Java Server Pages (JSP)

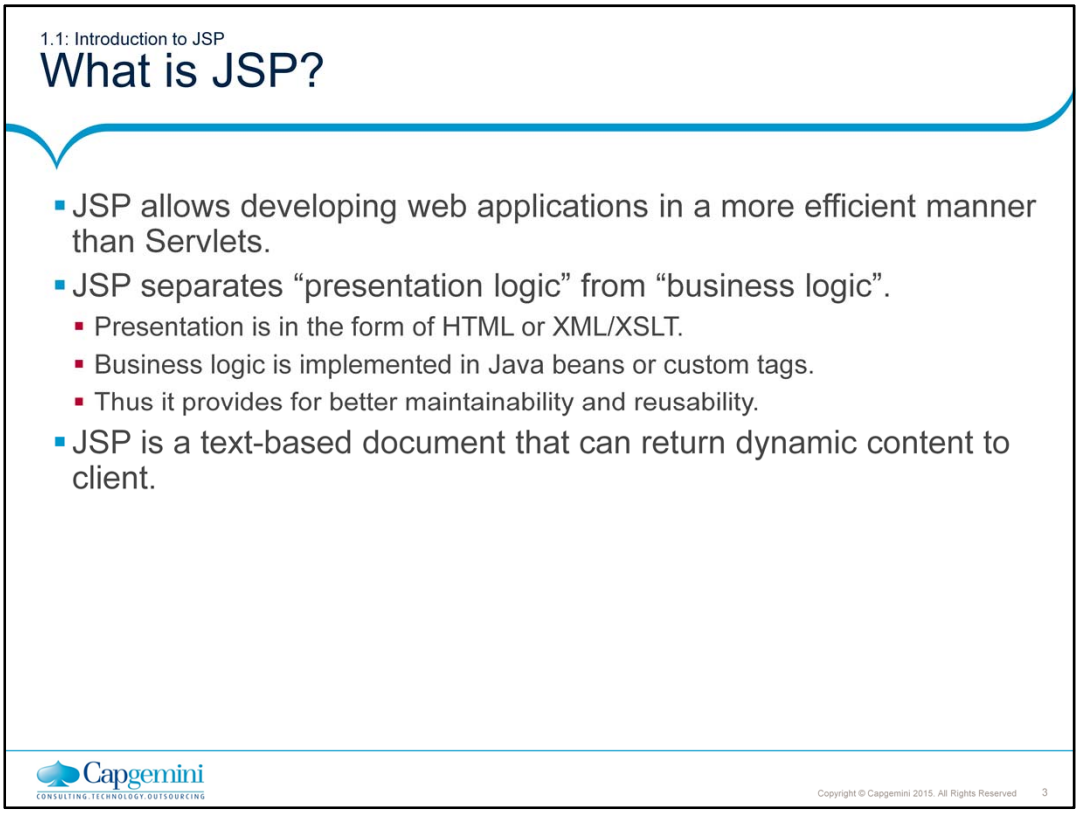
Lesson 01: Introduction to JSP 2.2

Lesson Objectives

- In this lesson, you will learn:
  - Introduction to Java Server Pages (JSP)
  - Features of JSP 2.2
  - Access Models
    - JSP Model 1 Architecture
    - JSP Model 2 Architecture
  - Advantages of JSP over competing technologies

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

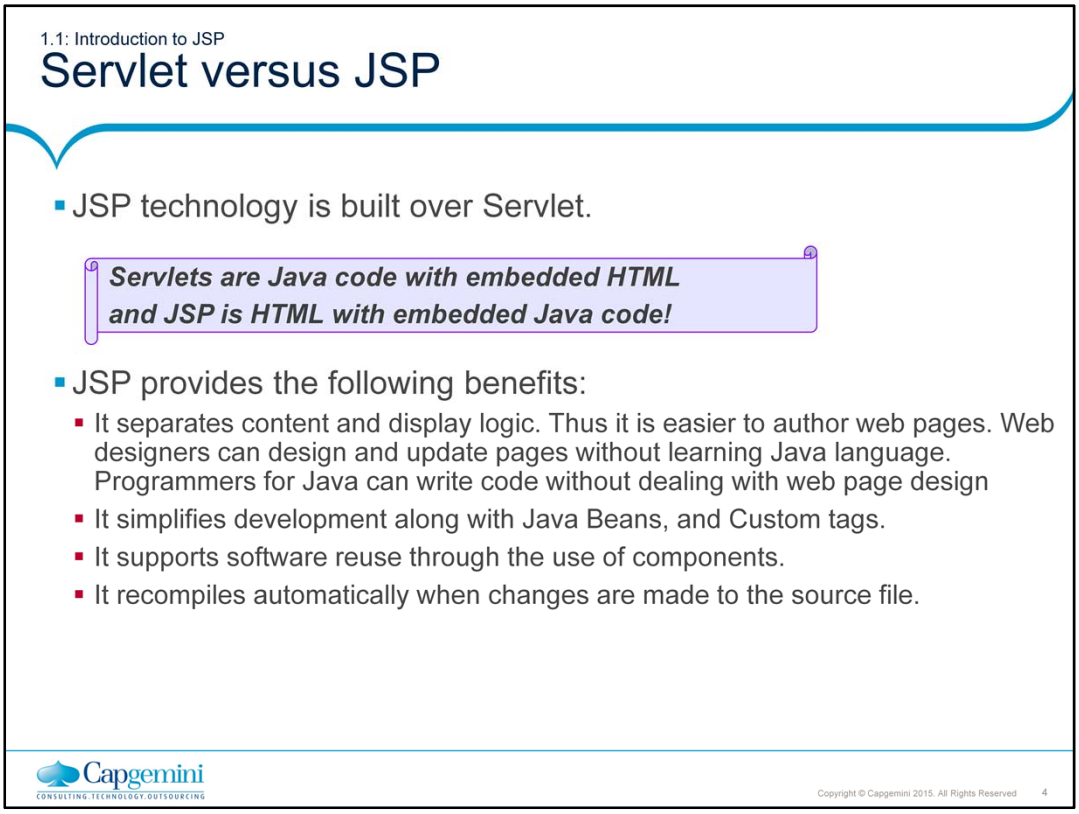
 2



Introduction to JSP:

What is JSP?

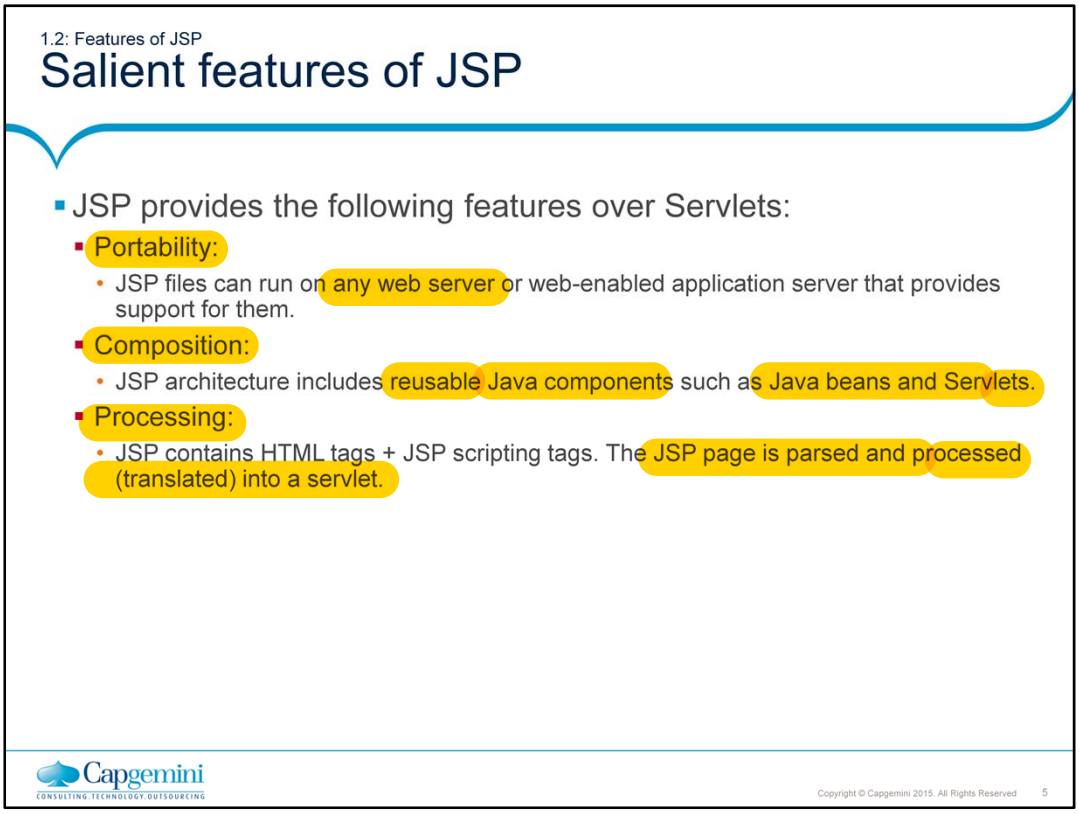
- **Java Server Pages (JSP)** is a simple, yet powerful technology for creating and maintaining dynamic-content webs pages. Based on the Java programming language, JSP offers proven portability, open standards, and a mature re-usable component model.
- JSP was introduced as a follow-on technology to Servlets.
  - **Downside of Servlet:** Servlet needs to provide many print statements to generate HTML response. Any change to the presentation means recompilation and re-deployment!
- JSP separates “presentation logic” from “business logic” and thus enables the Web page designer and Web Application developer to work independently! Besides, the business logic is encapsulated in Java beans or custom tags. Hence the use of JSP increases **reusability of code**, as well.
- JSP is a **text-based document** that can return dynamic content to client. It contains both “**static**” and “**dynamic content**”.



Introduction to JSP:

JSP versus Servlet:

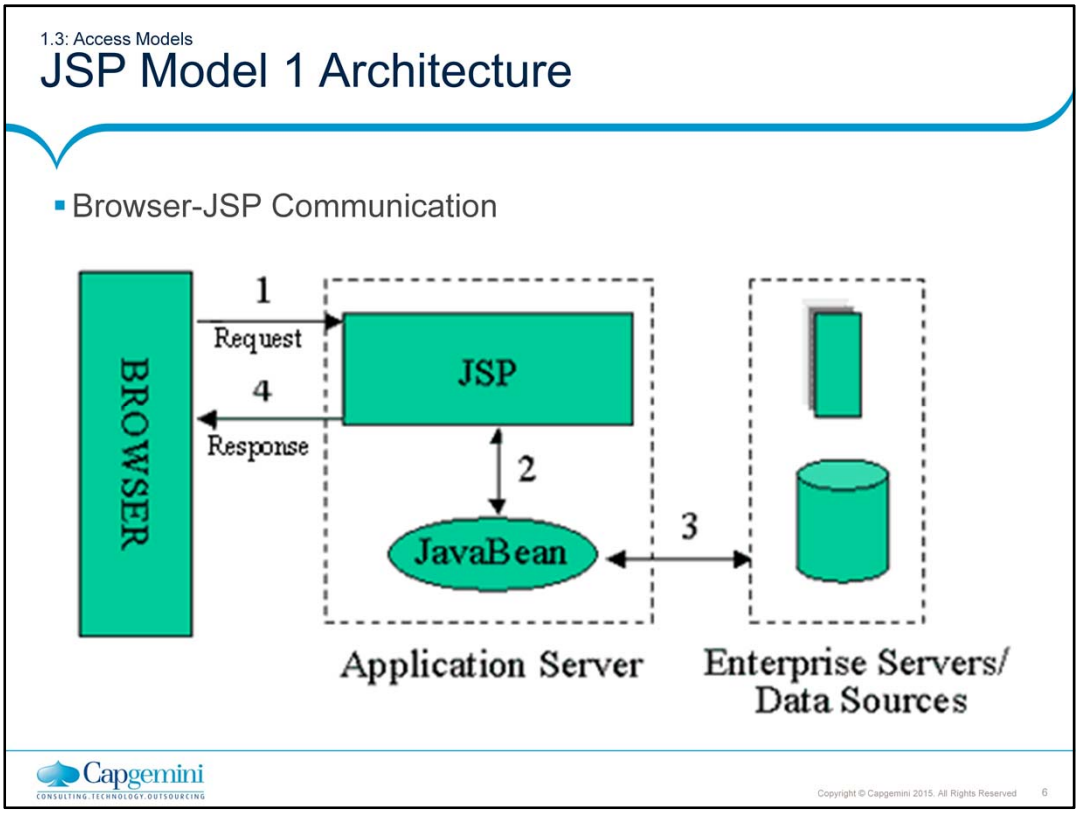
- **JSP** technology is built over **Servlet**. In fact, all JSP pages when requested, are first converted into Servlet. Thus they inherit all benefits of Servlets.
- Although dynamic, interactive pages with Servlets alone can be created using JSP technology makes the process even easier.
  - With JSP pages, it is easy to combine "static templates" with code to generate "dynamic content".
  - JSP page structure makes it easier to author pages "by hand".
  - JSP provides easy XML-like tags to invoke Java beans, thus hiding business complexity from page authors.
- JSP allows to clearly separate **content** from **presentation.** This is unlike Servlets where business logic and presentation logic co-exist many times in a single program! By separating the look from the content, different people can work on different tasks. Web page design experts can build the HTML, leaving places for Servlet programmers to insert the dynamic content.



Features of JSP Pages:

JSP provides the following features:

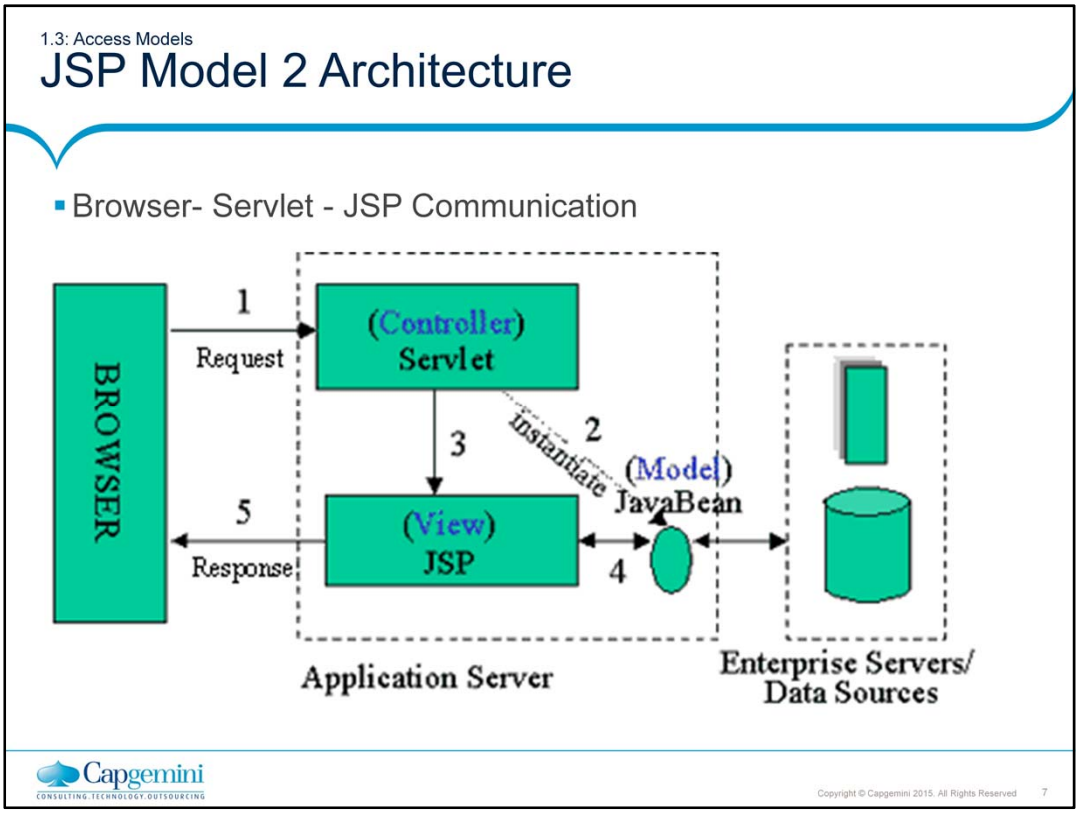
- **Portability:** JSPs files can be run on any web server or web-enabled application server that provides support for them. This support involves recognition, translation, and management of the Java Server Page lifecycle and its interactions with associated components. As long as the server, on which the Java Server Pages executes, supports the same specification level as that to which the file was written, no changes should be necessary as files are moved from server to server. However, instructions for the setup and configuration of the files may differ between files.
- **Composition:** The Java Server Pages architecture includes reusable Java components. The architecture also allows embedding a scripting language (default language for now is Java) directly into the Java Server Pages file. The components currently supported include JavaBeans and Servlets. Support for Enterprise Java Beans components are likely be added in a future release.
- **Processing:** A JSP file is essentially an HTML document with JSP scripting or tags. The Java Server Pages file has a *.jsp* extension. Before the page is served, the JSP syntax is parsed and processed into a servlet on the server side. The servlet that is generated outputs real content in straight HTML for responding to the client. Since it is standard HTML, the dynamically generated response looks no different to the client browser than a static response.



Access Methods:

JSP Model 1 Architecture:

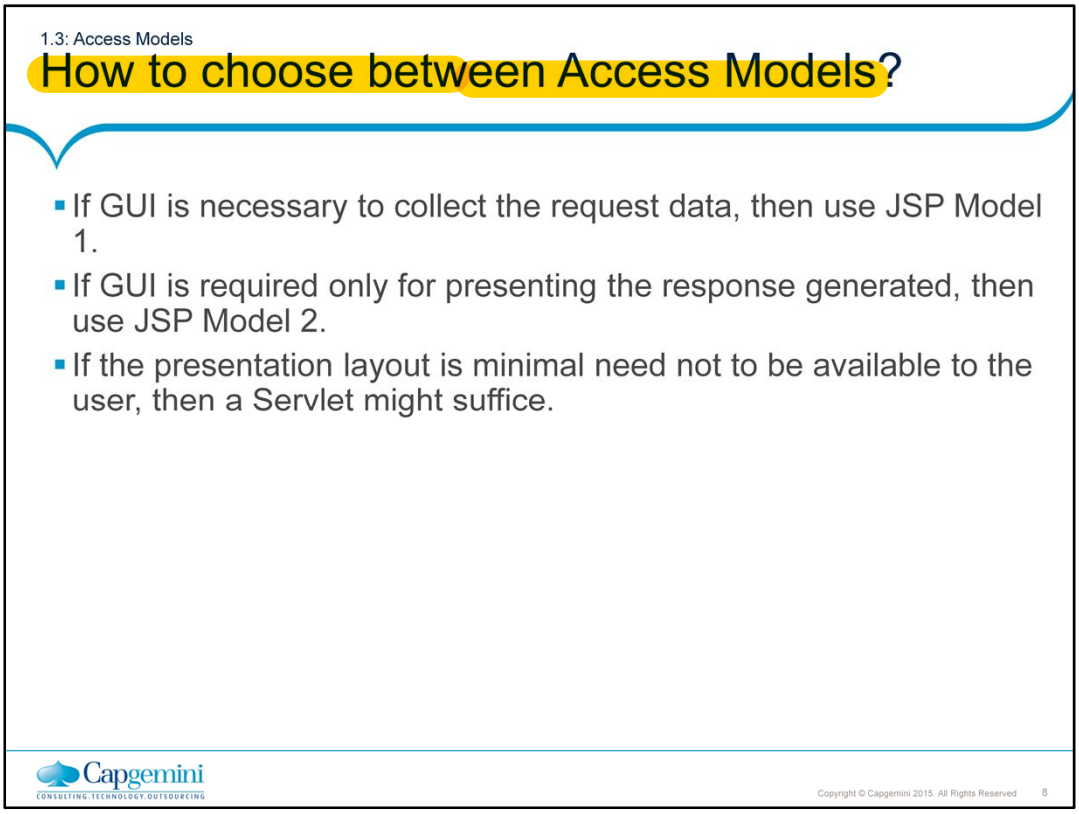
- In this model, the client request comes directly to a Java Server Page.
- Suppose the page accesses reusable JavaBean components (for example: JavaBeans) that perform particular well-defined computations like accessing a database. Then the result of the Bean's computations, called **result sets** are stored within the Bean as properties. The page uses such beans to generate dynamic content and present it back to the client.



Access Models:

JSP Model 2 Architecture:

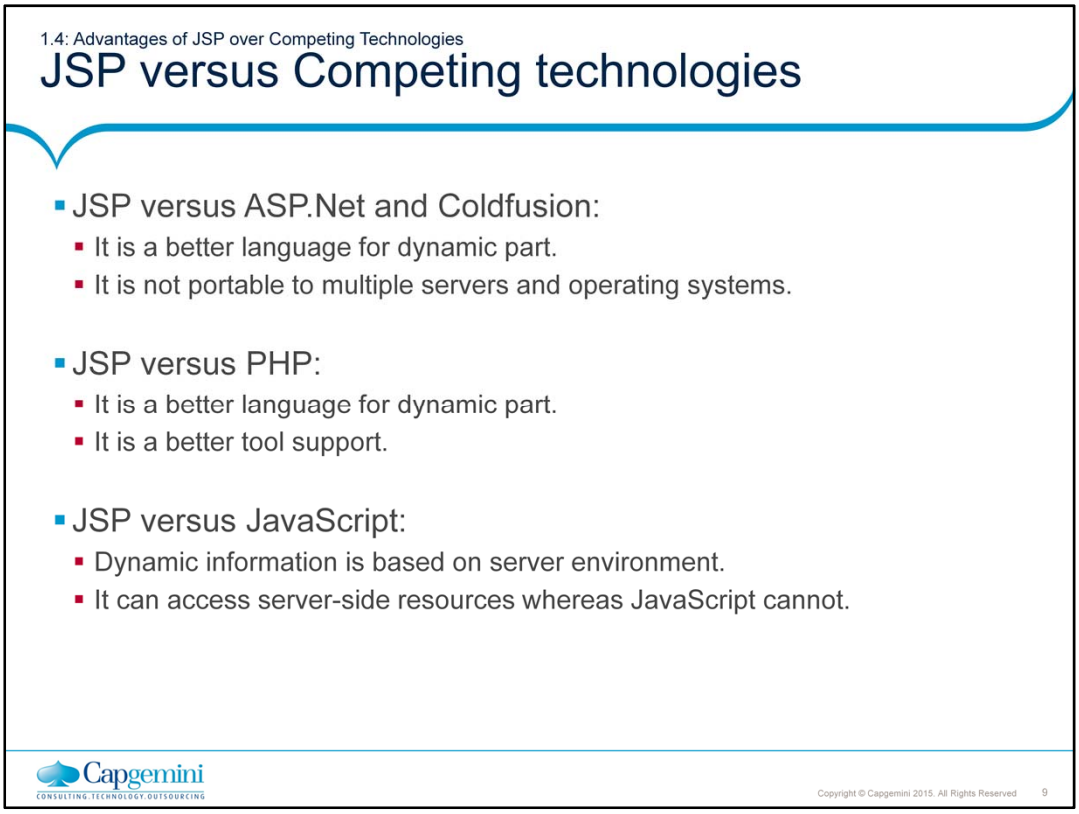
- In this model, the request comes through a servlet. The servlet generates the dynamic content. To handle the response to the client, the servlet creates a Bean and stores the dynamic content (sometimes called the **result set**) in the Bean. The servlet then invokes a Java Server Page that will present the content generated by the Bean.
- In this model, the servlet acts as a ***controller*** responsible for processing requests and creating any beans needed by the JSP page. The controller is also responsible for deciding to which JSP page to forward the request. The JSP page retrieves objects created by the servlet and extracts dynamic content for insertion within a template. This model promotes the use of the **Model View Controller (MVC)** architectural pattern.
- There are two APIs to support this model of request processing using Java Server Pages.
  - One API facilitates passing context between the invoking servlet and the Java Server Page.
  - The other API lets the invoking servlet specify which Java Server Page to use.
- In both the above cases, the page can contain any valid Java code, as well. The Java Server Pages architecture encourages separation of content from presentation, however it does not mandate it.



Access Models:

How to choose between Access Models?

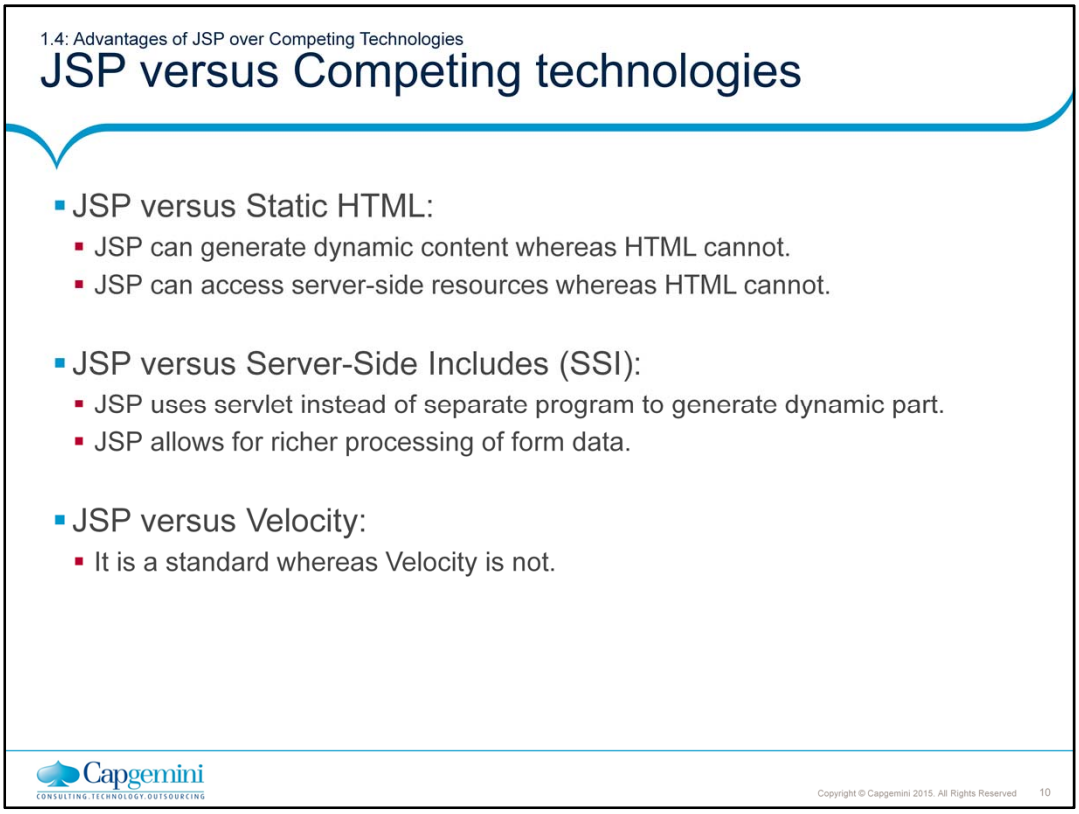
- With at least two access models, the question naturally arises "When does it make sense to have a Java Server Page as the front-end to a servlet, as the back-end, or use only the servlet?" Here are some possible guidelines:
  - If a graphical interface (GUI) is necessary to collect the request data, then use a Java Server Pages file.
  - If the request and request parameters are otherwise available to the servlet, but the results of the servlet processing requires a graphical interface to present them, then use a Java Server Pages file.
  - If presentation layout is minimal (will not require many println lines in servlet code) and does not require to make that presentation logic available to a customer or webpage designer, then a Servlet might sufficient.



Advantages of JSP over Competing Technologies:

JSP versus Competing technologies:

1. **JSP versus ASP.Net and Coldfusion:** ASP.Net is a Microsoft technology. Hence the dynamic part is written in Microsoft-specific language whereas in JSP it is written in Java, so it is powerful and promotes portability. Coldfusion also uses a proprietary language hence not portable. Secondly, JSP is portable to multiple servers and operating systems whereas ASP.Net and Coldfusion are not.
2. **JSP versus PHP:** The PHP is a scripting language which again not portable across servers. Unlike for JSP, the tool support for PHP is not very great.
3. **JSP versus JavaScript:** JavaScript can generate dynamic html on the client. However, the dynamic information is based on the client's environment. With the exception of cookies, form submission data is not available to JavaScript. Additionally, since JavaScript runs on client, it cannot access server-side resources.



Advantages of JSP over Competing Technologies:

JSP versus Competing technologies:

4. **JSP versus Static HTML:** HTML cannot generate dynamic information. Also similar to Javascript it cannot access server-side resources.
5. **JSP versus Server-Side Includes (SSI):** SSI is a widely supported technology for including externally defined pieces into a static web page. JSP is better because it allows to use Servlets instead of a separate program to generate that dynamic part. Besides, SSI is intended for only simple inclusions, and not for “real” programs that use form data, make database connections, and so on.
6. **JSP versus Velocity:** JSP allows separation of application logic from presentation but it does not mandate it. Due to this, it is very easy for the developers to mix up these two. Whereas, Velocity enforces the Model2 architecture by enforcing separation between the application logic and presentation layout logic. However, it is not a standard.

Summary

- In this lesson, you have learnt:
  - Introduction to Java Server Pages (JSP)
  - Features of JSP 2.2
  - Access Models
  - Advantages of JSP over competing technologies

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 11

Review – Questions

- Question 1: JSP architecture enables the separation of content generation from content ___.
- Question 2: The servlet resulting from the JSP uses the ___ method to process the client request.
- Question 3: In JSP Model 1 architecture, the client request is processed by ___.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 12

Review – Questions

- Question 4: JSP Model 2 enforces ___ architectural pattern.
- Question 5: The advantage of JSP over ASP.Net is ___ and ___.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 13