**JavaFull Stack Training**

**JavaScript Day 1 –> 07-10-2019**

**ES5 Features :**

**ECMAScript : European Computer Manifacture Association**

**ECMAScript is concept.**

**JavaScript is a one of the implementation of ECMAScript**

**JavaScript was object based interpreter scripting**

**language.**

**Object based Vs Object oriented**

**Interpreter Vs compiler**

**Scripting Vs programming**

**JavaScript is case sensitive scripting language.**

**JavaScript mainly use to do client side validation as**

**well as help to create the dynamic web page.**

**Syntax to write simple JavaScript program**

**<script type=”text/JavaScript” Language=”JavaScript”>**

**</script>**

**Data Types :**

**In JavaScript Data types are divided into four types**

1. **Number data types**
2. **String data types**
3. **Boolean types**
4. **Object reference**

**In JavaScript up es5 to declare the variable we will use var keyword with variableName.**

**Syntax**

**var varaiableName;**

**var name;**

**var a=10;**

**var b =10.10;**

**var name1=”Ravi”;**

**var res = true;**

**var obj = new Date();**

var fname;

    var a=10;

    var b =10.10;

    var name1="Ravi";

    var res = true;

    var obj = new **Date**();

    document.**write**("Welcome to JavaScript code<br/>")

    document.**write**("Welcome to JavaScript code<br/>")

    document.**write**("<font color='red'>Welcome to JavaScript code</font><br/>")

    document.**write**("VAlue is "+fname);

    document.**write**("<br/>VAlue is "+a);

    document.**write**("<br/>VAlue is "+b);

    document.**write**("<br/>VAlue is "+name1);

    document.**write**("<br/>VAlue is "+res);

    document.**write**("<br/>VAlue is "+obj);

**Operators :**

**Arithmetic Operator**

**Logical Operator**

**Conditional Operator**

**Increment and decrement**

**Ternary operator**

**typeof();**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <h2>Welcome to HTML Page</h2>

    <script *type*="text/JavaScript">

    var a;

    document.**write**("Value of a "+a);

    document.**write**("<br/>Value of a "+typeof(a));

    a=10;

    document.**write**("<br/>Value of a "+a);

    document.**write**("<br/>Value of a "+typeof(a));

    a=10.10;

    document.**write**("<br/>Value of a "+a);

    document.**write**("<br/>Value of a "+typeof(a));

    a="Ravi";

    document.**write**("<br/>Value of a "+a);

    document.**write**("<br/>Value of a "+typeof(a));

    a=true;

    document.**write**("<br/>Value of a "+a);

    document.**write**("<br/>Value of a "+typeof(a));

    a=new **Date**();

    document.**write**("<br/>Value of a "+a);

    document.**write**("<br/>Value of a "+typeof(a));

    </script>

</body>

</html>

**=, ==, ===**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <h2>Welcome to HTML Page</h2>

    <script *type*="text/JavaScript">

    var a =10;

    var b ="10";

    document.**write**("== <br/>")

    document.**write**(a==b);

    document.**write**("=== <br/>")

    document.**write**(a===b);

    </script>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <h2>Welcome to HTML Page</h2>

    <script *type*="text/JavaScript">

    var a =10;

    var b ="10";

    document.**write**("== <br/>")

    document.**write**(a==b);

    document.**write**("=== <br/>")

    document.**write**(a===b);

    </script>

</body>

</html>

**If statements**

**Switch statements**

**Looping**

**While loop**

**Do while loop**

**For loop**

**Functions :**

**In JavaScript function are divided into two types**

1. **Pre-defined functions**
2. **User-defined functions.**

**Few pre-defined function**

1. **alert(“Msg”); : it is use to display the pop up message**
2. **prompt(“Msg”);: it us use to receive the value through keyboards.**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <h2>Welcome to HTML Page</h2>

    <script *type*="text/JavaScript">

    var fname = **prompt**("Enter the name");

**alert**("Welcome to JavaScript "+fname);

**alert**("Hi");

    </script>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <h2>Welcome to HTML Page</h2>

    <script *type*="text/JavaScript">

    var a = **prompt**("Enter the value of a");

    var b = **prompt**("Enter the value of b ");

    var sum1 = a+b;

**alert**("Sum "+sum1);

    var sum2 = **parseInt**(a)+**parseInt**(b); *//string to int*

**alert**("Sum "+sum2);

    var sum3 = **parseFloat**(a)+**parseFloat**(b); *//string to float*

**alert**("Sum "+sum3);

    var sum4 = **eval**(a)+**eval**(b); *//string to number*

**alert**("Sum "+sum4);

    </script>

</body>

</html>

1. **confirm():**

**do {**

**alert(“1:Add/2:Sub”)**

**prompt()**

**switch**

**case 1: Addition**

**case 2 : Substration**

**}while()**

**User-defined functions:**

**1st ways**

**Syntax**

**function functionName(parameterList) {**

**}**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

    <script *type*="text/JavaScript">

    function **abc**() {

**alert**("Welcome to User-defined function")

    }

    function **addNumber**(a,b){

        var sum = a+b;

**alert**("Sum of two number is "+sum)

    }

    function **empInfo**(id,name,salary){

        return "Welcome to JavaScript "+name;

    }

    </script>

</head>

<body>

    <h2>Welcome to HTML Page</h2>

    <script *type*="text/JavaScript">

**alert**("Main script tag")

**abc**();

**addNumber**(100,200);

    var result = **empInfo**(100,'Ravi',15000);

**alert**(result);

    </script>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

    <script *type*="text/JavaScript">

    function **addNumber**() {

*//alert("fun called..")*

        var leng = addNumber.arguments.length;

**alert**("Number of arguments are "+leng);

        var sum = 0;

        for(i=0;i<leng;i++){

            sum = sum+addNumber.arguments[i];

        }

**alert**("Sum of arguments are "+sum);

    }

    </script>

</head>

<body>

    <h2>Welcome to HTML Page</h2>

    <script *type*="text/JavaScript">

**addNumber**();

**addNumber**(10);

**addNumber**(10,20);

**addNumber**(10,20,30);

    </script>

</body>

</html>

**Events : Event is a interaction between user and**

**HTML components like textfield, radiobutton, button,**

**Keyboard, mouse, etc.**

**Types of events**

**onClick**

**onDblClick**

**onMouseOver**

**onMouseOut**

**onKeyUp**

**onKeyDown**

**onBlur**

**onFocus**

**onSubmit**

**onLoad**

**onUnLoad**

**onChange**

**1 way**

**document.formName.componentName.value**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

    <script *type*="text/JavaScript">

    function **abc**() {

    var name = document.f1.user.value;

**alert**("Welcome "+name);

    }

    </script>

</head>

<body>

    <h2>Welcome to HTML Page</h2>

    <form *name*="f1">

    <label>UserName</label>

    <input *type*="text" *name*="user"/>

    <input *type*="button" *value*="Click Here" *onclick*="**abc**()"/>

    </form>

</body>

</html>

**DOM : Document Object Model**

**document.getElementById(“idName”).value;**

**document.getElementById(“idName”).**

**innerHTML=”Msg”**

**1 2 3**

**4 5 6**

**7 8 0**

**jQuery**

**XML**

**Maven and Gradle**

**Servlet and JSP**

**JavaScript Day 2 –> 09-10-2019**

**JavaScript objects**

1. **pre-defined**
2. **user-defined**

**pre-defined objects.**

**BOM : Browser Object Model**

**DOM : Document Object Model**

**Object**

**Property : field**

**Behavior : functions**

**Object**

**Property**

**Behavior**

**Object**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    var obj = [10,20,30,40,50];*// array object using literal*

    window.document.**write**(obj+"<br>");

    document.**write**(obj[0]+"<br>")

    obj.**push**(100);

    window.document.**write**(obj+"<br>");

    window.document.**write**(obj.**pop**()+"<br>");

    window.document.**write**(obj.**join**("--->")+"<br>");

    for(i=0;i<obj.length;i++){

        document.**write**("<br> "+obj[i]);

    }

    var obj1 = new **Array**(10);

    document.**write**("<br/>Size of array "+obj1.length);

    var obj2 = new **Array**(10,2);

    document.**write**("<br/>Size of array "+obj2.length);

    </script>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    var dd = new **Date**();

    document.**write**(dd+"<br>");

    document.**write**(dd.**getDate**()+"<br>");

    document.**write**(dd.**getMonth**()+"<br>");

    dd.**setMonth**(dd.**getMonth**()+1);

    document.**write**(dd.**getMonth**()+"<br>");

    document.**write**(dd.**getYear**()+"<br>");

    document.**write**(dd.**getFullYear**()+"<br>");

    document.**write**(dd.**getHours**()+"<br>");

    document.**write**(dd.**getMinutes**()+"<br>");

    document.**write**(dd.**getSeconds**()+"<br>");

    </script>

</body>

</html>

**String**

**Boolean**

**Number**

**Math**

**window objects property and behavior**

**Error**

**setInterval()**

**setTimeout()**

**clearInterval()**

**alert(“Hello”);**

**alert(“Hi”);**

**These function is known as callback asynchronouse functions.**

**Callback functions: Passing the function body or**

**function itself as a parameter to another functions is**

**known callback functions like a lambda expression**

**in a Java(ES6 Arrow functions).**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    function **abc**(data) {

*//  alert(data);*

**data**();

    }

    function **mno**() {

**alert**("Mno function called...");

    }

**abc**(mno);

    var obj = [10,20,30,40,50,60];

*/\* obj.forEach(function(v){*

*document.write("<br>"+v);*

*})\*/*

    obj.**forEach**(displayInfo);

    function **displayInfo**(data){

        document.**write**("<br/>"+data);

    }

    </script>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    function **abc**(data) {

*//  alert(data);*

**data**();

    }

    function **mno**() {

**alert**("Mno function called...");

    }

**abc**(mno);

    var obj = [10,20,30,40,50,60];

*/\* obj.forEach(function(v){*

*document.write("<br>"+v);*

*})\*/*

    obj.**forEach**(displayInfo);

    function **displayInfo**(v,i){

        document.**write**("<br/>"+v+"--"+i);

    }

    </script>

</body>

</html>

**Asynchronouse communication**

**L1**

**L2**

**L3**

**L4**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    var n=0;

    function **abc**() {

*//document.write("Hi");*

*//alert("Hi")*

        document.**getElementById**("ele").innerHTML=n;

        n++;

    }

*//setTimeout(abc,6000);*

**setInterval**(abc,500);

    document.**write**("Hello");

    </script>

    <div *id*="ele">

    </div>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    var n=0;

    function **abc**() {

            if(n%2==0){

*//  document.bgColor="red";*

          document.**getElementById**("ele").style.fontSize=20+"px";

            }else {

*//  document.bgColor="green";*

          document.**getElementById**("ele").style.fontSize=40+"px";

            }

            n++;

    }

*//setTimeout(abc,6000);*

    var obj = **setInterval**(abc,1000);

    document.**write**("Hello");

    function **stop**() {

**clearInterval**(obj);

    }

    </script>

    <input *type*="button" *value*="Stop" *onclick*="**stop**()"/>

    <div *id*="ele">

            Welcome to HTML JavaScript code

    </div>

</body>

</html>

**JSON : JavaScript Object Notation**

**Java(req)**

**Client SBI XML/JSON HDFC**

**.net (res)**

**JEE Asp.net**

**Syntax :**

**“key”:value;**

**{“key1”:value1,”key2”:”value2”}**

**OOPs in JavaScript Using ES5 (User-defined Objects)**

**object : any real world entity.**

**person, bank, animal etc.**

**function Person() {**

**}**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    function **Person**() {

*//alert("fun called...")*

        this.pid=100;

        this.pname = "Ravi";

        this.**displayPersonInfo** = function() {

**alert**("Id is "+this.pid+" Name is "+this.pname);

        }

    }

*//Person();*

    var p1 = new **Person**();

**alert**(p1.pid);

    p1.**displayPersonInfo**();

    </script>

</body>

</html>

**Parameterized constructor:**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    function **Person**(pid,pname) {

*//alert("fun called...")*

        this.pid=pid;

        this.pname = pname;

        this.**displayPersonInfo** = function() {

**alert**("Id is "+this.pid+" Name is "+this.pname);

        }

    }

*//Person();*

    var p1 = new **Person**(101,"Mahesh");

**alert**(p1.pid);

    p1.**displayPersonInfo**();

    </script>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    function **Person**(pid,pname) {

*//alert("fun called...")*

        this.pid=pid;

        this.pname = pname;

        this.**displayPersonInfo** = function() {

**alert**("Id is "+this.pid+" Name is "+this.pname);

        }

    }

*//Person();*

    var p1 = new **Person**(101,"Mahesh");

**Person**.prototype.age = 21;

**alert**("Id is "+p1.pid+" Name is "+p1.pname+"Age is "+p1.age);

**Person**.prototype.**setInfo**=function(pid,pname,age){

        this.pid = pid;

        this.pname = pname;

        this.age = age;

    }

    p1.**setInfo**(102,"Ravi",24);

**alert**("Id is "+p1.pid+" Name is "+p1.pname+"Age is "+p1.age);

    </script>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    function **Person**(pid,pname,salary) {

        this.pid=pid;

        this.pname = pname;     *//public property*

        var salary = salary;*//private property*

        this.**displayPersonInfo** = function() {   *//public function*

**alert**("Id is "+this.pid+" Name is "+this.pname);

**alert**("Salary is "+salary);

**displayInfo**();

        }

        function **displayInfo**() {*//private function*

**alert**("Private funtion....")

        }

    }

    var p1 = new **Person**(101,"Mahesh",14000);

**alert**(p1.pid);

**alert**(p1.pname);

**alert**(p1.salary);

    p1.**displayPersonInfo**();

*//displayInfo();*

    </script>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <meta *http-equiv*="X-UA-Compatible" *content*="ie=edge">

    <title>Document</title>

</head>

<body>

    <script *type*="text/JavaScript">

    function **Employee**(id,name,salary){

        this.id = id;

        this.name = name;

        this.salary = salary;

        this.**disEmp**= function(){

**console**.**log**("Id is "+this.id+" Name is "+this.name+" Salary "+this.salary);

        }

    }

    var emp1 = new **Employee**(100,"Ravi",12000);

    emp1.**disEmp**();

    function **Manager**(numberOfEmp){

            this.numberOfEmp = numberOfEmp;

            this.**disMgr** = function() {

**console**.**log**("Number of emp "+this.numberOfEmp);

            }

    }

    var mgr1 = new **Manager**(12);

    mgr1.**disMgr**();

**Manager**.prototype =  **Object**.**create**(emp1);

   var mgr2 = new **Manager**(14);

   mgr2.**disEmp**();

   mgr2.**disMgr**();

**Manager**.prototype =  **Object**.**create**(new **Employee**(101,"Ajay",16000));

   var mgr3 = new **Manager**(16);

   mgr3.**disEmp**();

   mgr3.**disMgr**();

    </script>

</body>

</html>

**jQuery and XML–> 10-10-2019**

**jQuery : jQuery is a external JavaScript library.**

**It contains lot of pre-defined function which internally**

**Connected to each other to perform task on DOM element**

**or tags like read, write and modify the dom easily.**

**Syntax**

**jQuery()**

**or**

**$()**

**Syntax**

**$(“selectors”).do(task);**

**var sel = $(“p”).click().css(“color”,”red”);**

**var obj = sel.css(“color”,”red”);**

**jQuery effect methods :**

**jQuery DOM Manipulation methods :**

**XML : eXtensible Markup Language:**

**XML is a platform independent, language independent**

**And application independent as well as**

**Browser independent markup up language.**

**HTML Vs XML**

1. **HTML is use to display the data where xml is use to describe the data or store the data or hold data or transfer the data.**
2. **HTML contains pre-defined tags where xml contains user-defined tags as well as pre-defined tags according to domain or server or tool specific.**
3. **HTML is a case insensitive. Where xml is a case sensitive.**
4. **In html tags are optional without html or any tags we can display the output. But in xml it require at least one tag that tag is known as root tag or document tag or parent tag**
5. **In html closing tag is optional. But in xml every tag must be close properly as well as it must be properly nested.**

**Like**

**<p><b><i>Welcome </p></i></b>**

**JAX\_B: Java API for XML Binding :**

**DTD: Document Type Definition**

**XSD: XML Schema Definition**

**Day 1 – Servlet / JSP –11/10/2019**

**Types of XML file**

1. **Well form xml file or non validate xml file**
2. **Validate xml file using dtd or xsd**

**Parser: parser is a plugin or software which help to parse the xml file.**

**Type of parser**

**Non validation parser**

**Validation parser**

**By default all browser contains non validation parser which help to check well form xml file.**

**DTD and XSD file**

**DTD Limitation**

1. **DTD is use to make the validate xml file or rules for xml file but dtd itself doesn’t follow xml rules.**
2. **DTD support only two type of data types pcdata and cdate (both character data).**
3. **Dtd doesn’t support min and max occurrence.**

**?(0 or 1), +(1 or many) , \*(o or many)**

1. **Dtd doesn’t support regular expression.**
2. **Dtd doesn’t support namespace like package in java.**

**<Emp xmlns:a=”c:\\abc” xmlns:b=”c:\\abc”>**

**<a:Name></a:Name>**

**<Name></Name>**

**<b:Name></b:Name>**

**</Emp>**

**XSD : Xml Schema definition**

**<Emp>**

**<Name LName=”Deep”>Raj</Name> DTD**

**<Name LName=”Deep” FName=”Raj”> </Name>**

**<Name>**

**<FName>Raj</FName>**

**<LName>Deep</LName>**

**</Name>**

**</Emp>**

**JEE: Java Enterprise Edition:**

[**https://www.google.com**](https://www.google.com) **-🡪 URL**

**req (http/https)---🡪**

**Client Server**

**🡨-res (http/https)------**

**HTML/HTML5**

**CSS/CSS3**

**Bootstrap**

**JavaScript**

**jQuery**

**Jee**

**Servlet /JSP and EJB**

**Java Server Pages**

**Enterprise Java Bean**

**Asp.net**

**Php**

**Cgi**

**Python**

**Node js**

**Server :**

**2 types of server (application point )**

1. **Web Server : Tomcat**
2. **Application Server : Web Logic, JBoss, WAS(WebSphere), Glashfish etc.**

**Container : It is a part of server also known as engine. It is responsible to execute Servlet, JSP and EJB application like loading class, create the object, call the life methods and destroy the objects etc.**

**2 types of container.**

1. **Web Container**
2. **EJB Container**

**If server is a web server like tomcate which contains only one type of container ie web container. Web container is responsible to execute servler and jsp.**

**If server is application server like jboss, it contains different type of container like web container and ejb container. Where web container responsible to execute servlet and jsp and ejb container is use to execute ejb application.**

**Application server provide lot of extra features like connection pooling, thread management, resource management, security etc.**

**Servlet : Servlet is a normal java program which help to create dynamic web page on server side.**

**javax.servlet.\*;**

**servlet: servlet is a package which contains set of classes and interfaces which help to develop the web page.**

**javax.servlet.Servlet;**

**Servlet :Servlet is a interface part of servlet package which contains set of methods.**

**5 methods**

**init**

**service**

**destroy life cycle methods.**

**getServletInfo**

**getServletConfig**

**class Demo implements Servlet {**

**all five methods mandatory has to override**

**}**

**GenericServlet : It is type of abstract class which internally implements Servlet interface and provided body for four methods except service() methods.**

**class Demo extends GenericServlet {**

**only one methods has override mandatory**

**ie service()**

**}**

**HttpServlet : It is a type of abstract class which internally extends GenericServlet class and provide body for service methods. This class provide some extra methods in the form of doXXX()**

**Like**

**doGet(), doPost(), doPut(), doDelete() etc**

**import javax.servlet.http.\*;**

**import javax.servlet.\*;**

**class Demo extends HttpServlet {**

**service() or doGet() or doPost()**

**}**

**class Employee {**

**id**

**name**

**salary**

**}**

**Employee emp = new Employee();**

**emp.id=100;**

**emp.name=”Ravi”;**

**emp.salary=14000;**

**emp**

**json or xml**

**{“id”:100,”name”:”Ravi”,”salary”:12000}**

**<Emp>**

**<Id>1</Id>**

**<Name>Ravi</Name>**

**<Salary>12000</Salary>**

**</Emp>**

**package com;**

**import javax.servlet.\*;**

**import javax.servlet.http.\*;**

**public class Demo extends HttpServlet {**

**public void doGet(HttpServletRequest req, HttpServletResponse res) {**

**System.out.println(“Welcome”);**

**PrintWriter pw = res.getWriter();**

**pw.println(“Welcome to Servlet Program”);**

**}**

**}**

**Configuration details of servlet using xml or annotation**

**dd file (deployment descriptor file)**

**web.xml**

**<web-app>**

**<servlet>**

**<servlet-name>A</servlet-name>**

**<servlet-class>com.Demo</servlet-class>**

**</servlet>**

**<servlet-mapping>**

**<servlet-name>A</servlet-name>**

**<url-pattern>/hi</url-pattern>**

**</servlet-mapping>**

**</web-app>**

[**http://localhost:8080/DemoWebApp/hi---**](http://localhost:8080/DemoWebApp/hi---)**> URL**

**RequestDispatcher : It is a interface which provided set of methods which help to navigate from servlet to jsp/html/servlet page.**

**Syntax**

**RequestDispatcher rd = request.getRequestDispatcher(“path”);**

**Path:**

**Target page is Servlet 🡪 then path must be URL pattern of that servlet page.**

**Target page is html/jsp 🡪 then path must be pageName.html/pageName.jsp.**

**rd.forward(request,response);**

**Output : only target page**

**rd.include(request,response);**

**Output : Source + Target page.**

**Servlet Life cycle :**

1. **When first client send the request to Servlet using URL pattern. Container load the class and after loaded successfully it will create the object of the Servlet class.**
2. **After object created successfully it will call init() method**

**Syntax**

**public void init(SerlverConfig con) {**

**}**

**Init method contains ServletConig interface reference as a parameter. This reference is use to receive the value or resources from web.xml file. This method call only once at the starting time.**

1. **After init method Container call service method.**

**Syntax**

**public void service(HttpServletRequest req, HttpServletResponse res) {**

**}**

**req is use to receive the request from a client and res is use to give the response back to client.**

**This method will call again and again if client send same request to application.**

1. **At last destroy method get call by container to destroy the Servlet object.**

**public void destroy(){**

**}**

**ServletCofig and ServletContext :**

**Both are interface which is use to receive value or resources from a web.xml file.**

**ServletContext sc;**

**String admin, user;**

**public void init(SevletConfig con) {**

**sc = con.getServletContext();**

**user = con.getInitParameter(“n1”);**

**admin = sc.getInitParamter(“n2”);**

**}**

**web.xml**

**<web-app>**

**<context-param>**

**<param-name>driver</param-name>**

**<param-value>com.myql.jdbc.Driver</param-value>**

**</context-param>**

**<servlet>**

**<servlet-name>A</servlet-name>**

**<servlet-class>com.Demo</servlet-class>**

**<init-param>**

**<param-name>qry</param-name>**

**<param-value>insert into login value(?,?)</param-value>**

**</init-param>**

**</servlet>**

**<servlet-mapping>**

**<servlet-name>A</servlet-name>**

**<url-pattern>/hi</url-pattern>**

**</servlet-mapping>**

**</web-app>**

**JSP : Java Server Pages : JSP is a tag based scripting language on server side which help to create dynamic web page.**

**Limitation of Servlet**

1. **Servlet is normal Java program if we do any changes we have to re-compile and re-deploy the application.**
2. **Servlet is a complex, if we want do display simple message through servlet, we have to create normal java class extends or implements type of servlet then override life methods, then create the object of PrintWriter class, then give the descriptor details using web.xml file or annotation.**
3. **If we want to write any presentation logic using Servlet those html tags must be inside println. ie HTML or presentation logic embedded inside a servlet.**

**JSP tags**

1. **Scripting tags**
   1. **Scriptlet tag**

**<%**

**Java coding (The code which we write inside a Java methods)**

**%>**

* 1. **Declarative tags**

**<%!**

**Variable declaration or object creation of java classes.**

**%>**

* 1. **Expression tags**

**<%=**

**%>**

1. **Directive tags** 
   1. **page**
      1. **<%@page import=”java.sql.\*”%>**
   2. **include**
   3. **taglib**
2. **Implicit object** 
   1. **out : It is PrinitWriter class reference.**
   2. **request : It is HttpServletRequest interface reference.**
   3. **response : It is HttpServletResponse interface reference**
   4. **config : It is ServletConfig interface reference**
   5. **application : It is ServletContext interface reference.**
3. **Actions tags** 
   1. **Jsp include**
   2. **Jsp forward**
4. **JSTL tags**
5. **Custom tags**

**Limitation of JSP**

1. **JSP itself is a type of Servlet. When we run jsp page internally convert back to Servlet page ie Page translation phase.**
2. **JSP is not secure.**
3. **JSP code can’t do re-usability.**

**JSP Model 1 Architecture**

**View – Presentation Logic -🡪 HTML/JSP**

**Model -🡪Business logic --🡪 Normal Java classes**

**JSP Model 2**

**MVC Model View Controller**

**View -🡪 Login.jsp / Login.html**

**Controller -🡪 LoginController (Servlet)**

**Model --🡪 Login.java**

**Model Layer**

**Login.java LoginSerivce LoginDao**

**Pure Java bean class business Logic Database**

**Index.jsp**

**Create.jsp post**

**Login.jsp get**

**LoginController.java**

**doGet()**

**doPost**

**Login.java**

**LoginService.java**

**LoginDao.java jdbc or hibernate or jpa**