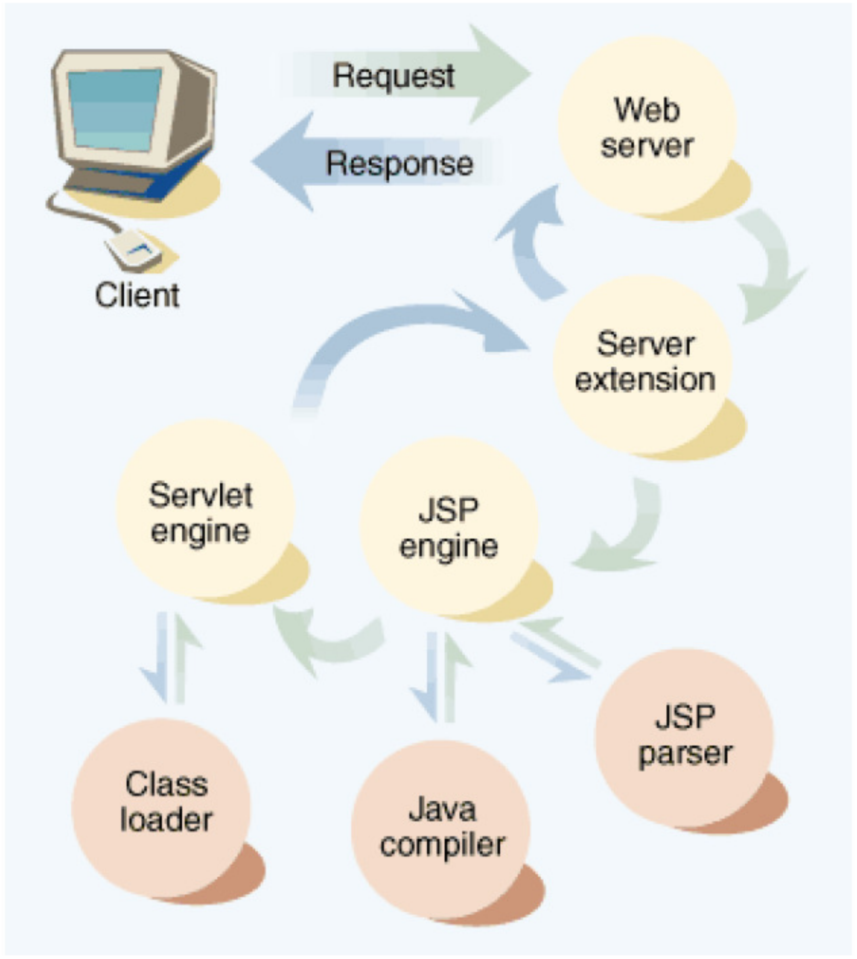
How it really works... (1/2)

- Client requests a page ending with ".jsp"
- Web Server fires up the JSP engine
- JSP engine checks whether JSP file is new or changed
- JSP engine converts the page into a Java servlet (JSP parser)
- JSP engine compiles the servlet (Java compiler)



How it really works... (2/2)

- Servlet Engine executes the new Java servlet using the standard API
- Servlet's output is transferred by Web Server as a http response



Structure of a JSP file.

- Similar to a HTML document. Four basic tags:
  - Scriplet
  - Expression
  - Declaration
  - Definition

```
<html>
  <head>
    <title>Hello World</head>
  </head>
<body>
<h1>Hello, World</h1>
It's <%=(new java.util.Date()).toString()%>
and all is well.
</body>
</html>
```

Scriptlet Tag

- Two forms:
  - <% any java code %>
  - <jsp:scriptlet> ... </jsp:scriptlet>
    - (XML form)
- Embeds Java code in the JSP document that will be executed each time the JSP page is processed.
- Code is inserted in the service() method of the generated Servlet

```
<html>
<body>
   <% for (int i = 0; i < 2; i++) { %>
      <p>Hello World!</p>
   <% } %>
</body>
</html>
```

```
<html>
<body>
   <p>Hello World!</p>
   <p>Hello World!</p>
</body>
</html>
```

Expression Tag

- <%= expr %>
- <jsp:expression> expr </jsp:expression>
- Expression expr is evaluated (in Java) and its value is placed in the output.
  - Note: no semi-colon “;” following expr

```
<html>
<body>
<p>
<%= Integer.toString( 5 * 5 ) %>
</p>
</body>
</html>
```

```
html>
<body>
       <p>25</p>
</body>
</html>
```

(Embedded) Expression language

- An EL expression always starts with a ${ and ends with a }
- All EL expressions are evaluated at **runtime**
- The EL usually handles data type conversion
- The expression can include
  - literals (“1”, “100” etc)
  - variables
  - implicit variables

Examples

- ${1+2+3}
- ${param.Address}

EL Operators

```
==     !=     <      >      <=     >=
+      /      div    -      *
&&     and    ||     or     !      not
empty
```

JSP Implicit Objects

- In JSP, need to be able to access information about the environment in which the page is running e.g. the parameters passed in a request for a form, the browser type of the user etc.
- Implicit objects are a set of Java objects that the JSP Container makes available to developers in each page. These objects may be accessed as built-in variables via scripting elements
- The JSTL EL allows these objects to be accessed as ‘Implicit Variables’
- Implicit variable are just pre-agreed fixed variable names that can be used in JSTL Expressions
  – Think of as “variables that are automatically available to your JSP page”

Implicit Objects in Expression language

- Very common implicit object is param
  - param refers to parameter passed in a request message (e.g. information entered into a form by a user).
- Example
  - ${param.userName}

Declaration Tag

- <%! declaration %>
- <jsp:declaration> declaration(s)</jsp:declaration>
- Embeds Java declarations inside a JSP document
- Code is inserted in the body of the servlet class, outside the service method.
  - May declare instance variables
  - May declare (private) member functions

```
<html>
<body>
<%! private int accessCount = 0; %>
   <p> Accesses to page since server reboot:
   <%= ++accessCount %> </p>
</body>
</html>
```

Warning!

- JSP declarations add variables in the servlet instance class
  - Variables shared by all threads (all requests to the same servlet)
  - Until servlet container unloads servlet
  - Beware simultaneous access! Must use synchronized methods

```
<html>
<body>
<%! private int accessCount = 0; %>
   <p> Accesses to page since server reboot:
   <%= ++accessCount %> </p>
</body>
</html>
```

Directive Tag

- <%@ directive att="value" %>
- <jsp:directive.page att="val" />
- Directives are used to convey special processing information about the page to the JSP container.
  - page directive
  - include directive

```
<%@ page import="java.util.*" %>
<%@ page contentType="text/xml" %>
<%@ page errorPage="error.jsp" %>
```

The JSP @page Directive

- import="package.class" or

  import="pkg.class1,...,pkg.classN“
  - This lets you specify what packages should be imported. The import attribute is the only one that is allowed to appear multiple times.
  - Example: <%@ page import="java.util.*" %>
- contentType="MIME-Type" or

  contentType="MIME-Type;

  charset=Character-Set“
  - Specifies the MIME type of the output. Default is text/ html.
  - Example: <%@ page contentType="text/plain" %> equivalent to <% response.setContentType("text/plain"); %>

The JSP @page Directive

- session="true|false“
  - A value of true (the default) indicates that the predefined variable session (of type HttpSession) should be bound to the existing session if one exists, otherwise a new session should be created and bound to it.
  - A value of false indicates that no sessions will be used, and attempts to access the variable session will result in errors at the time the JSP page is translated into a servlet.

The JSP @page Directive

- errorPage="url“
  - This specifies a JSP page that should process any Throwables thrown but not caught in the current page.
- isErrorPage="true|false“
  - This indicates whether or not the current page can act as the error page for another JSP page. The default is false.

The JSP @page Directive

- isThreadSafe="true|false“
  - A value of true (the default) indicates normal servlet processing, where multiple requests can be processed simultaneously with a single servlet instance, under the assumption that the author synchronized access to instance variables.
  - A value of false indicates that the servlet should implement SingleThreadModel, with requests either delivered serially or with simultaneous requests being given separate servlet instances.
    - Don’t use it, since it reduces performance!

The JSP @page Directive

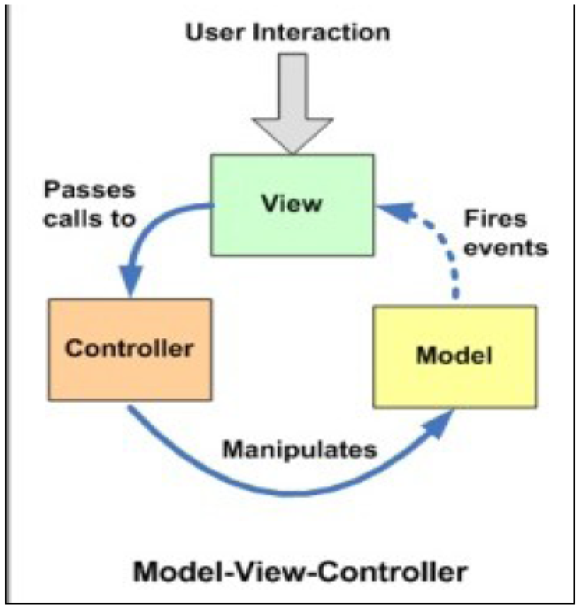
- buffer="sizekb|none“
  - This specifies the buffer size for the jspWriter out. The default is server-specific, but must be at least 8kb.
- autoflush="true|false“
  - A value of true, the default, indicates that the buffer should be flushed when it is full.
  - A value of false, rarely used, indicates that an exception should be thrown when the buffer overflows.
  - A value of false is illegal when also using buffer="none".

The JSP @page Directive

- extends="package.class“
  – This indicates the superclass of servlet that will be generated. Use this with extreme caution, since the server may be using a custom superclass already.
- info="message“
  – This defines a string that can be retrieved via the getServletInfo method.
- language="java“
  – Java is both the default and the only legal choice.

MVC design pattern

- A web application:
  - Collects data and action requests from users…
  - …elaborates/stores them…
  - …visualize the results

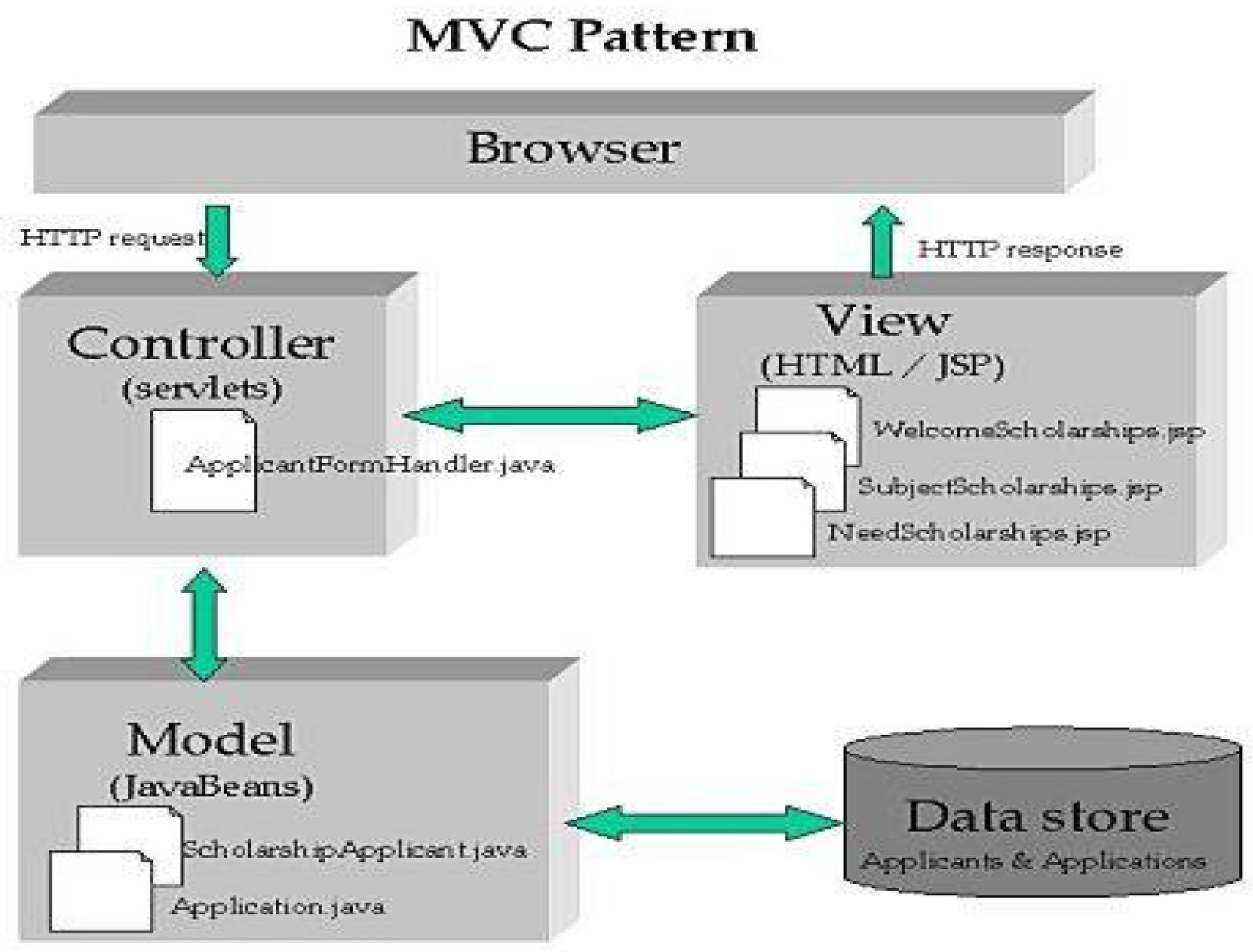


- MVC – Model View Controller paradigm
- The model represents the current state of the applications (with respect to a finite state machine)
- The view corresponds to a presentation of the state
- The controller verifies collected data and updates the model

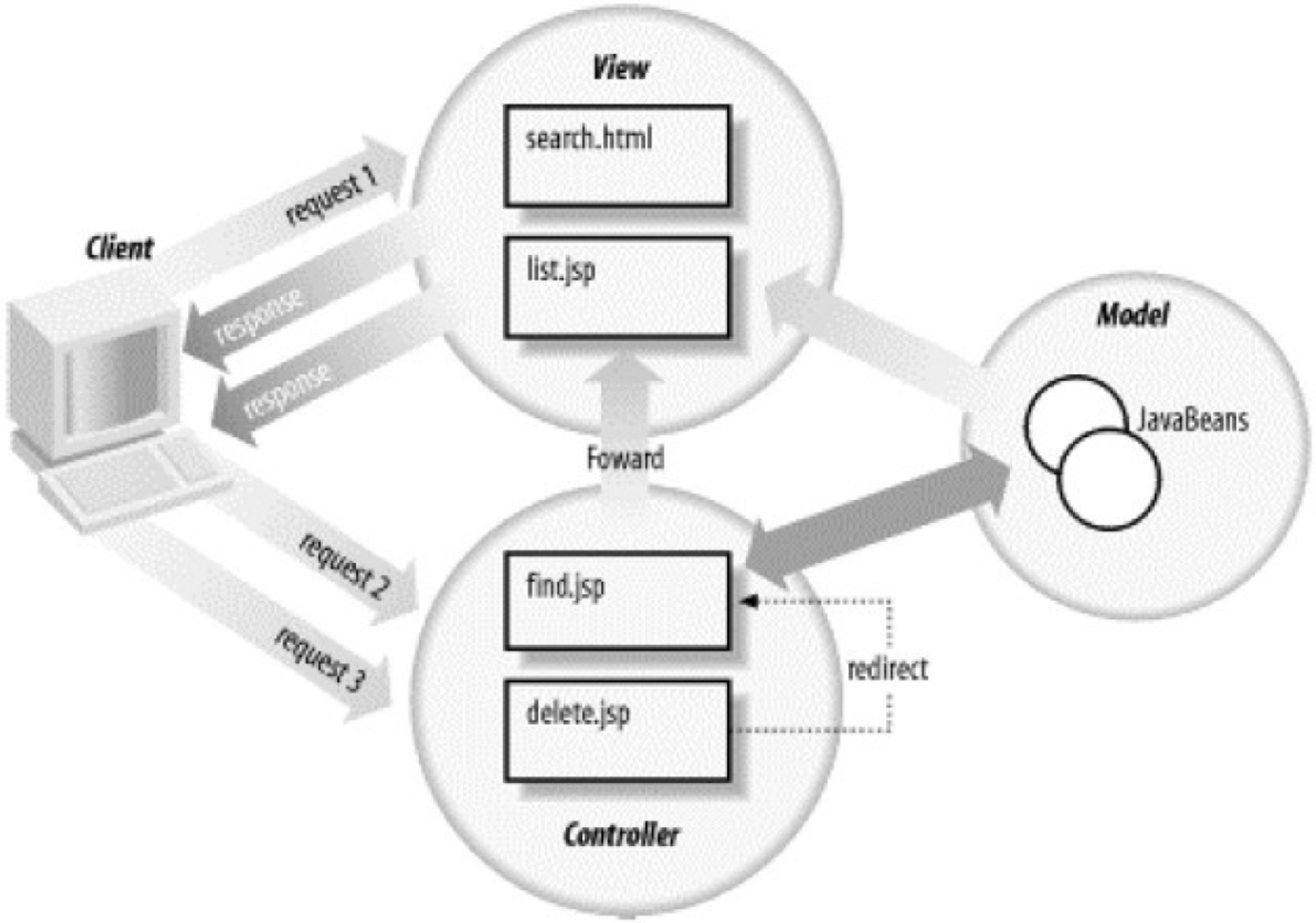
MVC

- Applications that present lots of data to the user, often wish to separate data (Model) and user interface (View) concerns
- Changing the user interface do not impact the data handling, and that the data can be reorganized without changing the user interface.
- The MVC design pattern solves this problem by decoupling data access and business logic from data presentation and user interaction.

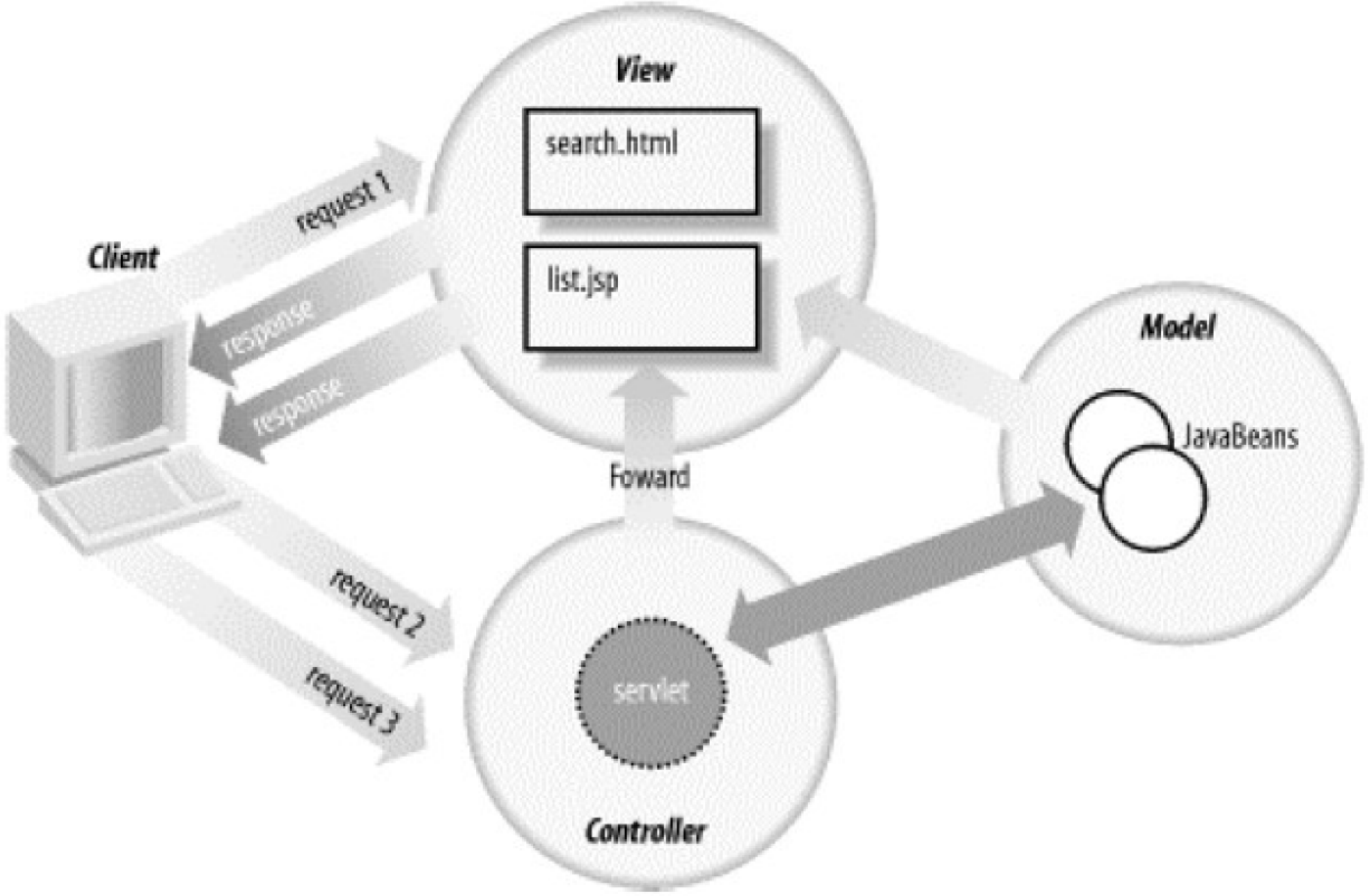
MVC in the Java Server architecture



MVC with JSP only



MVC with JSP and servlets



小结

- Web发展的历史
- HTML tag
- HTTP Methods
- JSP
- Servlet

考试时间地点

- 时间：9月1日晚上7:20开始
- 地点：六教6C300
- 考试内容：前七讲
- 考试形式：填空、选择（多选）

考试样题1－选择题

- 下面哪些是Object的方法？
  - read
  - write
  - clone
  - toString

考试样题2—选择题

给出下面代码：

```
public class Person{
  int arr[] = new int[9];
  public static void main(String args[]) {
    System.out.println(arr[10]);
  }
}
```

下面哪个说法是正确的？

A 编译时将产生错误　B. 编译时正确，运行时将产生错误

C. 输出零　　　　　　D. 输出空

考试样题2—答案

给出下面代码：

```
public class Person{
  int arr[] = new int[9];
  public static void main(String args[]) {
    System.out.println(arr[10]);
  }
}
```

下面哪个说法是正确的？

A 编译时将产生错误　B. 编译时正确，运行时将产生错误

C. 输出零　D. 输出空

考试样题2— 修改

给出下面代码：

```
public class Person{
  static int arr[] = new int[9];
  public static void main(String args[]) {
    System.out.println(arr[10]);
  }
}
```

下面哪个说法是正确的？

A 编译时将产生错误　B. 编译时正确，运行时将产生错误

C. 输出零　D. 输出空

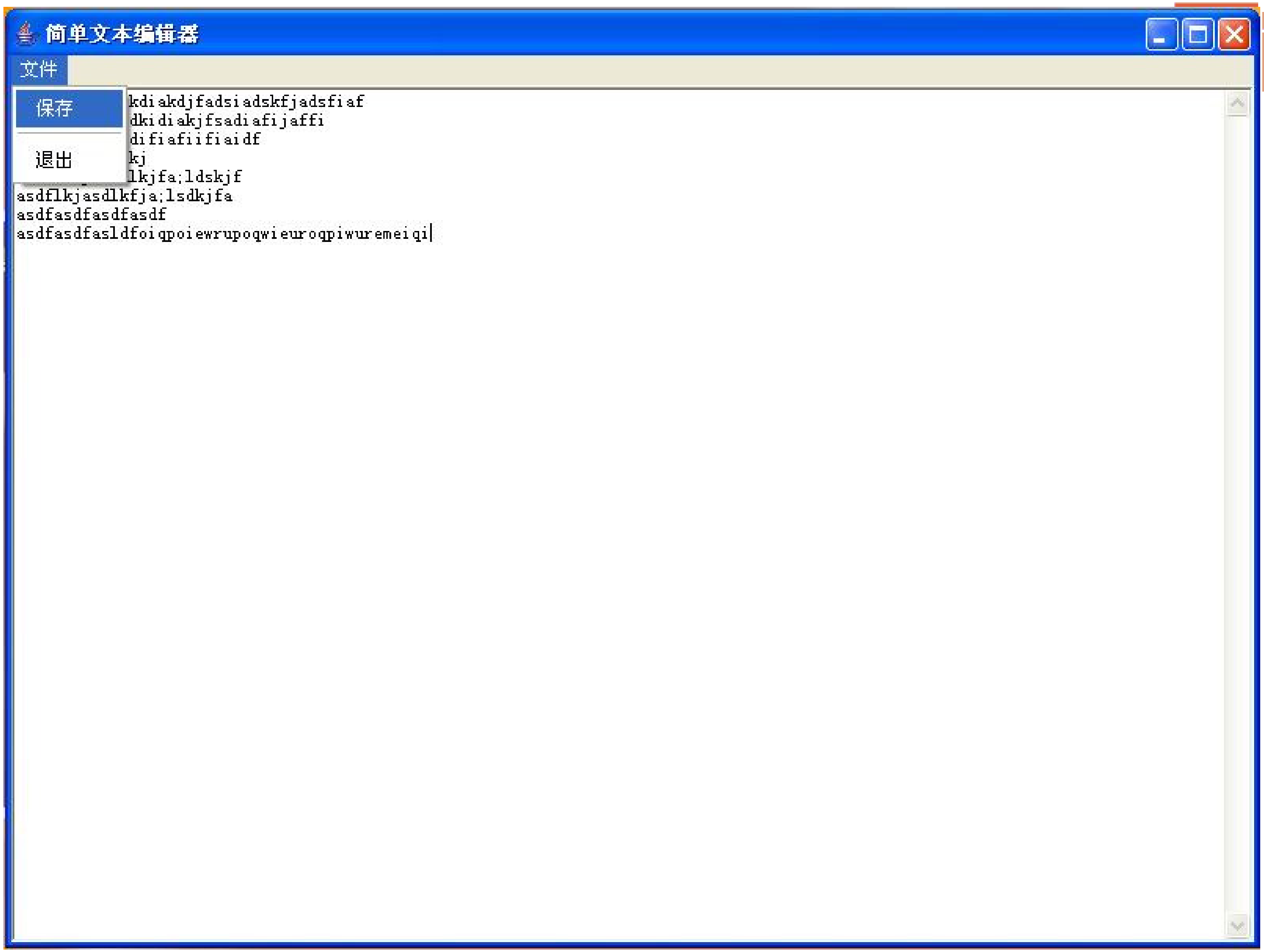
考试样题3－填空题

接口ActionListener中的方法是：

____________________（）___

考试样题4——编程

- 完成如下图所示的一个简单的文本编辑器的应用程序。功能要求如下：标题为“简单文本编辑器”；“文件”菜单下有“保存”子菜单和“退出”子菜单；“保存”子菜单可以弹出文件对话框选择文件名，并将文本编辑器中输入的字符保存到该文件中；“退出”子菜单可以关闭窗口；点击窗口的关闭按钮也可以关闭窗口。



```
kdiakdjfadsiadskfjadsfiaf
dkidiakjfsadiafijaffi
difiafiifiaidf
kj
lkjfa;ldskjf
asdflkjasdlkfja;lsdkjfa
asdfasdfasdfasdf
asdfasdfasldfoiqpoiewrupoqwieuroqpiwuremeiqi
```

```
import java.awt.*;
import java.awt.event.*;
import java.io.*;
public class SimpleTextEditor {
  private Frame f;
  private MenuBar mb;
  private Menu mFile;
  private MenuItem mSave,mExit;
  private FileDialog fd;
  private String fileName;
  private char ch;
  private String s="";
  private TextArea ta;
private MenuListener menuListener;
 private FileOutputStream fos;
 private OutputStreamWriter osw;
 public SimpleTextEditor(){
   f=new Frame("______________");
   mb=new MenuBar();
   mFile=new Menu(____);
   mSave=new MenuItem(____);
   mExit=new MenuItem(____);
```

```
f.setMenuBar(___);
  mb.add(____);
  mFile.add(mSave);
  mFile.addSeparator();
  mFile.add(mExit);
  ta=new TextArea("",10,10);
  f.add("Center",ta);
  ta.setForeground(Color.black);
  ta.addKeyListener(new KeyAdapter(){
        public void keyTyped(KeyEvent ev) {
              s+=ev.getKeyChar();
        }
  });
```

```
f.setSize(800,600);
  f.setVisible(true);
  menuListener=new MenuListener();
  mExit.addActionListener(________);
  mSave.addActionListener(________);
  f.addWindowListener(new WindowAdapter(){
    public void windowClosing(WindowEvent e){
      ________; //正常关闭窗口
    }
  });
}
```

```
public static void main(String args[]){
  SimpleTextEditor te=new SimpleTextEditor();
 }
 class MenuListener implements ActionListener{
  public void actionPerformed(ActionEvent ev){
        MenuItem i=(MenuItem)ev.getSource();
        String label=i.getLabel();
        if (label=="退出")   _______; //正常关闭窗口
              else if (label=="保存") saveFile();
  }
 }
```

```
void saveFile(){
  fd=new FileDialog(f,"请输入要保存的文件名
   ",FileDialog.SAVE);
  fd.setVisible(_______);
  fileName=fd.getFile();
  try{fos=new FileOutputStream(fileName);
     osw=new OutputStreamWriter(fos);
     System.out.println(s);
     osw.write(s);
     osw.flush();
     osw.close();
     fos.close();
  }catch(Exception e){}
 }
}
```

预祝大家取得好成绩！