ServletConfig
ServletContext

Servlet Config Interface

- An object of ServletConfig is created by the web container for each servlet. This object can be used to get configuration information from web.xml file.
- If the configuration information is modified from the web.xml file, we don't need to change the servlet. So it is easier to manage the web application if any specific content is modified from time to time.

Advantage of ServletConfig

- The core advantage of ServletConfig is that you don't need to edit the servlet file if information is modified from the web.xml file.

Methods of ServletConfig interface

- **public String getInitParameter(String name):**Returns the parameter value for the specified parameter name.
- **public Enumeration getInitParameterNames():**Returns an enumeration of all the initialization parameter names.
- **public String getServletName():**Returns the name of the servlet.
- **public ServletContext getServletContext():**Returns an object of ServletContext.

Get the object of ServletConfig

- **getServletConfig() method** of Servlet interface returns the object of ServletConfig.

```
public   ServletConfig   getServletConfig();
```

- Example

```
ServletConfig config=getServletConfig();
//Now we can call the methods of ServletConfig interface
```

Syntax to provide the initialization parameter for a servlet

- The init-param sub-element of servlet is used to specify the initialization parameter for a servlet.

```
<web-app>
        <servlet>
                    ......

        <init-param>
                    <param-name>parametername</param-name>
                    <param-value>parametervalue</param-value>
        </init-param>
                     ......
        </servlet>
</web-app>
```

Example of ServletConfig to get initialization parameter

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class DemoServlet extends HttpServlet {
public void doGet(HttpServletRequest request, HttpServletResponse response)
   throws ServletException, IOException {

   response.setContentType("text/html");
   PrintWriter out = response.getWriter();

   ServletConfig config=getServletConfig();
   String driver=config.getInitParameter("driver");
   out.print("Driver is: "+driver);

   out.close();
   }

}
```

web.xml:

```
<web-app>

     <servlet>
          <servlet-name>DemoServlet</servlet-name>
          <servlet-class>DemoServlet</servlet-class>

            <init-param>
                        <param-name>driver</param-name>
                        <param-value>sun.jdbc.odbc.JdbcOdbcDriver</param-value>
            </init-param>

            </servlet>

    <servlet-mapping>
            <servlet-name>DemoServlet</servlet-name>
            <url-pattern>/servlet1</url-pattern>
    </servlet-mapping>

</web-app>
```

Example of ServletConfig to get all the initialization parameters

Getting all the initialization parameter from the web.xml file and printing this information in the servlet.

```
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Enumeration;

import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


public class DemoServlet extends HttpServlet {
public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {

  response.setContentType("text/html");
  PrintWriter out = response.getWriter();

  ServletConfig config=getServletConfig();
  Enumeration<String> e=config.getInitParameterNames();

  String str="";
  while(e.hasMoreElements()){
  str=e.nextElement();
  out.print("<br>Name: "+str);
  out.print(" value: "+config.getInitParameter(str));
  }

  out.close();
}

}
```

Web.xml:

```
<web-app>

    <servlet>
                <servlet-name>DemoServlet</servlet-name>
                <servlet-class>DemoServlet</servlet-class>

                <init-param>
                            <param-name>username</param-name>
                            <param-value>system</param-value>
                </init-param>

                <init-param>
                            <param-name>password</param-name>
                            <param-value>oracle</param-value>
                </init-param>

    </servlet>

    <servlet-mapping>
                <servlet-name>DemoServlet</servlet-name>
                <url-pattern>/servlet1</url-pattern>
    </servlet-mapping>

</web-app>
```

ServletContext Interface

- An object of ServletContext is created by the web container at time of deploying the project. This object can be used to get configuration information from web.xml file. There is only one ServletContext object per web application.
- If any information is shared to many servlet, it is better to provide it from the web.xml file using the **<context-param>** element.

Advantage of ServletContext

- **Easy to maintain** if any information is shared to all the servlet, it is better to make it available for all the servlet. We provide this information from the web.xml file, so if the information is changed, we don't need to modify the servlet. Thus it removes maintenance problem.

Usage of ServletContext Interface

There can be a lot of usage of ServletContext object. Some of them are as follows:

1. The object of ServletContext provides an interface between the container and servlet.
2. The ServletContext object can be used to get configuration information from the web.xml file.
3. The ServletContext object can be used to set, get or remove attribute from the web.xml file.
4. The ServletContext object can be used to provide inter-application communication.

Commonly used methods of ServletContext interface

There is given some commonly used methods of ServletContext interface.

1. **public String getInitParameter(String name):**Returns the parameter value for the specified parameter name.
2. **public Enumeration getInitParameterNames():**Returns the names of the context's initialization parameters.
3. **public void setAttribute(String name,Object object):**sets the given object in the application scope.
4. **public Object getAttribute(String name):**Returns the attribute for the specified name.
5. **public Enumeration getInitParameterNames():**Returns the names of the context's initialization parameters as an Enumeration of String objects.
6. **public void removeAttribute(String name):**Removes the attribute with the given name from the servlet context.

Get the object of ServletContext interface

1. **getServletContext() method** of ServletConfig interface returns the object of ServletContext.
2. **getServletContext() method** of GenericServlet class returns the object of ServletContext.

Syntax

public ServletContext getServletContext()

Example:

//We can get the ServletContext object from ServletConfig object

ServletContext application=getServletConfig().getServletContext();

//Another convenient way to get the ServletContext object

ServletContext application=getServletContext();

Context Initialization Parameter

- Context Initialization parameters are the parameter name and value pairs that you can specify in the ***deployment descriptor file (the web.xml file)***. Here we can specify the parameters that will be accessible to all the servlets in the web application.
- When we deploy the Web application, the Servlet container reads the initialization parameter from the **web.xml** file and initializes the **ServletContext** object with it. We can use the ***getInitParameter()*** and ***getInitParameterNames()*** methods of the **ServletContext** interface to get the parameter value and enumeration of parameter names respectively.
- For example, The parameter ***email_id*** with the value, since this is common to all the servlets, we can get the parameter name and value in any servlet.

```
<context-param>
    <param-name>email_id</param-name>
    <param-value>beginnersbook@gmail.com</param-value>
</context-param>
```

Example of ServletContext interface

Two java servlet classes, Both classes contains different information of students of same college. So instead of providing college name every time we are sharing this information from web.xml file.

DemoServletContext.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class DemoServletContext extends HttpServlet
{
public void doGet(HttpServletRequest req,HttpServletResponse res) throws
    ServletException,IOException
{
        res.setContentType("text/html");
        PrintWriter pw=res.getWriter();
        ServletContext con=getServletContext();
        String st=con.getInitParameter("College");
        pw.println("<html><body>");
        pw.println("<h1>Name:Anuj</h1>");
        pw.println("<h1>Roll number:101</h1>");
        pw.println("<h1>College:"+st+"</h1>");
        pw.println("</body></html>");
  }
}
```

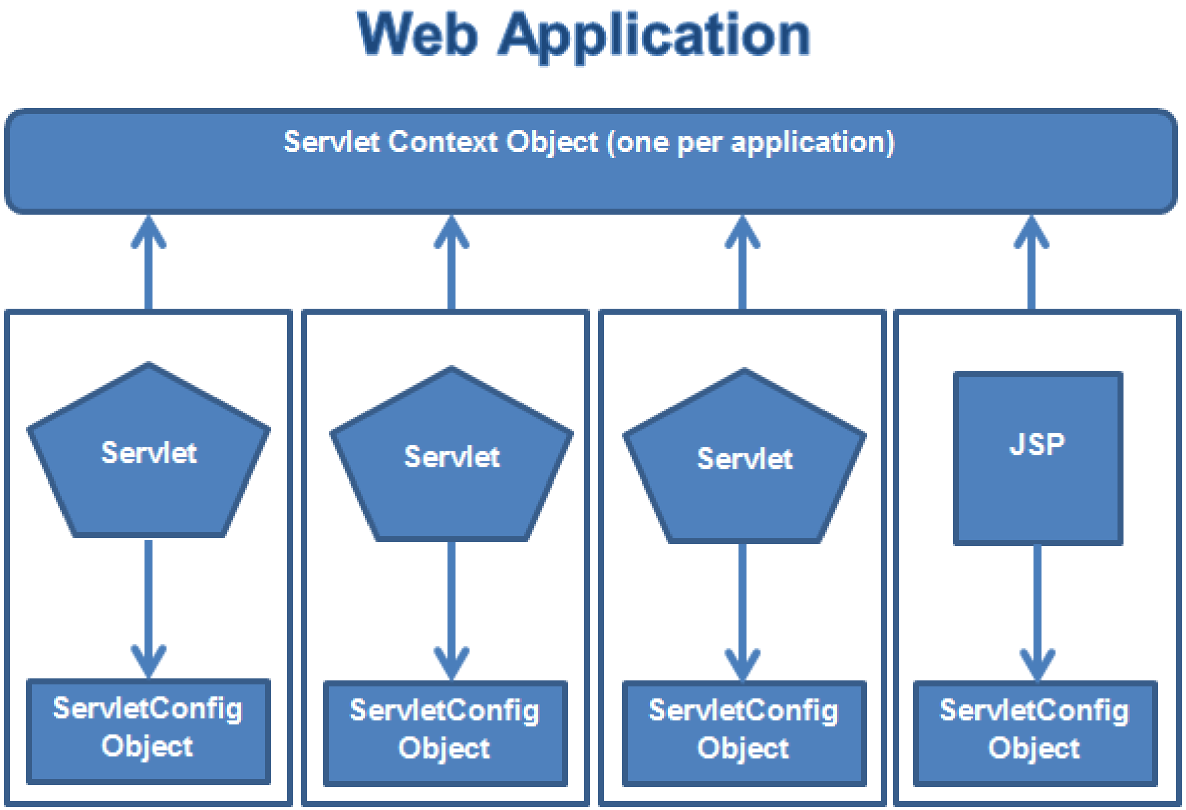
DemoServletContext1.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class DemoServletContext1 extends HttpServlet
{
public void doGet(HttpServletRequest req,HttpServletResponse res) throws
   ServletException,IOException
{
      res.setContentType("text/html");
      PrintWriter pw=res.getWriter();
      ServletContext con=getServletContext();
      String st=con.getInitParameter("College");
      pw.println("<html><body>");
      pw.println("<h1>Name:Abhishek</h1>");
      pw.println("<h1>Roll number:102</h1>");
      pw.println("<h1>College:"+st+"</h1>");
      pw.println("</body></html>");
}
}
```

Web.xml

```
<servlet>
     <servlet-name>DemoServletContext</servlet-name>
     <servlet-class>DemoServletContext</servlet-class>
  </servlet>
   <servlet>
     <servlet-name>DemoServletContext1</servlet-name>
      <servlet-class>DemoServletContext1</servlet-class>
  </servlet>
  <context-param>
     <param-name>College</param-name>
     <param-value>Vasavi College of Engineering</param-value>
  </context-param>
  <servlet-mapping>
     <servlet-name>DemoServletContext1</servlet-name>
     <url-pattern>/Demo1</url-pattern>
  </servlet-mapping>
   <servlet-mapping>
     <servlet-name>DemoServletContext</servlet-name>
     <url-pattern>/Demo</url-pattern>
  </servlet-mapping>
```

Servlet Context and Servlet Config



Web Application

Servlet Context Object (one per application)

Servlet

Servlet

Servlet

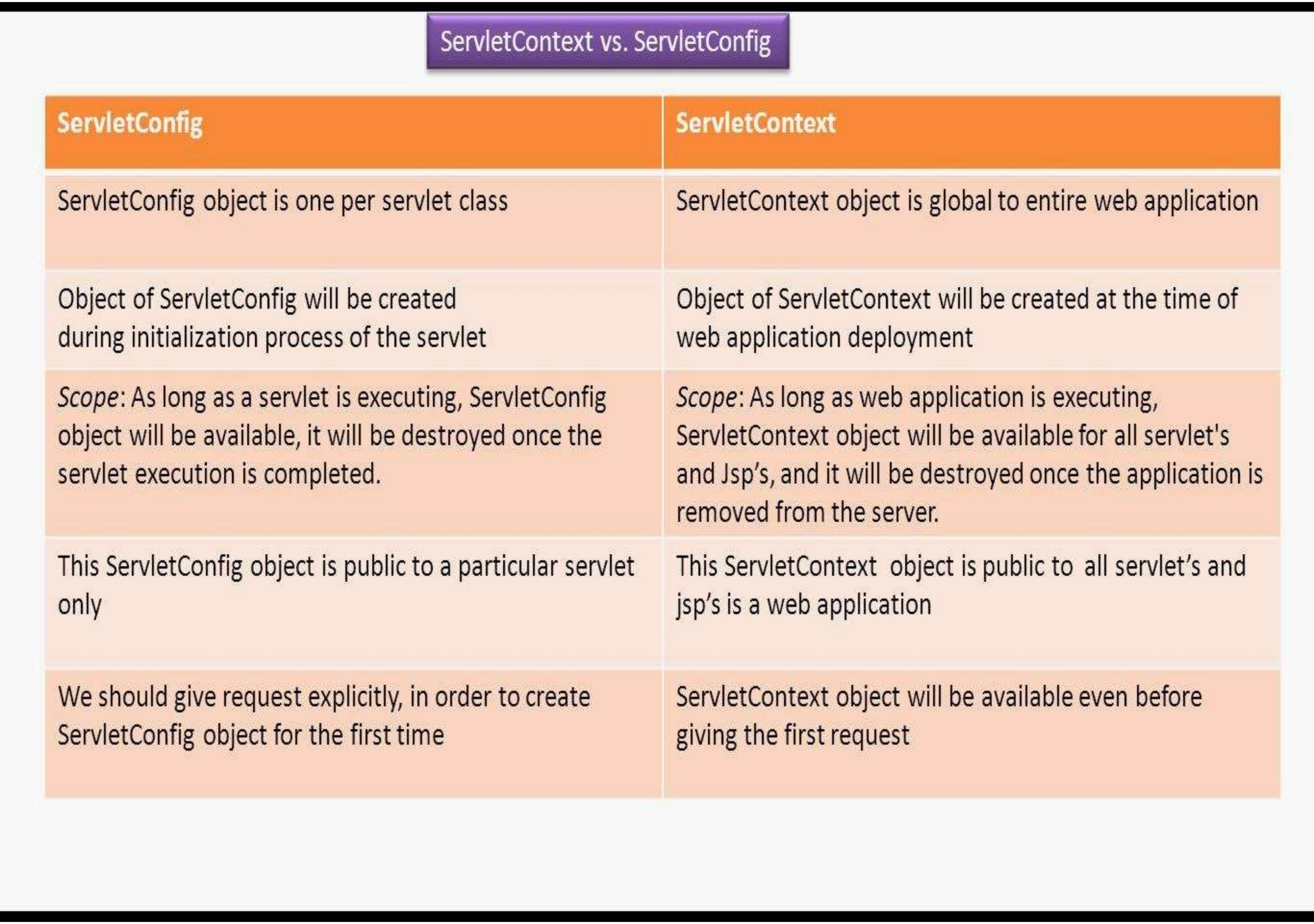
JSP

ServletConfig Object

ServletConfig Object

ServletConfig Object

ServletConfig Object



ServletContext vs. ServletConfig

ServletConfig	ServletContext
ServletConfig object is one per servlet class	ServletContext object is global to entire web application
Object of ServletConfig will be created during initialization process of the servlet	Object of ServletContext will be created at the time of web application deployment
Scope: As long as a servlet is executing, ServletConfig object will be available, it will be destroyed once the servlet execution is completed.	*Scope*: As long as web application is executing, ServletContext object will be available for all servlet's and Jsp's, and it will be destroyed once the application is removed from the server.
This ServletConfig object is public to a particular servlet only	This ServletContext object is public to all servlet's and jsp's is a web application
We should give request explicitly, in order to create ServletConfig object for the first time	ServletContext object will be available even before giving the first request

Attribute in Servlet

- An **attribute in servlet** is an object that can be set, get or removed from one of the following scopes:
  - request scope
  - session scope
  - application scope
- The servlet programmer can pass informations from one servlet to another using attributes. It is just like passing object from one class to another so that we can reuse the same object again and again.

Attribute specific methods of ServletRequest, HttpSession and ServletContext interface:

There are following 4 attribute specific methods. They are as follows:

- **public void setAttribute(String name,Object object):**sets the given object in the application scope.
- **public Object getAttribute(String name):**Returns the attribute for the specified name.
- **public Enumeration getInitParameterNames():**Returns the names of the context's initialization parameters as an Enumeration of String objects.
- **public void removeAttribute(String name):**Removes the attribute with the given name from the servlet context.

Example of ServletContext to set and get attribute

In this example, we are setting the attribute in the application scope and getting that value from another servlet.

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class DemoServlet1 extends HttpServlet{
public void doGet(HttpServletRequest req,HttpServletResponse res)
{
try{

res.setContentType("text/html");
PrintWriter out=res.getWriter();

ServletContext context=getServletContext();
context.setAttribute("company","IBM");

out.println("Welcome to first servlet");
out.println("<a href='servlet2'>visit</a>");
out.close();

}catch(Exception e){out.println(e);}

}}
```

DemoServlet2.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class DemoServlet2 extends HttpServlet{
public void doGet(HttpServletRequest req,HttpServletResponse res)
{
try{

res.setContentType("text/html");
PrintWriter out=res.getWriter();

ServletContext con=getServletContext();
String n=(String)con.getAttribute("company");

out.println("Welcome to "+n);
out.close();

}catch(Exception e){out.println(e);}
}}
```

web.xml

```
<web-app>

	<servlet>
		<servlet-name>s1</servlet-name>
		<servlet-class>DemoServlet1</servlet-class>
	</servlet>

	<servlet-mapping>
		<servlet-name>s1</servlet-name>
		<url-pattern>/servlet1</url-pattern>
	</servlet-mapping>

	<servlet>
		<servlet-name>s2</servlet-name>
		<servlet-class>DemoServlet2</servlet-class>
	</servlet>

	<servlet-mapping>
		<servlet-name>s2</servlet-name>
		<url-pattern>/servlet2</url-pattern>
	</servlet-mapping>

</web-app>
```