Unit -01

Date 06/10/22
Page : 01

Introduction to Java Script

Q.1 Server क्या है? और कैसे काम करता है?

Answer:- Server -

★ Server या Web server एक ऐसा computer होता है जो website को चलाता है ये Actual में एक computer program होता है जो कि Web page को divide कर देता है जिसके द्वारा ही उन्हें request किया जाता है।

★ Web server का जो मुख्य उद्देश्य होता है वो ये कि web pages को store करें, उन्हें process करें और अंत web pages को deliver करे

Example - किसी search engine में कुछ search किया जाता तो ये एक request के रूप में उत्पन्न हो जाता है और ये request internet के द्वारा जिस search किये गये, word के server पर चला जाता है जहाँ उसका सारा Data store रहता वहाँ पर server आपके द्वारा request किये गये video या word ढूँढ कर उसका Data आपके Device पर भेज देता

★ Internet से हम जितने भी काम करते है जैसे कि - कोई file download करते है, Browsing करते है, Mail भेजते है, Social network site का use करते है इसके अलावा हम जितने भी काम करते है server हमारी मदद करता है और हम तक Data पहुँचाता है।

Q.2 What is Web Page?

Answer:- Information access करने के लिए हम जिन website को देखते है या use करते है वे HTML जैसी Markup languages में develop होते है इन भाषाओं में विकसित करने वाली website को technical रूप से Web document कहा जाता है और इन Web Document को अलग अलग Pages में divide किया जाता है जिसमें important informations होता है और उन्हें web pages कहा जाता है।

* हर website के पहले page को Home page कहा जाता है/इस Home page में website या कोई company के बारे में Introduction informations और other web page के link include होते हैं।

* एक Web page को access कर कंप्यूटर द्वारा सूचनाएँ एक विशेष program (Browser) computer या Mobile screen पर प्रदर्शित की जाती हैं।

* एक Web page generally एक text file होती है जिसमें बड़ी मात्रा में सामान्य text होता है और कुछ अन्य कुछ Hyper text होते हैं।

* Web page html या इसके समान अन्य Mark up languages में लिखे जाते हैं।

* एक Website, web page का समूह होगा, जो कि एक link द्वारा एक-दूसरे से जुड़े होते हैं। एक Website पर सबसे महत्वपूर्ण web page, index page है।

* www एक Multimedia आधारित network है इसलिए एक web page में text, graphics, audio, image और कई अन्य documents, यहाँ तक कि link भी हो सकते हैं।

Q.3 What is Website ?

Answer :- Website -

(1) एक website विशेष web document भी कहा जाता है। Web pages का एक set होता है और उन्हें एक विशेष Computer पर

रखा जाता है, जिसे web server कहा जाता है।

(2) हम Desktop, PC, Laptop, Tablet या Smart phone, TV की help से website को देख सकते हैं। किसी भी website के पहले Page को Home page कहा जाता है यह page मूलरूपतः Website के introduction page के रूप में कार्य करता है और इसमें अन्य Pages के link होते हैं जो उस site का हिस्सा होते हैं। कभी-कभी इसमें उन Pages के link भी होते हैं जो अन्य sites का हिस्सा होते हैं।

(3) Website का use किसी भी वस्तुओं का Advertisment देने या sales के लिए भी किया जा सकता है। इसके अलावा website का उपयोग information को अन्य लोगों के साथ transfer करने के लिए भी किया जा सकता है।

(4) एक Blog भी एक website होती है जहां information का स्थान वैसे लिखने वाले की सूचना कम प्रासंगिक है और जो अधिक ध्यान केन्द्रित करता है। हर website का पहचान उसके unique नाम से होता है जिसे URL (Uniform Resource locator) कहा जाता है।

(5) Amazon.in, Wikipedia.org, google.com, jio.com, website के example हैं। हम Website को public internet protocol द्वारा access कर सकते हैं। जैसे internet पर site की पहचान उसके URL द्वारा की जा सकती है।

(6) Website में कई work हो सकते हैं और विभिन्न प्रकार के कार्यों में उसका use किया जा सकता है। एक Website, व्यक्तिगत site, एक company की Business website, एक govt. website या एक गैर सरकारी संगठन website हो सकती है।

(7) World wide web पर important information या services को search करने के लिए एक website से दूसरे website पर जाने की प्रक्रिया को Surfing कहा जाता है।

(8) World Wide Web पर प्रत्येक website को एक विशिष्ट पहचान देने के लिए उसे URL या Web Address provide किया जाता है।

(9) एक URL के following parts होते हैं-

http://www.mywebsite.com

Protocol Domen Name Domen name extension
SubDomen

(a) http:-

- http means Hypertext transfer protocol यह एक Protocol है जिसके द्वारा website का access किया जा सकता है और web server के साथ communication स्थापित है।

(b) www -

- यह उस server के प्रकार का प्रतिनिधित्व / Interpret indigate करता है जिसके साथ हम communication कर रहे हैं।

(c) Domen Name (my website)-

- यह उस website का actual name या Organization name को indigate करता है जहां सभी web page available रहते हैं।

(d) Domen Name Extension (.com) :-

- यह Organization के बारे में indicate करता है।

	Domain	Types of Domain
(1)	.com	Commercial Organization
(2)	.edu	Educational Organization
(3)	.gov	Government Organization
(4)	.org	Non-Government Organization
(5)	.mil	Military
(6)	.Net	Network Service Organization

* Front End and Back end :-

(1) Front end and Back end, web industry में use किये जाने वाले दो सबसे महत्वपूर्ण शब्द word है या Software engg के दो important part है जो Web development important भूमिका निभाते है।

(2) Front end वह होता है जिसके साथ हम interact करते है और देखते है कि यह सब कैसे work करता है। Frontend graphical user interface को produce करता है। जब कि Backend actual program end को introduce करता है। जो frontend के according work करता है। Web development के लिए ये दो term very important है लेकिन एक दूसरे

(3) Frontend graphical user को वह Web browser से प्राप्त होता है जैसे - button, check box, colour image, navigation menu

e/c. Frontend को client side के रूप में implement किया जाता है। क्योंकि कार्य वहीं client side पर होते हैं जो user से related होती है।

(1) एक user जो पूरी की website पर देखता है या उसके साथ communication करता है। वह सारी Front end का हिस्सा होता है। Design के विषय में एक बेहतर user अनुभव और उसके उपयोग को आसान बनाने के लिए एक अच्छी website design करने के लिए जिम्मेदार होता है।

(2) एक Web designer न केवल code के साथ work करता है बल्कि वह user के अनुकूल visual प्रदान करने वाले element और website के निर्माण वह पुनः निर्माण करने वाले सभी पहलू जिम्मेदार होते हैं। Web Designer एक ऐसा वातावरण बनाते हैं जिसे user, Css, Java Scripting, Html, tool की सहायता से देख सकते हैं।

★ Back End :-

(1) Back end वह भाग होता है जिसे देख नहीं सकते है Communicate नहीं कर सकते Server side के रूप में जाना जाता है। Actual में यह Web development के लिए पर्दे के पीछे होने वाला हर काम के लिए जिम्मेदार होता है।

(2) यह उस प्रणाली का हिस्सा है जो सीधे user के संपर्क में नहीं आता है लेकिन ठीक से कार्य करने के लिए Front end के साथ संचार करता है। प्रत्येक application में non-user interface code की एक important भूमिका होती है और Backend में होने वाले सभी जटिल प्रणाली से संबंधित होता है, जो आमतौर पर Programmer और Developer होते हैं।

Client-Server Model:

① Client server Model वह होता है जो LAN, WAN, internet के माहौल में work करता है इस Model में एक server होता है। जिससे client network में मौजूद client computer को manage किया जाता है, इसमें resources या service provide किया जाता है।

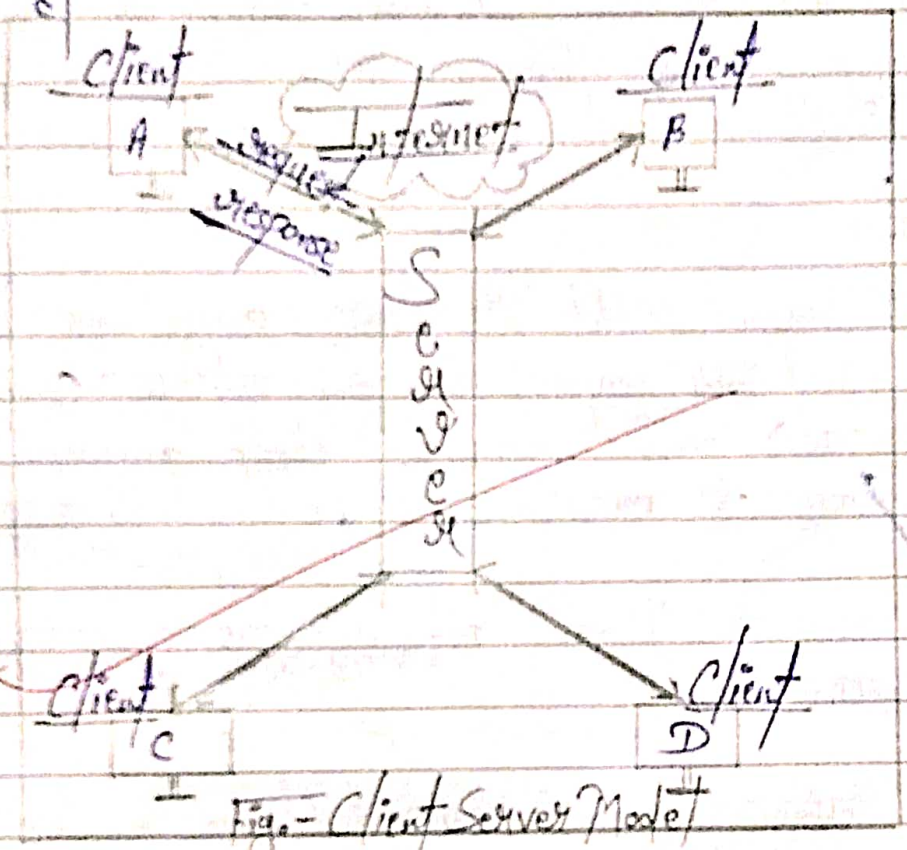


Fig:- Client Server Model

Q. What is client ?

Answer: ① Client एक computer system है जो किसी एक network के जरिये अन्य computers पर service पर exchange करता है।

② Client एक क्रिया है जो server को संदेश भेजता है और server उस काम को complete करता है।

③ client programs कम्प्यूटर पर application के user interface

हिस्से का management करते हैं। client आधारित प्रक्रिया application का frontend है जिसे उपयोगकर्ता (users) देखता है। उससे contact करता है।

(4) Client प्रक्रिया स्थानीय संसाधनों का Management भी करता है। यह user को monitor, keyboard, work-station, CPU जैसे को interact करता है। Client work station के प्रमुख भागों में से एक graphical user interface है।

Server:-

(1) Client Server Model में, Server process एक ऐसा program है, जो client द्वारा request किये गये कार्य को Complete करता है। आमतौर पर server program client program से request प्राप्त करता है। तथा client को response करता है।

(2) Server आधारित Process network की दूसरी मशीन पर भी चल सकता है।

(3) Server प्रक्रिया / Process एक Software engine के रूप में कार्य करती है। जो share संसाधनों Example - Database, printer, संचार link या High processor से संबंधित होती है। Server Process backend कार्यों को execute करती है।

Example of Client Server Model:-

(1) Client Server Model का first example e-mail server से ले सकते हैं जहाँ पर एक central location पर e-mail server रखा जाता है। जिस विभिन्न location (lan/wan) से e-mail application द्वारा users अपने computer, laptop या Mobile phone पर access कर पाते हैं। जो यहाँ पर E-mail server centr

ally emplement किया जाता है और विभिन्न जगहों पर मौजूद (E-mail client application) server application से Authenticate कर e-mail send या receive कर पाते हैं।

② Client Server Model का second example LAN network में install किये जाने वाले Antivirus solution को ले सकते हैं। यहाँ पर किसी एक अच्छे या performance देने वाले computer पर antivirus server application को install किया जाता है इस server application द्वारा client application package को generate किया जाता है।

फिर इस client Package को LAN पर मौजूद सभी computers पर install कर दिया जाता है जिससे Client application एवं server application आपस में संपर्क स्थापित कर पाते हैं और Client application किसी भी प्रकार का data जैसे update, setting या alert etc. server पर application से ही प्राप्त करते हैं।

Advantages of Client Server Model-

① इसका पहला Advantage इस model का centrallized होता है। यहाँ पर एक central server द्वारा network में मौजूद बहुत से clients को एक साथ manage किया जाता है।

② यह network model काफी safe होता है क्योंकि services और resources को centrally manage किया जाता है।

③ इस network model के flexible होने के कारण इसमें नये clients को कभी-भी बढ़ाया या घटाया जा सकता है।

④ इस model में Data centrally share होता है जिससे Data की सुरक्षा काफी बढ़ जाती है।

(5) इसका result काफी अच्छा होता है क्योंकि इसमें Dedicated server का use किया जाता है।

* What is the Difference between Scripting language and Programming language ?

Answer

	Scripting language	Programming language
(1)	Scripting language एक programming language होता है जो किसी दूसरे Programming language के साथ communicate करने के लिए design किया गया है।	Programming language instruction का set होता है, जिससे आप computer को ये बता पाते हैं कि उसे क्या करना है? Computer की language को Programming language कहा जाता है।
(2)	Scripting language में compile नहीं होता है यह केवल interpreter के द्वारा interpret execute होता है। (line by line)	Programming language में compile और interpret दोनों होता है सबसे पहले compiler के द्वारा compile होता है और उसके बाद interpreter के द्वारा interpret होता है।
(3)	Scripting language को मुख्यतः दो भागों में divide किया जाता है (a) client side scripting (b) server side scripting.	Programming language को मुख्यतः दो भागों में divide किया गया है- (a) Low level programming language (b) High level programming language
(4)	Scripting language, Programming language की तुलना थोड़ा slower होता है।	Programming language faster होता है क्योंकि यह एक बार compile हो जाने के बाद fastly interpret होता है।
(5)	Scripting language अकेला काम नहीं कर सकता है। यानि की	Programming language अकेले या कुछ languages के मिलकर भी काम कर

Notepad, Browser की जरूरत होती है।	चलता है।
⑥ Example of Scripting language Javascript, Visual Basic.	Example of Programming language C, C++, Java.

★ Client side and Server side Scripting language

① Script :-

- एक Script आमतौर पर Program निर्देशों की एक फाइल होती है, जिसे अन्य application द्वारा Execute किया जाता है। Web client server Environment में work करता है। इसलिए Script को दो रूपों में लिखा जा सकता है। एक Server end (Backend) और दूसरा Client end (Frontend)।

- Client side script, client side पर code को access करती है जो कि user को दिखाई देता है जबकि Server side script, server end पर Execute होती है जिसे user देख नहीं सकते।

- Server side scripting और client side scripting के बीच main difference यह है कि Server side scripting उन मूल्यों को प्राप्त करता है, जो Server पर Processing के लिए आवश्यक होते हैं जबकि Client side scripting में Script या client Machine को चलाने के लिए Browser की आवश्यकता होती है।

Client Side Scripting -

① Client side Scripting एक Code generate करती है जिसे Client end (Browser) पर चलाया जा सकता है। इस प्रकार के Scripts को मूल रूप से Html document के अन्दर रखा जाता है। Client side scripting का उपयोग user input के अनुसार Submit करने से पहले validation के लिए user के form को जाँच कर सकता है।

② एक सम्भावी Client side script, Server के भार को काफी कम कर सकती है। यह एक Web browser का use एक Host program के रूप में करते हुए Scripting language को चलाने के लिए Design की गई है। For example - जब कोई user server को Web browser द्वारा Web Page के लिए अनुरोध करता है, तो Html और Css साधारण Text के रूप में भेजा जाता है। और अंत में Browser Client end को Web content उपलब्ध करवाता है।

③ Html, Css, Java script अधिक उपयोग की जाने वाली Client side scripting language है अंत में Client side और Server side scripting co-ordinate manner से एक-दूसरे के साथ कार्य करती है, ताकि Website ठीक से कार्य कर सके।

Server Side Scripting -

① Server side scripting, Code निर्माण के लिए Programming की एक ऐसी तकनीक है, जो Server side पर Software चलती है। सरल शब्दों में, कोई भी Scripting या Programming भाषा जो Web Server पर चल सकती है, Server side scripting के रूप में जानी जाती है।

② Server Side में Three Parts (Database, API, Backend web software) include होते हैं।
इन सभी भागों को Server side scripting language द्वारा तैयार किया जाता है जब कोई Browser server side scripting वाले Web Page के लिए Server को एक request भेजता है तो Web server को Page search करने से पहले Script को process करता है यहाँ Script processing में database से information निकालना, Calculation करना या Customer को प्रदर्शित करने वाली उपयुक्त सामग्री को select करना शामिल हो सकता है। इसके लिए बहुत-से Programming language Developed की है जैसे- PHP, Python, Java, C, C++ etc.

★ Introduction of HTML-

① Web page को बनाने के लिए Program को लिखने हेतु प्रयुक्त की जाने वाली Text oriented, Standard Markup language होती है। HTML को Hyper Text Markup Language के रूप में परिभाषित किया जाता है।

② HTML एक Hyper Text Markup Language है, जिसे Web document (Web Page) बनाने के लिए Developed किया गया है।

③ इसका विकास 1990 में Tim Berners Lee के द्वारा किया गया था। Tim Berners Lee को father of HTML कहा जाता है यह Web Page का Base (आधार) होता है। और Web Page एक Website का base होता है।

④ HTML, Web document बनाने के लिए "Tags" का इस्तेमाल करते हैं।

Hypertext -

① Hypertext वह Technique है, जिसके द्वारा Web को Explore किया जाता है यह एक General text होता है लेकिन Hypertext अपने साथ किसी अन्य Text को जोड़े रखता है जिसे Mouse click, Tab से click करके activate किया जाता है।

② इसकी यही विशेषता इसे साधारण Text से अलग करती है। Hypertext को Hyperlink कहते हैं।

③ HTML के anchor tag <a> के द्वारा किसी भी Text को Hyperlink बनाया जा सकता है इसके अलावा image, video, Sound आदि को Hyperlink बनाया जा सकता है। इस प्रकार का link data, Hyper media कहलाता है।

④ Note - HTML Document के साथ compile करने या और कुछ भी अन्य करने की जरूरत नहीं होती जैसा कि हमें कोई अन्य Programming language में करना होता है इसकी कोई Web Browser द्वारा observation की जाती है।

Markup -

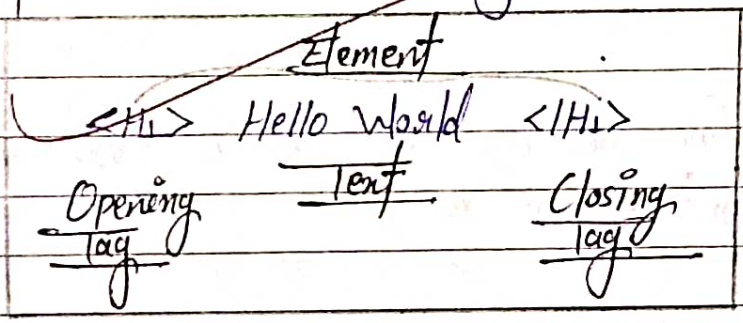
① HTML, Web document बनाने के लिए "HTML tag" का उपयोग करती है। प्रत्येक HTML tag अपने बीच आने वाले Text को किसी प्रकार में परिभाषित करता है। इसे ही Markup कहते हैं।

② <i> ÷ ① I एक HTML tag है जो अपने बीच आने वाले Text को Italic (तिरछा) करता है।

② This example, हम एक Word देते हैं "Hello CSE" जिसे साधारण भाषा में written किया गया है जो हमें सामान्य Text की तरह Hello CSE दिखाई दे रहा है अब हमें इसे HTML के द्वारा Markup करते हैं और Hello CSE के सामने और अंत में   लिखा जायेगा तो यह Hello CSE Italic form में दिखाई देगा। इस पूरे Process को ही Markup करते हैं।

Language -

① HTML, एक language है क्योंकि यह Web document बनाने के लिए code words का use करती है जिन्हें Tags कहते हैं और इन Tags को लिखने के लिए Html का syntax भी है। इसलिए यह language भी है। नीचे Html का Syntax दिया गया है।



② इसके तीन मुख्य भाग होते हैं- जो क्रमशः निम्न हैं-

ⓐ Element
ⓑ Tag
ⓒ Attribute

③ Html, element, Html tag से मिलकर बनता है Angle bracket के बीच जो Word लिखा होता है उसे Html tag कहते हैं। यह दो प्रकार का होता है-

① Opening tag
② Closing tag
③ Text

Characteristics of Html-

① यह एक बहुत ही आसान और सरल language है इसे आसानी से understand किया जा सकता है।

② यह Html के साथ प्रभावी Presentation बनाने के लिए बहुत ही आसान है। इसमें बहुत सारे formatting tags होते हैं। यह Programmer को Text, Tables, Graphics, Video एवं Sound को जोड़ने की सुविधा देती है, जो इसे और अधिक interactive बना देते हैं।

③ यह Platform independent होता है क्योंकि इसे किसी भी Platform जैसे Windows, Linux, एवं Macintosh आदि पर प्रदर्शित किया जा सकता है।

④ यह Programmer को Web Page पर links को जोड़ने की सुविधा देता है (By Html Anchor tag) इसलिए यह user की Browsing की expectation को बढ़ाता है।

Html Element, Tags and attributes :-

① Html Element-

An elements in Html represent some kind structure or semantic and generally consist of a starting tag and end tag.

```
<tag> Content </tag>
```

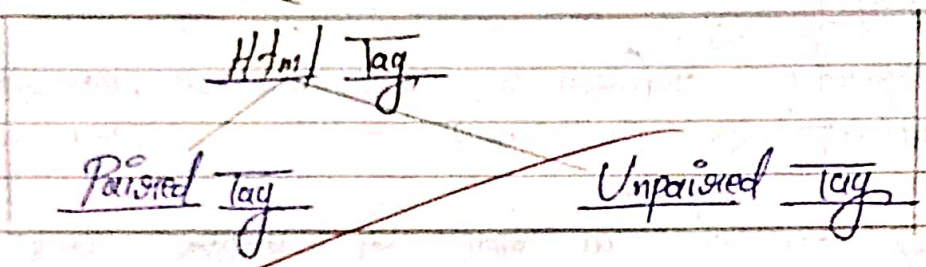
Example- The following is a paragraph elements-

```
<P>This is the Paragraph element </P>
```

② Html Tags-

① Tag ऐसे instruction होते है जो Document text में सीधे embedded होते है। एक Html tag किसी Browser के लिए ऐसा signal होता है जिसमें Html tag एक Open angle bracket (<) से शुरू होते है एवं एक close Angle bracket (>) से समाप्त होते है।

② Html tag दो प्रकार के होते है-



③ Paired Tag -

- एक tag को तब Paired tag होना कहा जाता है जब वह text के side में साथी tag के साथ हो।

- एक Paired tag होता है के अपने साथी tag के मध्य इन दोनों के बीच लिखे text को Bold में परिवर्तित किये जाने का कारण बनता है।

④ Unpaired Tag -

- The second type is of tag is unpaired tag also known as the singular and stand alone tag.

- एक unpaired tag का कोई साथी tag नहीं होता है।
Example-
 - एक line को तोड़ने के लिए (break) के लिए insert किया जाता है इस tag को किसी साथी tag की आवश्यकता नहीं होती।

② Html Attributes -

- Html tag को उनके सर्वोत्तम रूप में देखा गया है लेकिन अधिकांश Html tag में Attribute भी हो सकते हैं।
- ये Attributes को किसी Html element की विशेषताओं को Introduce करने के लिए प्रयुक्त किया जाता है एवं इसे Element के Opening tag के अंदर रखा जाता है।
- Attributes tag के मध्य Data के लिए अतिरिक्त element Properties (such as colorization, measurement, location, Alignment) एवं अन्य रूपों में प्रदान करते हैं।
- सभी Attributes, दो भागों से मिलकर बनते हैं- Name and Value.

Example-

```
<tag name = "Value"> Content </tag>
<p align = "Center"> This is paragraph that is centered </p>
```

- **Note-** Following Example में Paragraph <p> tag के attribute को दर्शाता है जिसका name align है जिसे हम Page पर paragraph के Alignments को इंगित करने के लिए use कर सकते हैं एवं यह Value को Center में लाता है।

```
<p align = "Center"> This is paragraph that is centered </p>
```

Starting Tag — Attributes — Value — Content — End Tag

★ Empty Html Element-

① Empty element (जिसे self closing और Void elements भी कहा जाता है) इसमें Container tag नहीं होते हैं।

Example-

<P> This paragraph Contain
 a link break </P>

②

★ Html को start करना-

① Html coding start करने के लिए हमें दो वस्तुओं की आवश्यकता होती है एक सरल text editor या Notepad एवं एक Web Browser.

① Creating the Html file-

① Computer के साधारण text editor को Open करें एवं नई file बनायें।

② Type some Html Code-

① एक खाली Window से प्रारंभ करें एवं कुछ Code लिखें।

<Html/> _ _ _ _ _ _ _ _ _ _ <\Html/>

③ Saving the file-

① अब file को ".Html" या ".htm" extension (Example- MyPage.Html) से save करें। याद रखें Computer पर एक ऐसी जगह पर Document को save करें जिसे हम फिर से पा सकें।

(4) Open the Html Document:-

① अब हम Web Browser की मदद से Html Document के परिणाम को देख सकते हैं/ File को इस Web Browser में खोलने के लिए file पर जायें फिर उस पर Double click करें / यह हमारे Default web Browser में खुलेगा, यदि ऐसा नहीं होता है, तो अपने Browser को खोलें और उस पर file को Drag करें/

★ Writing first HTML Document:-

We are know ready to create our first web page. for, example start with an empty window of Notepad and type the following code-

```
<html>
<head>
<title> This is my first web page </title>
</head>
<body>
    Hello World
</body>
</html>
```

Output:- Hello World

① Now save the file .html or .htm extension (For Example - Hello world.html) and open in the Browser. The above Html code will produce page in our web Browser as so in figure.

★ Basic Text Formatting -

① Internet पर किसी भी समय के अवधि के बाद हम देखेंगे कि हम एक Screen पर एक Webpage सिर्फ सामान्य शब्दों के आधिक के माध्यम से बना होता है। इसमें Headlines, Paragraph Graphics, Colour और बहुत कुछ होते हैं।

② Headline, paragraph and linebreak, horizontal look हमारे वर्तमान Page को बहुत अधिक Suitable बनाने में हमारी मदद करेंगे।

③ Html Headings -

① एक html heading अर्थात् html <h> tag को एक ऐसे शीर्षक या एक उप-शीर्षक के रूप में परिभाषित किया जा सकता है जिसे हम Web page पर प्रदर्शित करना चाहते हैं जब हम Heading tag के अंदर text को रखते हैं तो यह Browser पर Bold प्रारूप में प्रदर्शित किया जाता है एवं Text की size Heading की संख्या पर depend करती है।

②

```
<h1> Heading no. 1 </h1>
<h2> Heading no. 2 </h2>
<h3> Heading no. 3 </h3>
<h4> Heading no. 4 </h4>
<h5> Heading no. 5 </h5>
<h6> Heading no. 6 </h6>
```

(3) यहाँ पर 6 अलग-अलग Html heading है जिन्हें <h1> tag से <h6> tag से परिभाषित किया गया है।

(4) h1 सबसे बड़ा tag है और h6 सबसे छोटा है इसलिए h1 को Heading के रूप में use किया जाता है h6 को कम imp के लिए प्रयुक्त किया जाता है।

(5) Paragraph and line break:-

(1) Paragraph को <p> से represent किया जाता है Paragraph tag का मुख्यतः एक आमतौर पर देखा tag होता है जिसकी लगभग Web Page पर हमारे text को प्रस्तावित आवश्यकता होती है।

(2) यह एक Container tag होता है इसमें एक शुरुआती और अंतिम tag होना आवश्यक है।

Example:-

```
<P> This is a paragraph </P>
<P> This is another paragraph </P>
```

Output-

This is a paragraph
This is another paragraph.

(3)
 tag को एक नये Paragraph को प्रारंभ किये बिना एक line break को insert करने के लिए प्रयुक्त किया जाता है। यह एक खाली एवं रिक्त tag (stand alone) होता है जिसका कोई end tag नहीं होता है।

Example-

<P>This is a paragraph
 with the Break </P>
<P> This is
 another paragraph
 with line break </P>

Output :-

This is a paragraph
with the break
This is
another paragraph
with line break

⑤ Horizontal Rule-

① To create horizontal line a cross the screen, We use the <hr> tag. The Horizontal rule tag has no ending tag. That is, it is an empty tag.

Example-

```
<h1> HTML </h1>
<P> Html is a language for describing web
    Pages </P>
<hr>
<h1> CSS </h1>
<P>Css defines how to display Html element </P>
```

Output:-

HTML
Html is a language for describing web pages.

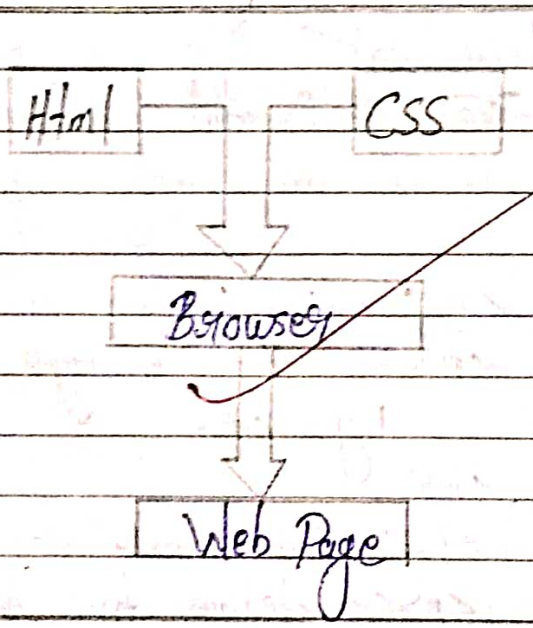
CSS
Css defines how to display Html element.

CSS (Cascading style sheets)

① CSS का Full form Cascading style sheets, यह एक मार्कअप लैंग्वेज है जिसके माध्यम से हम Website का Designing करते हैं।

② Simple शब्दों में कहें तो CSS आपके Browser को यह बताता है कि Web Page में उपस्थित Contents (जैसे - Text, Image, Paragraph etc.) के font, color, size, position आदि क्या होंगे।

③ CSS एवं Html की सहायता से Web Page बनाया जाता है Html, वेब पेज के size देता है और यही CSS का काम Web Page को attractive बनाना होता है।



Write a difference between Html and CSS -

① Html -

① Content, Which means हम Html से अपने Page में सिर्फ Content डालते हैं यहाँ Content का मतलब Text, Heading, Paragraph, Image, Video, audio, bullet, list etc. इसके अलावा Web Page का ऊपरी ढाँचा (structure) भी Html से बनाया जाता है

② CSS -

① Design, अब इस Content की color, Background, size, inside position etc. क्या होंगे? ये हम css में बताते हैं यानि Content Browser में कैसे दिखेगा यह css तय करता है।

Q. Why CSS is necessary and what are the benefits of the CSS?

① सिर्फ Html से एक Website बनाई जाती है जिसमें 50 Pages है अब हमें चाहिए कि सारे Pages की design same होनी चाहिए, तो हम एक Page को Design करते हैं और उसे Copy Paste करके 50 Page बना देते हैं इसमें कोई Problem नहीं है इसके बाद हम सभी Page में Content डाल देते हैं

② अब Website तो ready है लेकिन future में अगर हमें design में कुछ changes करने है जैसे कि हमें सारे Pages की colour Blue करना हो, तब हमें एक-एक Pages में changes करने पड़ेंगे, जो कि बहुत ही time consuming काम होगा।

③ अगर हम css use करते हैं, तो यह काम बहुत ही आसान होता है क्योंकि css से हम पेज के Design को Content वाले Parts से अलग कर देते हैं अब जब भी Design में changes की जरूरत पड़े तो हम आसानी से css की file में जाकर बदलाव कर देते हैं जो कि पूरी Website में apply हो जाता है।

④ Increase Website Speed -

① अगर हम Cascading style sheet का use करते हैं, तो Web Page का load होने की speed बढ़ जाती है ऐसा इसलिए क्योंकि हम css के लिए seperate file बनाते हैं और जब हम website की किसी एक Page पर visit करते हैं तो वह css file आपके Browser

Cache में store हो जाती है इससे बाद जब आप उसी side के दूसरे page पर जाएंगे तो फिर से आपके browser को उस css file को download करने की जरूरत नहीं होती।

(2) लेकिन यदि css use नहीं करते और आपने Html table से layout Design किया है तो Page loading slow हो जायेगी क्योंकि Browser को हर बार, हर Page में उस table को समझनी पड़ेगी।

(3) Device के According Design show करना -

(1) हम देखते हैं कुछ sites के Design desktop में अलग और Mobile में अलग दिखाई देते हैं जब आप youtube search वाले website से search का printout निकालते हैं तो only search वाला हिस्सा ही क्यों Print होता है? बाकी के Header, cyber क्यों Print नहीं होते?

(2) ऐसा इसलिए, क्योंकि वे, css के जरिये Devices के हिसाब से design show करते हैं, वे इससे Browser को बताते हैं कि अगर device mobile है तो क्या दिखाना है और device अगर Printer है तो क्या या कौन-सा हिस्सा Print करना है।

★ How, apply css in Web Page?

(1) Father of CSS Haykon Wium Lee को माना जाता है इन्होंने ही सबसे पहले 1994 में css rules को बनाया था और इसके बाद W3C (World Wide Web consortium) द्वारा css level 1 को दिसंबर 1996 में प्रकाशित किया गया, यह css का पहला Version था।

(2) जब हम किसी Web Page को Browser में open करते हैं, तो Browser सबसे पहले css style sheets को अलग करता है। Browser ये जानना चाहता है कि इस Webpage को किस style में दिखाना है? उसके बाद ही Web Browser उस

Date 7/10/22
Page : 27

Webpage को CSS के According ही दिखाता है।

Page: 23

① Introduction to Java Script:

① Netscape और Sunmicro system द्वारा संयुक्त रूप से विकसित JavaScript (JS) एक scripting language है जिसे HTML के विस्तार के रूप में माना जाता है।

② Java Script language "Brendon Eich" के द्वारा Develop की गई।

③ यह एक client side scripting language है आमतौर पर इसे HTML Pages में स्थित ही जोड़ दिया जाता है Java script HTML element event और webpage पर काम करती है।

④ Script, computer का कम knowledge रखने वाले user के लिए automatic होने वाले कार्यों को करने के लिए use किये जाने वाले छोटे Programs है Webpage के साथ Java applet जोड़ने के लिए Javascript को एक tool के रूप में introduce किया गया है। Java script, Java applet (event) properties based पर काम करता है।

② Features of Java Script:

Java script एक Popular या अधिक use होने वाली Programming language बन चुका है। Java script programming language को इतने Popular होने का एक महत्वपूर्ण कारण उसकी विशेषताएँ है-

① Java script, High level Programming language के अंतर्गत आता है जिसको वजह से इसे सीखना और समझना आसान है। Java script Programming language object पर आधारित scripting language है।

(a) JS एक light weighted और Powerful Programming language है।

(b) JS का वास्तविक नाम ECMA (European Computer Manufactures Association) script है।

(c) Java script का syntax, C Programming language के syntax से प्रभावित है।

(d) Java script, Web designer और Developer को Web page में Programming language use करने की सुविधा देता है।

(e) Java script की important बात यह है कि Web page के किसी element, text, image etc. को वांछित प्रकार कर सकता है जिसे Dynamic कहते है।

(3) Uses of Java Script-

Java script एक popular language है programming language है इसके बहुत सारे uses है जैसा आप use web application बनाने और web desktop page को active-live बनाना है यहाँ JS के सम्मुख uses दिए है-

(1) Dynamic और Animation के features से active web page बनाने के लिए Java script का use किया जाता है।

(2) Java script का use, Web application बनाने के लिए भी किया जाता है।

(3) Java script का use web Page में Dynamic menu बनाने के लिए किया जाता है।

Date 17.10.22
Page 51

(4) Website को interactive बनाने के लिए image slider का use किया जाता है।

(5) Website में अलग-अलग तरह के effect जैसे- Animation जैसे effect के लिए use किया जाता है।

(6) Website में cookies या suggestion create करने के लिए किया जाता है।

(7) इसके use से Web page में text या image को गतिशील बना सकते हैं।

(4) Advantage of Java Script

(1) Client side Programming language की तरह से JS बहुत fast होता है।

(2) यह किसी भी operating system और Web Browser पर काम करता है।

(3) इसे सीखना आसान होता है।

(4) यह Popular Programming language है और इसका update समय-समय पर आता रहता है।

(5) इसे अन्य Programming language के साथ use किया जा सकता है।

Ekm

Date 16/10/22
Page 52

⑤ Disadvantage of Java Script-

① यह अलग - अलग Web Browser पर अलग - अलग तरीके से काम करता है।

② Client side Programming language के वजह से Vulnerability ढूँढना आसान होता है इससे Security risk होता है।

⑥ ① Script -

Script एक Programm होता है जिसमें कई सारे Programming instruction यानि codes लिखे होते हैं जो कि Run time पर interpret होते हैं Run time पर interpret होने का तात्पर्य यह है कि जब कोई application Program में हो या चल रहा हो तब code को अलग करके Execute किया जाता है।

Example -

① ऐसे आसानी से समझने के लिए हम FaceBook का example लेते हैं। आप जब किसी Post को like करते हैं तब क्या होता है? या like button पर click करने से कितने Process होता है कि नहीं फिर भी हमारा काम हो जाता है।

② क्योंकि वहाँ पर Run time में एक Script execute होता है और वह बिना किसी compile के हमारा काम कर देता है। ऐसे Script को बनाने के लिए Scripting language का use किया जाता है जो कि एक प्रकार का Programming language है।

③ Scripting language की एक खासियत यह है कि यह किसी अन्य language के साथ communicate कर सकती है। इन

Script को Html जैसे language के साथ embedded किया जा सकता है यानि Html document में आप scripting के code लिख सकते हैं।

Note - यहाँ पर Program की Help से जानेंगे कि अपने Html Document में Java script का प्रयोग कैसे करेंगे ?

Write a Program to print Hello World in Java script -

```
<Html>
<head>
<title> First Webscript Program </Title>
</head>
<body>
<script>
document.Write ("Hello World");
</script>
</body>
</Html>
```

Output - Hello World.

② Java Script Syntax -

① A Java script consist of Java script statement that are placed within the <script> - - - - - - - - - - </script> Html tag in a web page.

(2) We can place script tag containing our Java script any where within our web page but it is preferred way to keep it within the Head tag.

(3) Head tag के अंदर <Script> से <Head> tag के अंदर से पता चल जाता है कि ये Java script code है और इसे कैसे Interpret करना है?

Syntax-

```
<script>
Javascript statements;
</script>
```

(4) The script tags are two important attributes -

```
<script language = "Javascript" type = "text/javascript">
    Javascript statements;
</script>
```

(5) <script> tag के दो Attributes होते हैं/ इन Attributes के द्वारा आप scripting language और type defined करते हैं-

(a) Language-

- इस attribute से scripting language define करते हैं जैसे- php और Java script।

(b) Type -

- ये Attributes बहुत आवश्यक होता है इससे अपने file का type निर्धारित करते हैं। जैसे कि "text/ Javascript".

Note- ये Java script define करने का standard syntax होता है। लेकिन चाहे तो language and type attributes के बिना भी <script> tag defined कर सकते हैं क्योंकि Java script को Html के लिए Default script language माना गया है। उसमें सिर्फ opening और closing <script> tag define करने की आवश्यकता होती है।

Write a Difference Between Java and Java script-

Java	Java Script
① Java एक object oriented Programming language है।	Java script एक Powerful scripting language है जिसका use Html code के साथ होता है।
② Java एक compile की जाने वाली Programming language है।	Java script एक interpret की जाने वाली scripting language है।
③ Java के short form में oops Programming language कहा जाता है।	Java script को oops scripting language कहा है।
④ Java Browser और JVM दोनों पर run हो सकते हैं।	Java script केवल Browser पर run होती है।
⑤ Java की coding को समझना कठिन होता है।	Java script की coding को समझना easy है।
⑥ Java के object, class पर आधारित है।	Java script के object prototype होते हैं।

* एक ATM machine है जो एक fully Programming machine है उसमें

इस Dummy मशीन को बनाया / sample बनाया Prototype कहलाता है / यहाँ फिर ~~सिर्फ~~ Prototype से side पर Program run होते हैं।

Good
Ravi
20/10/2022

Write a Program to find average of three number (39, 49, 41) in Java script.

```
<html>
<head>
<title> Average of three Number </title>
</head>
<body>
<script type = "text/Javascript">
var m1 = 39, m2 = 49, m3 = 41, avg ;
avg = (m1 + m2 + m3) / 3 ;
document.write ("Average = " .avg);
</script>
</body>
</html>
```

Output= Average = 43

mas Imp. Write a Difference Between Server side scripting and Client Side Scripting.

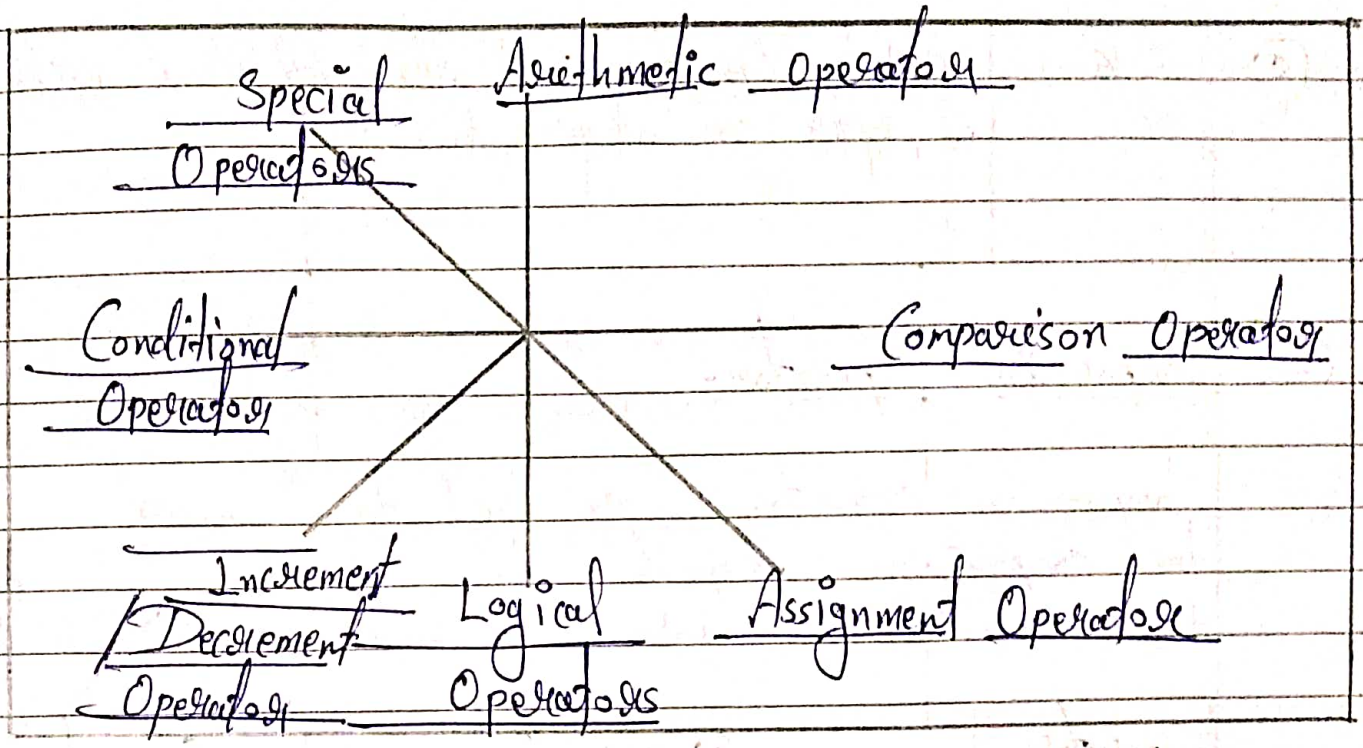
Basic for Comparison	Server Side Scripting	Client Side Scripting
① Basic	Work in the backend which in the	Work at the front-end and script are visible around the user
② Processing	Requires server interaction.	Does not need interaction with the server
③ Language involved	PHP, Python, Asp.net	HTML, CSS, Javascript
④ Affect	Could effectively customize the web pages and provide dynamic website.	Can reduse the load to the server.
⑤ Security	Relatively secured.	Insecure

~~most imp~~ ⑥ Java Script Operators -

① Operators एक sign (चिन्ह) होता है। जो Computer को कुछ Mathematical and logical operations करने के लिए कहता है। Operators का use Programm में Data और variable को Manage करने के लिए किया जाता है।

② Operators आमतौर पर Mathematical और logical expression का एक हिस्सा होता है।

③ Java Script Operators को नीचे दी गई निम्न श्रेणी में विभाजित किया गया है।



① Arithmetic Operators-

Arithmetic operators are the operators because they are used day to solve common mathematical expression. These are following Arithmetical Operators supported by Java script Language Assume variable A holds 10 and B holds 20 Then

	Operators	Description	Example
①	+	दो Operands को add करता है	A+B will give 30
②	−	दो Operands को Subtract करता है	A-B will give -10
③	*	दो Operands को multiply करता है	A*B will give 200
④	/	Divide numerator by Denomerator	B divide by A will give 2

(5)	%	Modulus operators and remender of after an integer division	B%A will give o

(2) Comparision Operator-

Comparision Operators are used to Compare value the comparision supported by Is and below. Assume variable A holds 10 and variable B holds 20

	Operator	Description	Example
(1)	==	Check करता है कि तथा दो Operands की value equal है कि नहीं यदि है तो Condition true हो जाती है।	A==B is not true.
(2)	!=	Check करता है कि तथा दो Operands की value equal है कि नहीं, यदि Condition दोनों equal नहीं है तो Condition true हो जाती है।	A!=B is true 10!=20
(3)	>	Check करता है कि left operand की value right operand की value बड़ी है, यदि है तो Condition true हो जाती है।	A>B is not true 10>20
(4)	<	Check करता है कि left operand की value right operand की value से छोटी है यदि है तो Condition true हो जाती है।	A<B is true 10<20

(5)	>=	Check करता है कि क्या left operand की value, right operand की value से बड़ी या equal है? अगर है तो condition true हो जाती है।	A>=B is not true 10>=20
(6)	<=	Check करता है कि क्या left operand की value, right operand की value छोटी या equal है? यदि है तो condition true हो जाती है।	A<=B is true 10<=20

(3) Logical Operator :-

Logical operator are used to perform boolean operand or Boolean operand AND, OR, NOT. The logical operator

	Operator	Description	Example
(1)	&& Logical AND	यदि दो variable की value true होती है तो ये operator true return करता है नहीं तो false return करता है।	True && True = True AND Operator condition A | B | Result True | True | True True | False | False False | True | False False | False | False
(2)	|| Logical OR	दोनों में से एक variable true है, तो result true return करेगा।	True || False = True OR Operator Condition

A	B	Result
True	True	True
True	False	True
False	True	True
False	False	False

(3) Logical NOT — यदि Condition true है और logical not operator आ गया तो condition false होगी और condition false है और logical not operator आ गया तो condition true होगी।

Condition	Result
! True	False
! False	True

(4) Increment and Decrement Operators :-

	Operator	Description	Example
(1)	++	Increment operator इससे operand की value एक हो जाती है।	a++. 100++ = 101
(2)	--	Decrement operator इससे operand की value एक कम हो जाती है।	a-- 100-- = 99

(5) Conditional Operators :-

Conditional Operators को किसी condition की जाँच करने के लिए एवं किसी condition की value के आधार पर एक value का चयन करने के लिए use किया जाता है।

व्यावहारिक रूप से चयनित से चयनित value के एक ऐसे Variable में Assign किया जायेगा जो निम्न Form होगा।

Conditional :-

Variable = (condition) : Value 1 : Value 2

जब यह Operator Computer द्वारा execute किया जाता है तब condition की value observe की जाती है यदि यह सही है तो Variable के लिए Value 1 assign की जाती है अन्यथा Variable value 2 Assign की जाती है।

Example :- A = (a>b) ? a : b

⑥ Special Operators :-

Java Script में ऐसे कई Special Operators को support करता है जो Operators की कोटियों में fit नहीं होते हैं।

① Delete Operators :-

इस Operator को किसी array, Index में किसी object या किसी element को delete Operator को element की Properties को delete करने के लिए use किया जाता है।

Syntax - delete Object [Property]

Example - Delete myobject [5] =

0	1	2	3	4

② Void Operator :-

यह Operator को value return नहीं करता है। सामान्यतः यह किसी URL को खाली मान (value) देने के लिए किया जाता है।

(7) Assignment Operator :-

Assignment Operator variable की values आपस में Assign करने के लिए प्रयोग किये जाते है। Java Script में प्रयोग होने वाले Assignment Operator को नीचे दिया जा रहा है।

	Operator	Description	Example
(1)	=	ये Operator, right variable की value को left variable को assign करता है।	a = b ;
(2)	+=	ये Operator, left और right variable की value को add करके left variable में store करता है।	a += b ;
(3)	-=	ये Operator, left side के variable की value में से right side के variable की value को घटाकर result left side के variable में store करता है।	a -= b ;
(4)	*=	ये Operator, left और right side के variable की values को multiply करके result, left side के variable में store करता है।	a *= b ;
(5)	/=	ये Operator, left side के variable की value को right side के variable से divide करके result, left side के variable store करता है।	a /= b

Q.1 Write a program to print one, two, three in Java script programming.

Program :-

```
<html>
<head>
<title> Print number </title>
</head> <body>
<script type = "text/Java Script">
    document.write ("One", "<br>");
    document.write ("Two", "<br>");
    document.write ("Three", "<br>");
</script>
</body>
</html>
```

Output -

One
Two
Three

Q.2 Write a program addition of two numbers of Java Script.

```
<html>
<head>
<title> Add two Number </title>
</head>
<body>
<script>
  var num one = 10;
  var num two = 20;
  var sum = num one + num two
  document.write ("Sum = " + Sum);
</script>
</body>
</html>
```

Output :- Sum = 30

Q.3 Write a program addition of two number of JavaScript.

Program :-

```
<html>
<head>
<title> Add two numbers </title>
</head>
<body>
<script>
  var a = 40;
  var b = 50;
  var c = a+b;
  document.write (c);
```

```
</script>
</body>
</html>
```

Output :-

90

Q4 Two Check Wheather the number even and odd is Java Script.

Program :-

```
<html>
<head>
<title> Even and Odd number </title>
</head> <body>
<script>
  var num = 6;
  if (num %2 ==0)
  document.write(num + "is an even number");
  else
  document.write(num + "is an odd number");
</script>
</body>
</html>
```

Output :-

6 is an even number.

Q.5 Write a program to Java script two add, subtract, multiply and divide.

Program :-

```
<html>
<head>
<title> Add Subtract multiply Divide </title>
</head>
<body>
<script>
  var num one = 12; , num two = 10, res;
  res = num one + num two;
  document.write ("Add =", res);
  res = num one - num two;
  document.write ("Subtract =", res);
  res = num one * num two;
  document.write ("multiply =", res);
  res = num one / num two;
  document.write ("Divide =", res);
</script>
</body>
</html>
```

Output :- Add = 22, Subtract = 2, multiply = 120, Divide = 1.2

Q.6 Write program in Java script Add two number using form and text box.

Program 8:

```
<html>
<head>
<title> Add two number form </title>
</head>
<body>
<script>
    function add( )
    {
    var num one , num two , Sum;
    num one = Parse Int(document.get element By Id ("First").value);
num two = Parse Int(document.get element By Id ("Second").value);
    Sum = num one + num two;
      document.get element By Id ("answer").value = Sum;
    }
<p> Enter the First number : <input id = "First"></p>
<p> Enter the Second number : <input id = "Second"></p>
<button onclick = "add( )"> Add </button>
<p> Sum <input id = "answer"></p>
</script>
</body>
</html>
```

Note :- यदि हम int देते हैं तो sum नहीं करेगा क्योंकि वह string मानेगा इसलिए Parse का use किया गया है।

`document.get element By Id (" ").value;` का use value लेना है।

Ex — Write a Program to find a loss number between two number in Java Script.

Program :-

```
<html>
<head>
<title> Less than number Program </title>
</head>
<body>
<script>
  var num = 7;
  if (num < 10)
    document.write(num + " is less than 10");
  else
    document.write(num + "is not less than 10");
</script>
</body>
</html>
```

output - 7 is less than 10.

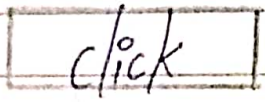
Q.8 Write a program to click a button and show.

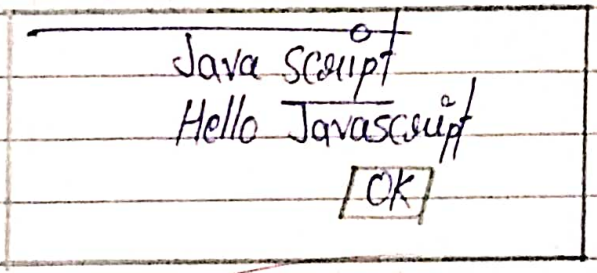
Program :-

```
<html>
<head>
<title> Show a alert message </title>
</head>
<body>
<script>
  Function SayHello()
```

```
{
  alert("Hello Javascript")
}
</script>
  <input type="button" values="Click" onclick="SayHello()">
</body>
</html>
```

Output :-





(4) (a) Java Script Variable :-

(1) Java Script variable को ऐसा मान सकते है कि यह users के द्वारा input किये गये value के लिए container है जिसमें Data value store होता है।

(2) Variable को declare करने के लिए var keyword का use किया जाता है।

(3) Variable, Java Script का कोई keyword नहीं हो सकता है।

(4) Variable case sensitive होता है "A" और "a" दोनों अलग-अलग variable है।

(5) Variable के नाम की शुरुआत किसी भी letter (A-Z या a-z) (-) या ($) sign से की जा सकती है।

(6) Variable की शुरुआत numeric value से नहीं हो सकता है लेकिन Variable alphanumeric हो सकता है।

For Example, A1 = 5

Example of Javascript variable :-

```
<html>
<head>
<title> Example of Javascript variable </title>
</head>
<body>
<script>
    var a = 5;
    document.write("Value of a is", a);
</script>
</body>
</html>
```

Output :- Value of a is 5.

(b) Variable Declaration in Javascript :-

(1) जब variable पर कोई value store नहीं की गई है, तो variable Declaration undefined होता है।

Program :-

```
<html>
<head>
<title> Example of Java script variable </title>
</head>
<body>
<script>
   var a;
   document.write("Values of a is "+a);
</script>
</body>
</html>
```

Output :- Value of a is undefine.

© Variable initialization in Javascript :-

① जब Variable initialize किया जाता है तब भी, variable declare करने के बाद भी initialize किया जाता है।

Program :-

```
<html>
<head>
<title>Variable initialization </title>
</head>
<body>
<script>
```

```
var a;
a=5;
var b = 10;
document.write("Value of a is"+a,"<br>");
document.write("Value of b is"+a);
</script>
</body>
</html>
```

Output :-

```
Value of a is 5.
Value of b is 10.
```

Note :- एक ही var keyword के साथ multivariable (,) से declare या initialize किये जा सकते हैं।

Program :-

```
<html>
<head>
<title>Variable initialization</title>
</head>
<body>
<script>
  var a=5, b, c = 10;
document.write("Value of a is "+a,"<br>");
document.write("Value of b is"+b,"<br>");
document.write("Value of c is ",c);
</script>
</body>
</html>
```

Output :-

```
Value of a is 5
Value of b is undefined
Value of c is 10
```

(d) Overwrite variable values :-

Javascript में variable की values को overwrite किया जा सकता है।

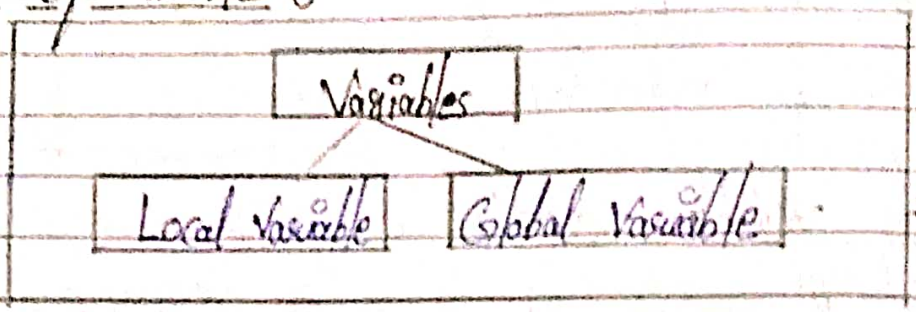
Program :-

```
<html>
<head>
<title> Overwrite variable values </title>
</head>
<script>
  var a = 5;
  document.write("Value of a is", +a "<br>");
  var b = 6;
  document.write("Value of b is", +b);
</script>
</body>
</html>
```

Output :-

```
Value of a is 5
Value of a is 6
```

© Types of variables :-



① Variables को मुख्य रूप से दो भागों में विभाजित किया गया है:-

ⓐ Local Variable
ⓑ Global Variable

ⓐ Local Variable -

ये सिर्फ function के अंदर ही visual है और local variable का जाता है।

Program :-

```
<html>
<head>
<title> Local Variable </title>
<head>
<body>
<script>
  function fun()
  {
    var a = 5;
    document.write("Value of a is " +a "<br>");
  }
  fun()
<\script>
<\body>
<\html>
```

Output :- Value of a is 5:

(b) Global Variable :-

यह function के अंदर या बाहर कहीं से भी access किये जा सकते हैं।

Program :-

```
<html>
<head>
<title> Global Variable </title>
</head>
<script>
    function fun()
    {
    document.write("Value of a is" +a, "<br>");
    document.write("Value of b is" b);
    }
    fun();
</script>
</body>
</html>
```

Output :-

```
Value of a is 10
Value of b is 5:
```

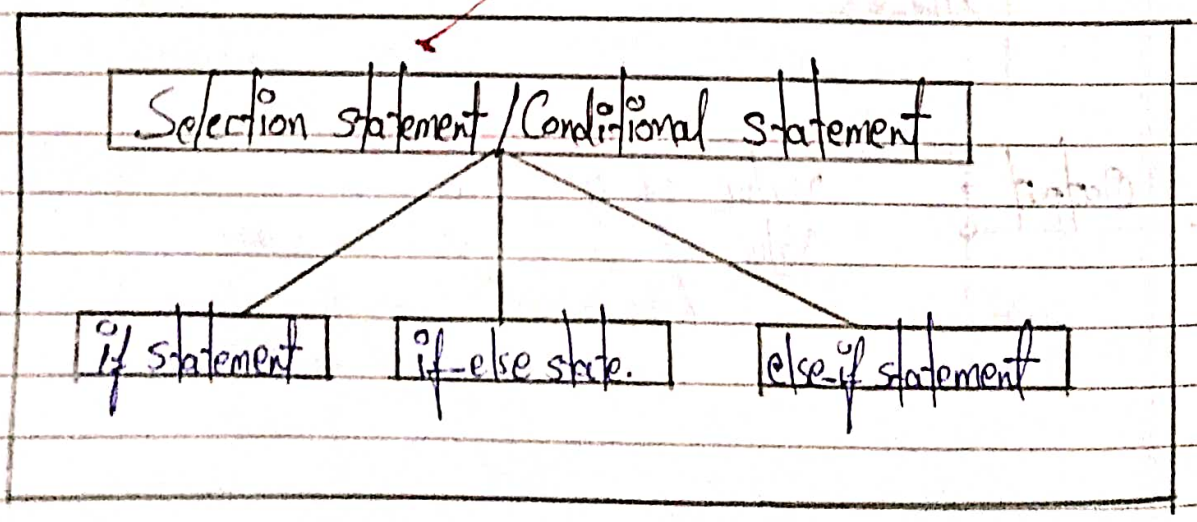
(5) Branching statement or Control flow Statement :-

(1) Branching statement or control flow statement give program the power to makes decision and perform task multiple times.

(2) Javascript provides the standard branching and looping structure that are available in many computer languages in fact, these structure where after those found in C, C++ and Java so if we have performed code in any of then languages, we will find these section very forward.

(3) एक Program को लिखते समय एक ऐसी स्थिति हो सकती है जब हमें दिये गये दो रास्ते में से एक रास्ते का चयन करने की आवश्यकता होती है उसलिए हमें ऐसे statement का use करने की आवश्यकता होती है जो हमारे Program को सही Decision लेने और सही Action Perform करने की Permission देते हैं।

(4) Javascript ऐसे तीन प्रकार के Conditional Statement को support करती है जिसे Different condition के आधार पर अलग-अलग action perform करने के लिए प्रयुक्त किया जाता है।



① if statement :-

① The if statement is the control statement that allows Java script to make decision and execute statement conditionaly.

Syntax :-

```
if (condition)
{
  statement to be executed if condition is true;
}
```

Q.1- Write a program in Java script to qualifies for driving in above age of 18.

Program :-

```
<html>
<head>
<title> Qualifies for driving </title>
</head>
<body>
<script>
    var age = 20
    if (age > 18)
    {
      document.write("Qualifies for driving");
    }
</script>
</body>
</html>
```

Output :- Qualifies for driving.

② if-else-statement :-

① if-else statement is the next form of control statement that allows Java script to execute statement in more control way.

② Here, Java script condition is evaluated if the resulting value is true given statement is the if block are executed. If the condition is false than given statement in the else block are executed.

Syntax :-

```
if (condition)
{
  Statement to be executed if condition is true.
}
else
{
  Statement to be executed if condition is false.
}
```

Example -

Q.1 Write a program to qualifies in driving in above age of 18 otherwise show the doesn't Qualifies for driving in Java script programming.

```
<html>
<head>
<title> if-else Example </title>
</head>
<body>
<script>
 var age = 15;
 if (age > 18)
 {
  document.write("Qualifies for driving");
 }
 else
 {
  document.write("Does not Qualifies for driving");
 }
</script>
</head>
</html>
```

Output - Does not Qualifies for driving.

③ Else-if Ladder-

① The else-if-ladder statement is the one level advance form of control statement that allows Java script to make correct decision out of several condition.

Syntax :-

```
if (condition 1)
{
 Statement to be executed if condition 1 is true.
}
else_if (condition 2)
{
 Statement to be executed if condition 2 is true.
}
else if (condition 3)
{
 Statement to be executed if condition 3 is true.
}
else
{
 statement to be executed if no condition is true.
}
```

Example -

```
<html>
<head>
<script type = "text/javascript">
var statecode = 'MO';
if (statecode == 'OR')
{
 taxPercentage = 3.5;
}
else if (statecode == 'CA')
{
 taxPercentage = 5.0;
}
```

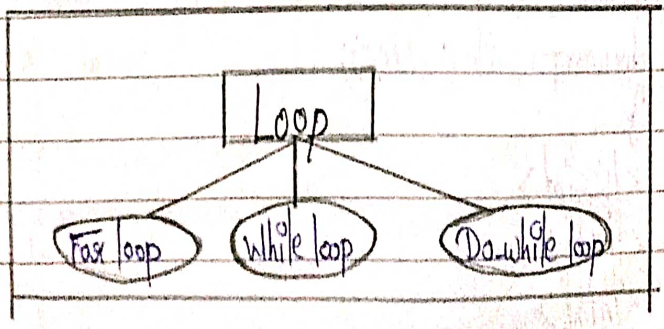
```
else if (StateCode == 'MO')
{
taxPercentage = 1.0;
}
else
{
taxPercentage = 2.0;
}
document.write(taxPercentage);
</script>
</head>
</body>
</html>
```

Output :- 1

★ Loop :-

① एक Programm को लिखते time एक ऐसी condition निर्मित हो जाती है जब हमें किसी Action को बार-बार करने की आवश्यकता होती है ऐसी स्थिति में हमें lines की संख्या को कम करने के लिए loop statement लिखने की आवश्यकता होगी।

② हमें Programming के सभी stage पर help करने के लिए Java script सभी आवश्यक loop को support करता है।



① for loop :-

① Java script Program में कोई ऐसा statement या Block of statement है जिसे आप multiple time execute करना चाहते हैं तो ऐसे situation में आप for loop का use कर सकते हैं। for loop में 3 expression का use किया जाता है।

Syntax :-

```
for(initialization ; test condition ; iteration statement)
{
  Statement to be executed if test condition is true
}
```

Example :-

Write a Program in Java Script print the number from 1 to 10 with the help of for loop.

Program :-

```
<html>
<head>
<title> Print the Number for loop </title>
</head>
<body>
<script>
for (i=1; i<=10; i++)
{
document.write(i)
document.write("<br>");
}
</script>
</body>
</html>
```

Output :-

```
1
2
3
4
5
6
7
8
9
10
```

(2) While loop :-

① The while is another type of loop supports the Java script. The purpose of while loop is execute the statement or code block repeatedly as long as expression as true, ones expression the becomes false the loop is existed.

Syntax :-

```
While (condition)
{
 Statement to be executed if expression
      is true;
}
```

Q.1 Write a program to print the number form 10 to 20. in Java script programming with the help of while loop :-

Date 14/11/22
Page : 66

Program :-

```
<html>
<head>
<title> While loop </title>
</head>
<body>
<script>
 var i = 10;
 while (i<=20)
 {
 document.write (i + "<br>");
 i++;
 }
 </script>
 </body>
 </html>
```

Output :-

10
11
12
13
14
15
16
17
18
19
20

③ do-while loop :-

do-while loop, while loop के समान ही होता है इसमें भी do-condition को check किया जाता है। इसका अर्थ है कि loop कम से कम एक बार execute किया जायेगा भले ही condition false हो।

Syntax -

```
do
{
  Statement to be Executed;
}
  while (condition)
```

Q.1 Write a program to print the even number of 2 to 20 in Java script Programming with the help of do-while loop.

Program :-

```
<html>
<head>
<title> while loop </title>
</head>
<body>
<script>
    var i=2;
    do
    {
      document.write(i);
      document.write("<br>");
        i=i+2;
```

Date 14/11/22
Page : 68

```
    }
    while (i<20)
</script>
</body>
</html>
```

UNIT- 2

What is jQuery

jQuery एक तेज़, छोटा, क्रॉस-प्लेटफ़ॉर्म और सुविधा संपन्न जावास्क्रिप्ट लाइब्रेरी है। यह HTML के क्लाइंट-साइड स्क्रिप्टिंग को सरल बनाने के लिए डिज़ाइन किया गया है। यह HTML डॉक्यूमेंट ट्रैवर्सल और manipulation, एनीमेशन, ईवेंट हैंडलिंग और AJAX जैसी चीजों को एक आसान से उपयोग करने वाले API के साथ बहुत सरल बनाता है जो कई तरह के ब्राउज़रों पर काम करता है।

JQuery का मुख्य उद्देश्य अपनी वेबसाइट पर जावास्क्रिप्ट का उपयोग करने का एक आसान तरीका प्रदान करना है ताकि इसे अधिक इंटरैक्टिव और आकर्षक बनाया जा सके। इसका उपयोग एनीमेशन जोड़ने के लिए भी किया जाता है।

jQuery एक छोटा, light weight और तेज़ जावास्क्रिप्ट लाइब्रेरी है। यह क्रॉस-प्लेटफ़ॉर्म है और विभिन्न प्रकार के ब्राउज़रों को support करता है। इसमें बहुत से सामान्य कार्य होते हैं जिन्हें पूरा करने के लिए जावास्क्रिप्ट कोड की कई लाइनों की आवश्यकता होती है, और उन्हें उन तरीकों में बाँधता है जिन्हें आवश्यकता पड़ने पर कोड की एक ही पंक्ति के साथ call किया जा सकता है। यह AJAX call और DOM manipulation की तरह जावास्क्रिप्ट से बहुत सारी जटिल चीजों को सरल बनाने के लिए भी बहुत उपयोगी है।

- jQuery एक छोटा, तेज और हल्का जावास्क्रिप्ट लाइब्रेरी है।
- jQuery प्लेटफॉर्म-स्वतंत्र है।
- jQuery का अर्थ है “Write less, do more” या “कम लिखो अधिक करो”।
- jQuery AJAX call और DOM manipulation को सरल करता है।

Features of jQuery:

निम्नलिखित jQuery की महत्वपूर्ण विशेषताएं हैं।

- HTML Manipulation
- DOM Manipulation
- DOM Element selection
- CSS Manipulation
- Effects and animation
- AJAX
- HTML Event handling
- JSON पार्सिंग
- प्लग-इन के माध्यम से एक्स्टेंसिबिलिटी

JQuery की आवश्यकता क्यों है?

- यह बहुत तेज और एक्स्टेंसिबल है।
- यह उपयोगकर्ताओं को न्यूनतम संभव लाइनों में UI से संबंधित फ़ंक्शन कोड लिखने की सुविधा देता है।
- यह एक आवेदन के प्रदर्शन में सुधार करता है।
- ब्राउज़र के संगत वेब एप्लिकेशन विकसित किए जा सकते हैं।
- यह ज्यादातर नए ब्राउज़रों की नई विशेषताओं का उपयोग करता है।

Add jQuery to website:-

Web pages में jQuery add करने के दो तरीके हैं:

1. jQuery library को jQuery.com से download करें.

2. Google या microsoft इत्यादि द्वारा प्रदत्त CDN का use करके अपने project में jQuery को include करना. -> **Recommended Method**.

1. jQuery library को jQuery.com से download करें

http://jquery.com/download/ official page को visit करके jQuery को download कर सकते हैं. यहाँ 2 version उपलब्ध है:

Production version: इस version को तब download करने जब, live website के लिए downloaded library को use करना चाहते हैं. **Development version:** इस version को testing और development purpose के लिए download करें. jQuery library को download करने के बाद webpage के साथ एक ही folder में रखें और इसका reference, website के प्रत्येक page में देने की आवश्यकता है. इसके लिए webpage के <head> section में <script> tag के अन्दर इस jQuery library के link को add करें.

```
<head>
<script src="jquery-3.4.0.min.js"></script>
</head>
```

1. jQuery library को CDN से use करें:

इस method में, jQuery file को download या host करने की आवश्यकता नहीं है.Google और microsoft CDN(Content Delivery Network) द्वारा प्रदत्त jQuery file को use कर सकते हैं. CDN fast होते हैं क्योंकि यह सबसे नजदीक के hosting server से file को deliver करते हैं.

jQuery file provided by Google CDN: इसे webpage के head section में refer करें:

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.0/jquery.min.js"></script>
</head>
```

---

jQuery Syntax

The jQuery syntax में HTML elements को select करने एवं element पर action perform किया जाना होता है.

jQuery basic syntax:

```
$(document).ready(function() {
   $(selector).action();
});
```

- $ sign define the jQuery.
- A (selector) defines the *Query element's* to find in HTML element's.
- And action() to be performed on the element's.

example:

$(this).hide() - hides the current element.

$("p").hide() - hides all <p> elements.

$(".test").hide() - hides all elements with class="test".

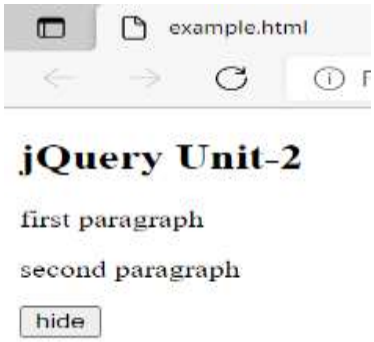
$("#test").hide() - hides the element with id="test".

First jQuery Example

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.0/jquery.min.js">
</script>
<script>
```

```
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide();
  });
});
</script>
</head>
<body>
<h2>jQuery Unit-@</h2>
<p>first paragraph</p>
<p>second paragraph</p>
<button>hide</button>
</body>
</html>
```

Output:



jQuery selector: -

jQuery selector एक function है जो expressions का use करके document object model(DOM) से matching element को find करता है. selector का use करके एक या अधिक HTML element को jQuery की सहायता से select करता है. एक बार element select होने के बाद, selected item पर विभिन्न operation perform करना आसान हो जाता है.

jQuery का use करके HTML element को उनके name, id, class, type attributes attributes की value के आधार पर find करता है.

jQuery में सभी selector , dollar sign($) और paranthesis ()से start होता है.

Systax: $(selector).action()

jQuery Selector Syntax

jQuery में निम्नलिखित basic selector है जो frequently use होते हैं:

Selector	Description
element Selector	दिए हुए सभी elements में से match element को select करता है.
this Selector	Current element को select करना.
#id Selector	दिए element के id से match करता है और उस element को Select करता है.
.class Selector	class के name के आधार पर element select करना.
*	Document के सभी element को select करना.

1. Element Selector

Element selector का use करके HTML document के tags या element को select किया जाता है. इसके लिए सीधे selector के अन्दर tag name को “” के अन्दर लिख दिया जाता है.

example:

$("p").hide() The jQuery hide() function, hide all <p> elements.

```
<script>
  $(document).ready(function() {
    $("button").click(function(){
      $("p").hide();
    });
  });
</script>
```

2. this selector

this selector का use करके current element को select किया जाता है.

$(this).hide() The jQuery hide() function, hide (this) element.

```
<script>
  $(document).ready(function() {
    $("button").click(function(){
      $(this).hide();
```

```
  });
 });
</script>
```

3. id selector

किसी element के id को match करके use element को select के लिए id selector का use किया जाता है. इसमें id की value के पहले # sign लिखा जाता है.

Syntax: $(“#id_name”).action_method();

Example:

$("#div1").hide() The jQuery hide() function, hiding whose id="div1" in the elements.

```
<script>
 $(document).ready(function() {
  $("button").click(function(){
   $("#div1").hide();
  });
 });
</script>
<p id="div1">First paragraph start here...</p>
```

4. class selector:-

जब class name का use करके किसी element को select करना हो तो class selector का use किया जाता है. इसमें selector में class name लिखने से पहले dot(.) लगाया जाता है.

example: $(“.class_name”).action_method();

$(".div1").hide() The jQuery hide() function, hiding whose class="div1" in the elements.

```
<script>
 $(document).ready(function() {
  $("button").click(function(){
   $(".div1").hide();
  });
 });
</script>
```

```
</head>
<body>
  <p class="div1">First paragraph start here...</p>
</body>
```

5. * selector

अगर document के सभी element को एक ही selector से select करना हो तो universal selector (*) का use किया जाता है.

$("*").hide() The jQuery hide() function, hide all the elements.

```
   <script>
   $(document).ready(function() {
     $("button").click(function(){
       $("*").hide();
     });
   });
 </script>
</head>
<body>
 <p>First paragraph start here...</p>
 <p>Second paragraph start here...</p>
 <button>Click here to hide above all paragraph</button>
</body>
```

Few custom selectors

Syntax	Description
$(":animated")	Currently animate होने वाले element को select करने के लिए.
$(":button")	किसी भी button element(inputs या button tag) को select करने के लिए.
$(":radio")	radio buttons को select करने के लिए.

$(":checkbox")	Checkboxes को select करने के लिए.
$(":header")	header elements (h1, h2, h3, etc..) को select करने के लिए.

jQuery Events

jQuery अपने user से interact होने के लिए action perform करता है, event ऐसे ही actions होते हैं, जिन्हें dynamic webpage में perform किया जाता है. events ऐसे actions हैं जो specific time में perform किये जाते हैं.

कुछ events:

- mouse click event
- mouse double click event
- submit a post request or form
- mouse hover on element

Ex. page element में click assign करने के लिए -

```
$("p").click();
```

Click के बाद perform किये जाने वाले action के लिए argument के रूप में function define किया जाता है-

```
$("p").click(function()
{
 //action goes here
});
```

Commonly used Event methods:

1. **$(document).ready()**: यह method, document के पूरी तरह से load होने के पश्चात् एक function execute करने के लिए allow करता है. jQuery में इस method का use document को select करने और आगे की processing के लिए तैयार करता है.
   Example:

```
$(document).ready(function(){
 ......
actions goes here
......
});
```

2. **click():** यह method किसी html element में event handler को attach कर देता है. इसके अन्दर का function, user द्वारा html element को click किये जाने के बाद execute होता है.
   Example: जब <p> element को click किया जायेगा. तब current <p>element hide होगा:

```
$("p").click(function(){
  $(this).hide();
});
```

3. **dblclick():** यह method किसी html element में event handler को attach कर देता है. इसके अन्दर का function, user द्वारा html element को double click किये जाने के बाद execute होता है.
   Example:

Function तब execute होगा जब HTML element पर user double-click करेगा.

```
$("p").dblclick(function(){
$(this).hide();
});
```

mouseenter()

mouseenter() method, event handler function में HTML element attach कर देता है. function तब execute होता है जब mouse pointer, selected HTML element में enter करता है.

Example:

```
$("#p1").mouseenter(function(){
  alert("You entered p1!");
});
```

mouseleave()

mouseleave() method, event handler function में HTML element attach कर देता है. function तब execute होता है जब mouse pointer, selected HTML element से हट(leave) जाता है.

Example:

```
$("#p1").mouseleave(function(){
  alert("Bye! You now leave p1!");
});
```

mousedown()

The mousedown() method, event handler function में HTML element attach कर देता है.

function तब execute होता है जब left, middle या right mouse button को pressed down किया जाये. जबकि mouse, HTML element के ऊपर हो.

Example:

```
$("#p1").mousedown(function(){
  alert("Mouse down over p1!");
});
```

mouseup()

The mouseup() method, event handler function में HTML element attach कर देता है.

function तब execute होता है जब left, middle या right mouse button को release किया जाये. जबकि mouse, HTML element के ऊपर हो.

Example:

```
$("#p1").mouseup(function(){
  alert("Mouse up over p1!");
});
```

hover()

hover() method, दो method mouseenter() और mouseleave() का combination है. इसलिए यह दो function को argument के रूप में लेता है. पहला function तब execute होता है जब mouse HTML element में enter होता है, और दूसरा function तब execute होता है जब mouse, HTML element को leave(छोड़ देता है) करता है.

Example:

```
$("#p1").hover(function(){
  alert("You entered p1!");
},
function(){
  alert("Bye! You now leave p1!");
});
```

jQuery effect methods

jQuery में html element को hide या show करने के लिए निम्नलिखित 3 method हैं:

1. show()
2. hide()
3. toggle()

1. **show():** इस jQuery method का use करके selected hidden html element को show (display) किया जाता है.

   syntax:

   $(selector).show(speed, callback);

   speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

   callback: optional, callback parameter एक function है, जो show complete होने के बाद execute होता है.

   Example:

```
$("button").click(function(){
 $("p").show();
});
```

2. **hide():** इस jQuery method का use करके selected visible html element को hide किया जाता है.

   syntax:

   $(selector).hide(speed, callback);

   speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

   callback: optional, callback parameter एक function है, जो hide complete होने के बाद execute होता है.

   Example:

```
$("button").click(function(){
 $("p").hide();
});
```

3. **toggle()**: यह method show() और hide() के बीच switch करता है. यह दोनों का combination है. यदि html element hidden(नहीं दिखाई दे रहा) है, तब toggle() इसे show कर देगा. यदि html element visible(दिखाई दे रहा) है, तब toggle() इसे hide कर देगा.

   syntax:

   $(selector).toggle(speed, callback);

   speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

   callback: optional, callback parameter एक function है, जो toggle method complete होने के बाद execute होता है.

   Example:

```
$("button").click(function(){
 $("p").toggle();
});
```

Jquery Fading effect

jQuery में fading के लिए निम्नलिखित methods हैं:

- fadeIn()
- fadeout()
- fadeToggle()
- fadeTo()

1. **fadeIn():** इस method का use करके hidden html element को धीरे धीरे उसकी दृश्यता को बढ़ाते हुए निश्चित समय बाद display कराया जाता है.

   syntax:

   $(selector).fadeIn(speed, callback);

   speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

   callback: optional, callback parameter एक function है, जो fading complete होने के बाद execute होता है.

   Example:

```
$("button").click(function(){
 $("p").fadeIn();
});
```

2. **fadeOut():** इस method का use करके visible html element को धीरे धीरे उसकी दृश्यता कम करते हुए निश्चित समय बाद गायब हो जाता है.

syntax:

$(selector).fadeOut(speed, callback);

speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

callback: optional, callback parameter एक function है, जो fading complete होने के बाद execute होता है.

Example:

```
1. $("button").click(function(){
    $("p").fadeOut();
   });
```

2. $("button").click(function(){
   $("p").fadeOut("slow");
   });
3.

3. **fadeToggle():** यह method fadeIn() और fadeOut() के बीच switch करता है. यह दोनों का combination है. यदि html element faded out (नहीं दिखाई दे रहा) है, तब fadeToggle() इसे fade in कर देगा. यदि html element faded in (दिखाई दे रहा) है, तब fadeToggle() इसे fade out कर देगा.

   Syntax:

   $(selector).fadeToggle(speed, callback);

   speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

   callback: optional, callback parameter एक function है, जो fading complete होने के बाद execute होता है.

Example:

```
$("button").click(function(){
 $("p").fadeToggle();
});
```

4. **fadeTo():** fadeTo() method, fadeIn() method की तरह ही है, लेकिन इसमें html element को एक निश्चित opacity level तक भी fade in कर सकते हैं.

syntax:

$(selector).fadeTo(speed, opacity, callback)

speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

opacity: Required, opacity parameter यह specify करता है कि target element की opacity 0 से 1 के बीच रह सकता है.

callback: optional, callback parameter एक function है, जो fading complete होने के बाद execute होता है.

Example:

```
$("button").click(function(){
 $("#B2").fadeTo(0.5);
});
```

jQuery sliding effect

sliding methods का use html element की height को धीरे धीरे कम या ज्यादा कर, ऊपर या नीचे slide करके hide या show किया जाता है. jQuery में sliding के लिए निम्नलिखित methods हैं:

- slideUp()
- slideDown()
- slideToggle()

1. **slideUp():** यह method, selected html element को sliding कर ऊपर करते हुए hide कर देता है.

   syntax:

   $(selector).slideUp(speed, callback)

   speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

   callback: optional, callback parameter एक function है, जो slidiing complete होने के बाद execute होता है.

Example:

```
$("button").click(function(){
  $("#B2").slideUp("fast");
});
```

2. **slideDown():** यह method, selected hidden html element को sliding कर नीचे करते हुए show कर देता है.

   syntax:

   $(selector).slidedown(speed, callback)

   speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

   callback: optional, callback parameter एक function है, जो slidiing complete होने के बाद execute होता है.

Example:

```
$("button").click(function(){
  $("p").slideDown(3000);
});
```

3. **slideToggle():** यह method slideUp() और slideDown() के बीच switch करता है. यह दोनों का combination है. यदि html element slide up (नहीं दिखाई दे रहा) है, तब slideToggle() इसे slide down कर देगा. यदि html element slide down (दिखाई दे रहा) है, तब slideToggle() इसे slide Up कर देगा.

   syntax:

   $(selector).slidedown(speed, callback)

   speed: optional, यह parameter effect के duration को specify करता है, इसकी value "slow", "fast" या millisecond हो सकती है.

   callback: optional, callback parameter एक function है, जो slidiing complete होने के बाद execute होता है.

   Example:

```
$("button").click(function(){
  $("h1").fadeToggle();
});
```

jQuery Animation methods

jQuery में animate() method का use करके custom animation create किया जाता है. इस method का use numeric css properties को animate करने के लिए किया जाता है. जैसे width, height, margin, padding, opacity, top, left इत्यादि. लेकिन basic jQuery functionality का use करके non-numeric properties जैसे color या back-ground को animate नहीं कर सकते.

syntax:

$(selector).animate({properties}, duration, callback)

properties: - required, यह properties parameter, animate होने वाले css properties को define करता है.

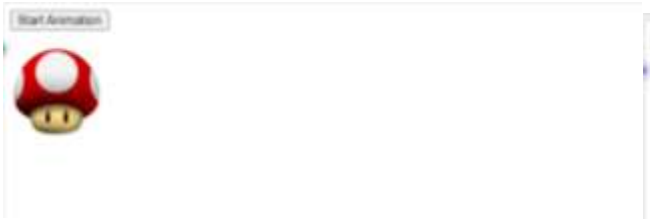
duration: - optional, duration parameter बताता है कि animation कितनी देर तक चलेगा. इसकी value 'slow' 'fast' या miliiseconds हो सकती है. miliisecond जितना ज्यादा होगा, animation, उतना ही slow होगा.

callback: - optional, callback parameter एक function है, जो animation complete होने के बाद call होता है.

Example : इस simple example में animate() method के द्वारा एक इमेज को animate करेगा जिसमें वह इमेज, button click करने पर origional position से 300 pixel right में शिफ्ट हो जायेगा.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of jQuery Animation Effects</title>
<script src="jquery.js"></script>
<style>
    img{
        position: relative; /* Required to move element */
    }
</style>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("img").animate({
            left: 300
        });
    });
});
</script>
</head>
<body>
    <button type="button">Start Animation</button>
    <p>
        <img src="/examples/images/mushroom.jpg" alt="Mushroom">
    </p>
</body>
</html>
```

Before click event: **After click event:**



Animate Multiple Properties At Once (एक साथ multiple properties को animate करना): **animate()** method का use करके किसी Element के multiple properties को एक साथ एक समय में animate करना संभव है. ये सभी properties बिना delay के साथ साथ animate होते हैं.

Note: animate() method का use करें तब **CSS properties** का name **camel-case** होना जरुरी है. जैसे यदि font size को animate करना हो तो font-size के बदले fontSize लिखना चाहिए.

```
<!DOCTYPE html>
<html>
<head>
<title>Example of jQuery Multiple Properties Animation</title>
<script src="jquery.js"></script>
<style>
    .box{
        width: 100px;
        height: 100px;
        background: #9d7ede;
        margin-top: 30px;
        border-style: solid; /* Required to animate border width */
        border-color: #6f40ce;
    }
</style>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $(".box").animate({
            width: "300px",
            height: "300px",
            marginLeft: "150px",
            borderWidth: "10px",
            opacity: 0.5
        });
    });
});
</script>
</head>
<body>
    <button type="button">Start Animation</button>
    <div class="box"></div>
</body>
</html>
```

Before click event: **After click event:**

jQuery stop() Method

jQuery stop() method का use selected element पर currently run हो रहे jQuery animation या effects को complete होने से पहले stop करने के लिए किया जाता है.

syntax:

```
$(selector).stop(stopAll, goToEnd);
```

stopAll: optional, यह boolean parameter specify करता है कि queued animation को remove करें या नहीं. default value false है. अर्थात, सिर्फ current animation stop होगा. और queue के अन्य animation run होता रहेगा.

goToEnd: optional, यह boolean parameter specify करता है कि current animation को तुरंत complete करें या नहीं. default value false है.

इस प्रकार by default, stop() method current animation को stop कर देता है.

example:

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("#flip").click(function(){
    $("#panel").slideDown(5000);
  });
  $("#stop").click(function(){
    $("#panel").stop();
  });
});
</script>
<style>
#panel, #flip {
  padding: 5px;
  font-size: 18px;
  text-align: center;
  background-color: #555;
  color: white;
  border: solid 1px #666;
  border-radius: 3px;
}

#panel {
  padding: 50px;
  display: none;
}
</style>
</head>
<body>

<button id="stop">Stop sliding</button>

<div id="flip">Click to slide down panel</div>
<div id="panel">Hello world!</div>

</body>
</html>
```

Before click event:

After click event: stops at middle

jQuery Callback Functions: -

callback function एक function है जो effect complete होने के बाद execute होता है. callback function को effect method में एक argument के रूप में pass किया जाता है और ज्यादातर यह method, last argument के रूप में दिखाई देता है. **JavaScript** statement, line by line execute होता है. लेकिन चूँकि, **jQuery** effect, complete होने के लिए कुछ समय लेता है, उसी दौरान next line execute हो रहा हो जबकि पुराना effect अभी भी run हो रहा हो, इसे रोकने के लिए jQuery**,** सभी effect method में callback function प्रदान करता है.

उदाहरण के लिए, **hide**() effect method का syntax निम्न है:

$(selector).hide(duration, callback);

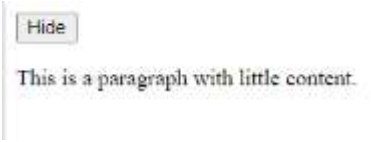
दिए example में hide() और alert() statement को use किये हैं,

```
$("button").click(function(){
  $("p").hide("slow", function(){
    alert("The paragraph is now hidden");
  });
});
```

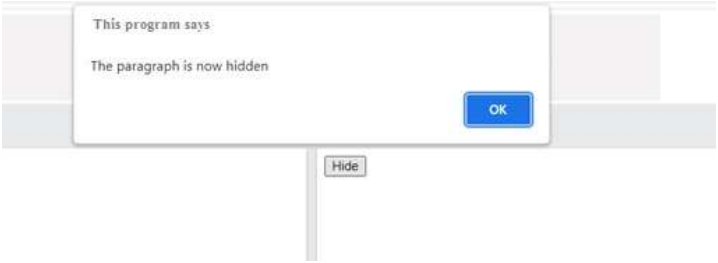
Example: -

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide("slow", function(){
      alert("The paragraph is now hidden");
    });
  });
});
</script>
</head>
<body>

<button>Hide</button>

<p>This is a paragraph with little content.</p>

</body>
</html>
```

Before click event:



After click event:



jQuery Method Chaining

jQuery, method chaining को support करता है, जिससे एक ही element के set में multiple action perform कर सकते है, और सभी single line वाले code से ही. ऐसा इसलिए संभव हुआ है क्योंकि ज्यादातर jQuery method, jQuery object return करता है जिसे आगे दुसरे method को call करने के लिए use किया जा सकता है. method chaining न सिर्फ, jQuery code को छोटा करता है, बल्कि script के performance को भी improve करता है, क्योंकि same element को खोजने के लिए browser को बार बार time देने की आवश्यकता नहीं होती.

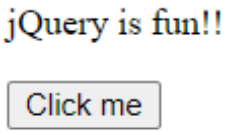
Methods का chain बनाने के लिए previous action के बाद दुसरे action को append किया जाता है. निम्नलिखित उदाहरण में css(), slideUp() और slideDown() method का chain बनाया गया है, “p1” element सबसे पहले red में change होगा, फिर slide up होगा और अंत में slide down होगा.

example:

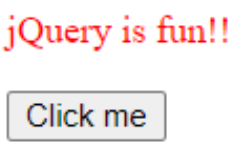
```
$("#p1").css("color", "red").slideUp(2000).slideDown(2000);
```

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("#p1").css("color", "red").slideUp(2000).slideDown(2000);
  });
});
</script>
</head>
<body>

<p id="p1">jQuery is fun!!</p>

<button>Click me</button>

</body>
</html>
```

Before click event:



After click event: 1. Red , 2. Slide up 3. Slide down



UNIT 3

jQuery HTML and AJAX

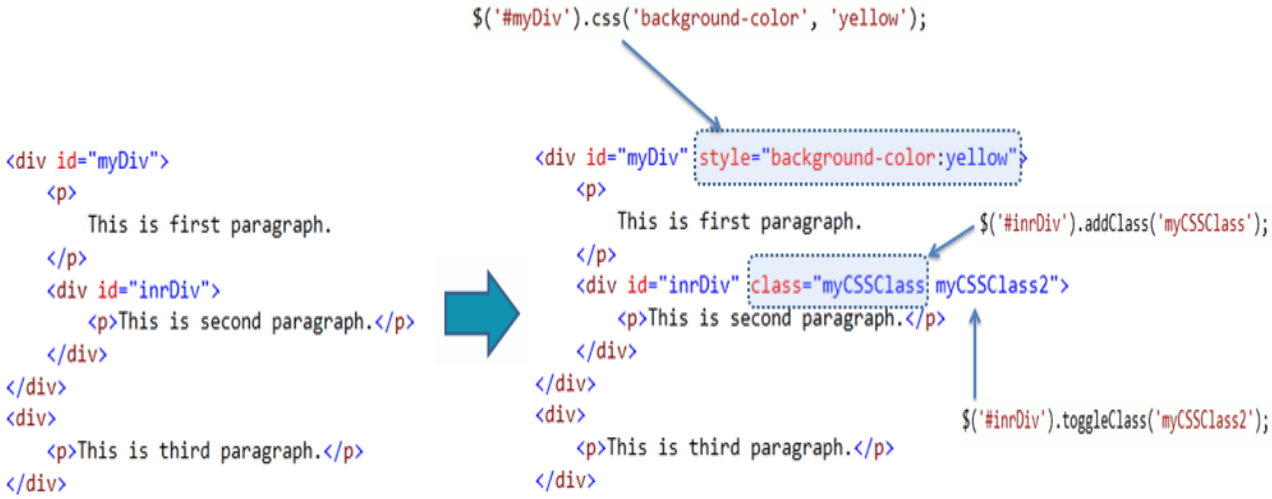
jQuery CSS Classes Manipulation

jQuery provides several methods to manipulate the CSS classes assigned to HTML elements.

The following table lists jQuery methods for styling and css manipulation.

jQuery Methods	Description
css()	Get or set style properties to the specified element(s).
addClass()	Add one or more class to the specified element(s).
hasClass()	Determine whether any of the specified elements are assigned the given CSS class.
removeClass()	Remove a single class, multiple classes, or all classes from the specified element(s).
toggleClass()	Toggles between adding/removing classes to the specified elements

The following figure shows how jQuery methods changes style and CSS class of the DOM elements.



jQuery Methods for Style & CSS Manipulation

jQuery css() Method

The jQuery css() method gets or sets style properties to the specified element(s).

```
$('selector expression').css('style property name','value');

$('selector expression').css({
                                'style property name':'value',
                            });
```

Specify a selector to get the reference of an elements to which you want to set the style property and then call css() method with style property name and value parameter. You can also set multiple style properties by passing JSON object with 'style property name':'value'.

Example: jQuery css() Method

```
$('#myDiv').css('background-color','yellow');
$('p').css({'background-color': 'red','width':'400px'});

$('#myDiv').css('background-color'); // returns rgb(255,255,0) for yellow color

<div id="myDiv">
    <p>This is first paragraph.</p>
</div>
<div>
    <p>This is second paragraph.</p>
</div>
<div >
    <p>This is third paragraph.</p>
</div>
```

Try it

In the above example, we set background-color of #myDiv and font styles to all <p> elements. The same way, we can get value of any style properties using css() method by specifying property name as first parameter.

jQuery addClass() method

The jQuery addClass() method adds single or multiple css class to the specified element(s).

```
$('selector expression').addClass('css class name');
```

First specify a selector to get the reference of an elements to which you want to set the css property and then call addClass() method with one or multiple class names as a string parameter. Multiple class names must be separated by space.

Example: jQuery addClass() Method

```
$('#myDiv').addClass('yellowDiv');
$('p').addClass('impPrg');

<div id="myDiv">
    <p>
        This is first paragraph.
    </p>
</div>
<div>
    <p>This is second paragraph.</p>
</div>
<div >
    <p>This is third paragraph.</p>
</div>
```

Try it

In the above example, we set css class of individual element (#myDiv) as well as multiple <p> elements using addClass() method.

jQuery toggleClass() Method

The jQuery toggleClass() method toggles between adding/removing classes to the specified elements.

```
$('selector expression').toggleClass('css class name');
```

Specify a selector to get the reference of an elements to which you want to toggle css classes and then call toggleClass() method with css class name as a string parameter.

Example: jQuery toggleClass() Method

```
$('#myDiv').toggleClass('redDiv');

<div id="myDiv" class="yellowDiv">
</div>
```

Try it

In the above example, css class yellowDiv will be first added into div element and then removed. Thus, css class will be added or removed consecutively.

Visit Manipulation methods reference to know all the CSS manipulation methods in jQuery.

 Points to Remember :

1. The jQuery CSS methods allow you to manipulate CSS class or style properties of DOM elements.
2. Use the selector to get the reference of an element(s) and then call jQuery css methods to edit it.
3. Important DOM manipulation methods: css(), addClass(), hasClass(), removeClass(), toggleClass() etc.

Unit-4

Basics of JSP:

JSP(java server page) एक server side programming technology होता है इसे dynamic, platform independent web page को create करने के लिए प्रयोग किया जाता है basically इसे java web application को develop करने के लिए create किया गया है.

- JSP technology java api(application program interface) को access कर सकती है जिससे java based web application को create करना और भी आसान हो जाता है
- JSP technology पूरी तरह से servlet technology के विपरीत काम करती है servlet technology में जहा java program के अन्दर html code को include करते हैं वहीँ jsp technology में html page में java प्रोग्राम को include करते है
- एक jsp page भी आउटपुट के रूप में servlet को generate करती है लेकिन jsp में coding, actual servlet से कई गुना अधिक आसान होती है
- एक jsp page, html page और jsp tags से मिलकर बना होता है jsp tags की मदद से java APIs को प्रयोग करते है इन tags की मदद से बड़े प्रोग्राम को आसानी से webpage में include कर सकते है

इस प्रकार JSP से web application को create कर सकते है जो कि java based होता है

java सबसे secure ,powerful और popular language होता है इसलिए एक java based web application दूसरी web application से fast secure और better मानी जाती है.

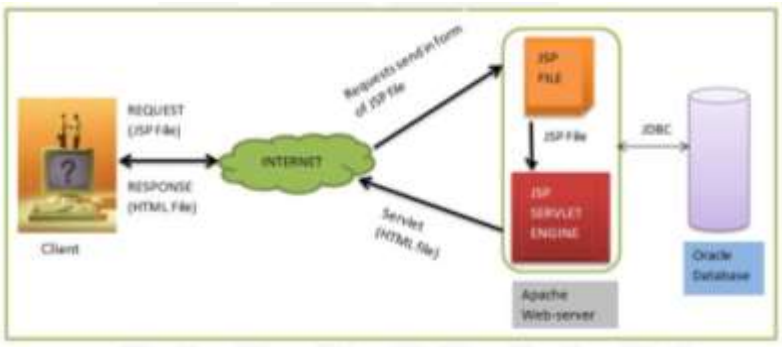
Advantages of JSP (JSP के फायदे):

- **Easy to learn** – JSP को सीखना बहुत ही आसान है, इसमें सामान्य HTML tags की तरह ही JSP tags का प्रयोग करते है। JSP के द्वारा आसानी से web applications को create कर सकते है।
- **Easy to manage** – JSP की सहायता से logic part और designing part अलग हो जाते है। क्योंकि designing HTML और CSS की मदद से करते है और logic को java की सहायता से perform करते है। इसलिए JSP application को manage करना बिल्कुल आसान होता है।
-
- **Automatic page compilation** – JSP में बार-बार program को compile नहीं करना पड़ता है, JSP, automatic page compilation प्रदान करती है। इससे application को बार बार deploy करने की problem से बच जाते है।
- JSP के साथ सभी Java API उपलब्ध हो जाती है। उदाहरण के लिए JSP के साथ JDBC API उपलब्ध होती है जिससे application, database से communicate कर सकती है।
- **Reliable & Secure** – Java की support होने से, JSP दूसरी technologies की तुलना में reliable (विश्वसनीय) और secure (सुरक्षित) है।

Working of JSP (Java Server Pages)

Java Server Pages की working में 4 elements बहुत महत्वपूर्ण होते हैं:-

1. Web browser
2. Web server
3. JSP engine
4. Servlet engine



जब वेब ब्राउज़र, JSP page के लिए request करता है यह request सीधे web server के पास पहुँचती है।

वेब सर्वर .jsp extension के माध्यम से identify करता है कि ये एक JSP page के लिए request है जो कि वेब सर्वर पर स्टोर है।

इस page को identify करके web server इसे आगे कि processing के लिए JSP engine को pass कर देता है। JSP engine वेब सर्वर पर installed होता है। Apache web server के साथ JSP engine पहले से ही enable होता है।

JSP engine इस JSP page को servlet में convert कर देता है। इसके बाद इस servlet को servlet engine को pass किया जाता है। Servlet engine इस servlet को process करता है और output के रूप में HTML page जनरेट करता है। यह HTML page वेब सर्वर को pass कर दिया जाता है। आखिर में web server इस HTML page को web browser को भेज देता है।

JSP vs Servlet

JSP(java server page)	Servlet
jsp page, servlets से slow होते है क्योकि इन्हे compile होकर servlet में convert होने में time लग जाता है|	Servlets, jsp से fast होते है क्योकि ये पहले से compiled होते है और directly execute होते है|
Jsp, automatic page compilation को provide करती है जिससे application को बार बार deploy करने की जरुरत नहीं होती है jsp में किये गए changes live होते है|	जब भी servlet program में कोई change करते है तो बार बार उसे compile करके run करना होता है
jsp सिर्फ html का requests ही handle करता है	servlet सभी प्रकार के protocol के requests को handle करते है

jsp में coding करना आसान होता है.	servlet programs बहुत ही lengthy होते है और उनकी coding भी बहुत difficult होती है.
jsp में एक html script में java प्रोग्राम को add करते है.	servlets में java प्रोग्राम में html के script add करते है.
jsp logic और design separate हो जाती है.	servlets में logic और design combined होते है.

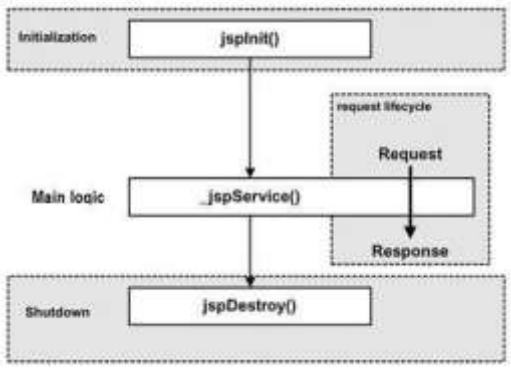
JSP Life cycle

JSP life cycle, jsp web page के creation (निर्माण) से उसके destruction(समाप्ति) तक का process है.

JSP life cycle के 4 phase है:

- Compilation
- Initialization
- Execution
- Cleanup

JSP life cycle के 4 phase, Servlet Life cycle के समान है, इन्हें निम्नानुसार वर्णन किया गया है:



1) JSP Compilation

जब कोई browser, JSP के लिए आग्रह करता है, तब jsp engine सबसे पहले यह check करता है कि कहीं page को compile करने की जरुरत न पड़ जाए. यदि page कभी भी compile नहीं हुआ है या यदि compile होने के बाद JSP, modify हुआ है, तब jsp engine, उस page को compile करेगा.

Compilation process में 3 step होते हैं:

- Parsing the JSP.
- Turning the JSP into a servlet.
- Compiling the servlet.

2) JSP Initialization

जब container JSP को load करता है, तब किसी request को service करने से पहले **jspinit()** method को access करता है. यदि JSP-specific initialization perform करना है तब **jspinit()** method को override() करें.

```
public void jspInit(){
  // Initialization code...
}
```

3) JSP Execution

JSP life cycle का यह phase, request के साथ सभी interaction को jsp के destroy (समाप्त) होने तक represent करता है. जब browser, jsp page को request करता है, और page, load और initialize हो गया हो तब, JSP engine, **_jspService()** method को invoke करता है. यह **_jspService()** method, **HttpServletRequest** और **HttpServletResponse** को parameter के रूप में निम्न प्रकार से लेता है:

```
void _jspService(HttpServletRequest request, HttpServletResponse response) {
  // Service handling code...
}
```

JSP का **_jspService()** method को request के आधार पर invoke किया जाता है. यही उस request के लिए response generate करने के लिए जिम्मेदार होते हैं और यही method, सभी 7 http methods जैसे GET, POST, DELETE etc. के लिए भी response generate करने के लिए जिम्मेदार हैं.

4) JSP CleanUp

यह JSP life cycle का destruction(समाप्ति) phase है, जब container, jsp को use करने से हटा देता है.

jspDestroy() method JSP का destroy method है. जब database connection को release करना हो या open file close करने, जैसे clean up perform करना हो तब, jspDestroy() method को Override करना होता है.

```
public void jspDestroy() {
  // Your cleanup code goes here.
}
```

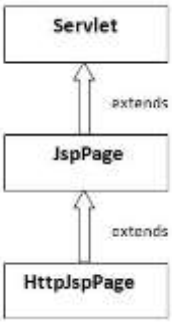
API

Application Programming Interface (API) एक विशिष्ट प्रकार का Code होता है। इसका उपयोग कर किसी दूसरे ऑपरेटिंग सिस्टम, एप्लीकेशन अथवा सर्विस के डाटा या फंक्शन्स को आसानी से एक्सेस किया जाता है। API Code बहुत सारे फंक्शन्स, कमांड्स, ऑब्जेक्ट्स एवं प्रोटोकॉल्स का कलेक्शन होता है। इसका यूज करके प्रोग्रामर्स सॉफ्टवेयर डवलप करते है और एक्सट्रनल सिस्टम से इंटरैक्ट करने के लिए मॉडल डिजाइन करते है।

API किसी सॉफ्टवेयर या एप्लीकेशन के उद्योग मानको या नियम व शर्तें के आधार पर बनाया जा सकता है ताकि विभिन्न घटकों में अंतर सुनिश्चित किया जा सके। यह पूरी तरह से कस्टम हो सकता है। API सिस्टम के Behaviour का वर्णन करती है जबकि लाइब्रेरी वास्तव में उस Behaviour को लागू करती हैं। एक सिंगल एपीआई में कई लाइब्रेरी हो सकती हैं क्योंकि इसमें कई प्रकार अलग-अलग Implementation हो सकते हैं। हालांकि कभी-कभी एक एपीआई को एक सॉफ्टवेयर फ्रेमवर्क से भी जोड़ा जा सकता है। एक फ्रेमवर्क कई लाइब्रेरी पर आधारित है जो विभिन्न एपीआई को Implemented करता है जिनका Behaviour फ्रेमवर्क में बनाया गया है।

JSP के लिए 2 API use होते हैं:

1. javax.servlet.jsp
2. javax.servlet.jsp.tagext



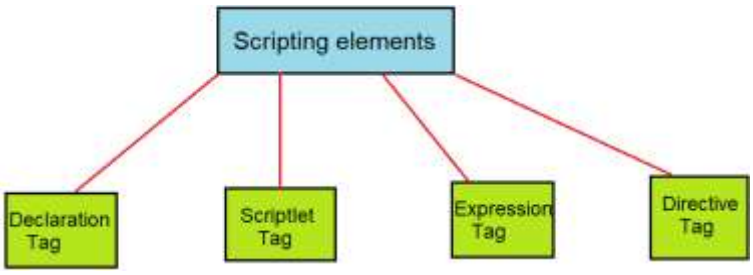
IDE

IDE या Integrated Development Environment, programming के विभिन्न प्रक्रिया को एक ही application/software में समेकित कर देता है, इससे programmer को programming करने में आसानी हो जाती है. software बनाने के ज्यादातर common activity जैसे editing code, building executables और debugging एक ही application में combined होने से programmer की productivity बढ़ जाती है. इसमें syntax highlighting, autocorrect colour code के कारण work करना आसान और fast हो जाता है. java server page(jsp) के लिए editor के रूप में ज्यादातर notepad, notepad++ का use किया जाता है, लेकिन इनमें debugging, autocorrect जैसे features नहीं होते, JSP में programming के लिए NetBeans, eclipes IDE का use किया जाता हैं.

Scripting Element

JSP, HTML script में java code insert करने के लिए tags provide करती है जिन्हें scripting elements कहा जाता है। इन scripting elements की मदद से अलग अलग purposes के लिए HTML script में java code लिखते है। Scripting elements में लिखा गया code, JSP engine द्वारा process किया जाता है। बाकी remaining code को HTML की तरह process किया जाता है। JSP 4 scripting elements provide करती है।

1. **Declaration tag**
2. **Scriptlet tag**
3. **Expression tag**
4. **Directive tag**



Scripting Element	Example
Comment	<%-- comment --%>
Directive	<%@ directive %>
Declaration	<%! declarations %>
Scriptlet	<% scriplets %>
Expression	<%= expression %>

1) JSP Declaration Tag

Declaration tag में ऐसे java code लिखे जाते हैं जिसे HTML script में बार बार use करते हैं जैसे variable या method declare करना। बाद में variable या method को script में कँही भी use कर सकते है।

इस tag का code, translate होने पर JSP life cycle के _jspService() method के बाहर होता है। इस tag में define किया गया java code execute नहीं होता है और कोई result भी produce नहीं करता है। इस code से output प्राप्त करने के लिए इन्हें expression tag में use किया जाता है.

Syntax:

<%! java code here %>

JSP declaration tag define करने के लिए < के बाद % और ! symbols use करते है। इसे close करने के लिए सिर्फ % और > symbol यूज़ किया जाता है। <% और %> symbols सभी scripting elements में common होते है। Declaration tag के syntax में ! symbol इसे और tags के बीच uniquely identify करने के लिए use होता है।

Example:

```
<%!
void MyFunction()
{
   out.println("Hello World!");
}

%>
```

ऊपर दिए गए उदाहरण में MyFunction नाम से एक function create किया गया है। ये function execute होने पर Hello World string print करता है। लेकिन declaration tag के द्वारा कोई output display नहीं कर सकते है इसलिए इस function को expression tag में call किया जायेगा।

2) JSP Scriptlet Tag

इस tag में बहुत अधिक java code लिखा जाता है। इस tag में लिखा हुआ java code translate होने पर _jspService() method के अंदर होता है। इस tag में execute किये गए code के output को out.println() method के द्वारा display करते है।

syntax:

<% java code here %>

scriptlet tag को < और % symbol के साथ start किया जाता है। इसके बाद java code लिखा जाता है। इस tag को % और > symbol से close करते है।

example:

```
<%
out.println("Hello World! ");
%>
```

example में out.println() method के माध्यम से Hello World string को print किया गया है।

3) JSP Expression Tag

Expression tag के माध्यम से java expressions को compute करके उनका result instantly show कर सकते है। इस tag में दिए गए java code का result show नहीं करना पड़ता है। ये tag एक तरह out.println statement ही होता है जिसमे वह evaluate होकर print हो जाता है। जैसे की (2 + 2) एक expression है।

Syntax:

<%= java code here %>

Expression tag को start करने के लिए क्रमशः < , % और = symbols यूज़ करते है। इस tag को close करने के लिए % और > symbol को use किया जाता है।

<%= MyFunction() %>

expression tag output show करता है। इस example में MyFunction() को execute करके उसका output show किया जायेगा।

JSP Directive Tag

Directive tag के द्वारा JSP engine को special instructions दिए जाते है। jsp directive, message है जो web container को यह बताता है कि JSP page को कैसे सम्बंधित servlet में translate किया जाये.

JSP 3 प्रकार के directive tags प्रदान करती है।

i. **page directive-** ये tag page properties जैसे language और sessions आदि define करने के लिए use किया जाता है।
ii. **include directive-** ये tag एक file को include करने के लिए use किया जाता है। इस tag के माध्यम से आप कोई भी java file HTML page में include कर सकते है।
iii. **taglib directive-** ये tag page में use की गयी tag library को declare करने के लिए use किया जाता है।

Syntax of JSP Directive:

```
<%@ directive attribute="value" %>
```

i. Page Directive –

Page directive ऐसे attributes define करता है जो पुरे JSP page पर apply होते हैं.
Syntax of JSP page directive
<%@ page attribute="value" %>

Page directive के attributes निम्न है:

1. import
2. contentType
3. extends
4. info
5. buffer
6. language
7. isELIgnored
8. isThreadSafe
9. autoFlush
10. session
11. pageEncoding
12. errorPage
13. isErrorPage

इनमें से कुछ attributes की जानकारी दी हुई है,

1. **import:** import attribute का use class, interface या किसी package के सभी member को import करने के लिए किया जाता है. यह java class या interface के import keyword की तरह है:
   example:
   <html>
   <body>

```
<%@ page import="java.util.Date" %>
Today is: <%= new Date() %>
</body>
</html>
```

2. **contentType**:

   contentType attribute, HTTP response के MIME(Multipurpose Internet Mail Extension) type को परिभाषित करता है. default value है: “text/html;charset=ISO-8859-1”

   Example:

```
<html>
<body>
<%@ page contentType=application/msword %>
Today is: <%= new java.util.Date() %>
</body>
</html>
```

3. **info:**

   यह attribute, JSP page के information को set करता है जिसे servlet interface के getServletInfo() method का use करके retrieve किया जाता है.

example:

```
<html>
<body>
<%@ page info="written for institute" %>
Today is: <%= new java.util.Date() %>
</body>
</html>
```

Web container, Servlet के result में getServleltInfo() method create करेगा.

```
public String getServletInfo() {
  return "composed by Sonoo Jaiswal";
}
```

4. buffer:

   buffer attribute, JSP page द्वारा generate किये गए output को handle करने के लिए kilobyte में buffer size को set करता है. buffer का default size 8KB है.

   Example:

```
<html>
<body>

<%@ page buffer="16kb" %>
Today is: <%= new java.util.Date() %>
```

```
</body>
</html>
```

5. **language :**

language attribute, JSP page में use किये गए scripting language को specify करता है. default value, “java” है.

```
<%@ page language="java" %>
```

6. **errorPage:**

errorPage attribute का use करके error page को define किया जाता है, यदि current page में exception आ जाये, तब इसे errorpage में redirect कर दिया जायेगा.

example:

```
//index.jsp
<html>
<body>
<%@ page errorPage="myerrorpage.jsp" %>
 <%= 100/0 %>
</body>
</html>
```

ii. ***include directive-***

include directive का use किसी file जैसे jsp file, html file या text file के content को include करने के लिए होता है. यह include directive, page translation के समय included resource के original content को include करता है. इसका advantage है code reusability.

Syntax:

```
<%@ include file="resourceName" %>
```

Example:

इस example में header.html file के content को include किया जा रहा है. इसके लिए header.html file create करना होगा.

```
<html>
<body>
<%@ include file="header.html" %>
Today is: <%= java.util.Calendar.getInstance().getTime() %>
 </body>
 </html>
```

iii. ***taglib directive:***

JSP taglib directive का use करके tag library को define किया जाता है, जो बहुत सारे tags को define करता है. TLD(Tag Library Descripter) file का use करके tags को define किया जाता है.

The JavaServer Pages API HTML tag, XML tag की तरह custom JSP tag define करने के लिए allow करता है. और tag library, user-defined tags का Set है जो custom behaviour को implement करता है.

Syntax:
<%@ taglib uri="uriofthetaglibrary" prefix="prefixoftaglibrary" %>
example:
इस example में, currentDate के नाम से tag बना है, इस tag को use करने के लिए taglib directive को specify करना पड़ेगा ताकि container, tag के बारे में information प्राप्त कर पायेगा.

```
<html>
<body>
<%@ taglib uri="http://www.xyz.com/tags" prefix="mytag" %>
<mytag:currentDate/>
</body>
</html>
```

JSP Implicit Object

JSP implicit objects, java objects ही है जिन्हें web container के द्वारा create किया गया है तथा इनका प्रयोग सभी JSP pages में किया जा सकता है. इनको translation phase (JSP से servlet में translation) में create किया जाता है.

इन्हें स्पष्ट रूप से declare तथा initialize किये बिना ही सीधे call किया जा सकता है. इन्हें pre-defined variables भी कहते है.

JSP में ढेर सारे implicit object है जिनमें से कुछ निम्न है:

- out
- request
- response
- config
- pageContext

○ out:

out object, javax.servlet.jsp.JspWriter का instance है, जो user को java servlet output streem को access करने देता है. client(browser) को भेजे जाने वाले output को इस out object के द्वारा आगे भेजा जाता है. आसान शब्दों में, JSP out implicit object का use करके client के लिए content write किया जाता है.
ये output, server से client की ओर आता है और client computer में display होता है या store होता है.
इसमें अनेक methods use किये जाते हैं:

1) void print()

यह method, इसे pass किये गए value को write करता है. output को screen (client browser) पर दिखाने के लिए इस method का use किया जाता है. यदि "I am a student" message को display करना हो तो निम्नलिखित statement लिखेंगे.

```
out.print("I am a student");
```

2) **void println()**

यह method, print() method जैसा है. print() और println() दोनों में सिर्फ एक difference यह है, कि println() method अंत में new line character को add कर देता है, जिससे print होने वाला output next line में दिखाता है.

example:

जब print() use करेंगे तब:

```
out.print("hi");
out.print(" ");
out.print("hello");
```

output: hi hello

जब println() use करेंगे तब:

```
out.println("hi");
out.println("hello");
```

output:

hi
hello

3) **void newline():**

यह method, output में नया line add कर देता है.

Example:

```
out.print("first sentence");
out.newLine();
out.print(" second sentence");
```

Output:

first sentence
second sentence

4) **void clear()**:

यह method output buffer को clear कर देता है यदि buffer का content, client में write न हुआ हो तब भी

syntax:

```
out.clear();
```

5) **void clearBuffer()**: यह method, clear() method की तरह है, सिर्फ यह difference है कि जब पहले से ही flushed buffer को out.clear() को call किया जाता है तब यह exception throw करता है.जबकि out.clearBuffer() कोई exception throw नहीं करता भले ही buffer पहले से ही flushed हो गया हो.

6) **void flush()** : यह method भी clear() method की तरह buffer को clear करता है किन्तु यह clear(flush) करने से पहले content को output के रूप में write कर देता है अर्थात buffer में जो भी content होता है, वो buffer के clear होने से पहले client screen पर display हो जाता है.

7) **boolean isAutoFlush()** :

इसका use यह check करने के लिए होता है कि buffer automatically flush करता है या नहीं. यदि buffer flushed है तब true return करेगा, नहीं तो false.

8) **int getBufferSize()**: यह method, output buffer के size को bytes में return करता है.

9) **int getRemaining()**: यह buffer overflow condition से पहले बचे हुए byte की संख्या को return करता है.

नीचे दिए हुए example में out object के print() और println() methods का use करके output screen पर message print करेंगे.

file name: **index.jsp**

```
<HTML>

<HEAD>

<TITLE> OUT IMPLICIT OBJECT EXAMPLE </TITLE>

</HEAD>

<BODY>

<%

out.print( "print statement " );

out.println( "println" );

out.print("Another print statement");

%>

</BODY>

</HTML>
```

Output:

print statement println

Another print statement

- ## request:

  request implicit object का मुख्य उद्देश्य है jsp page पर data प्राप्त करना जिसे पहले के jsp page में user द्वारा enter किया गया है. JSP में जब भी login और signup format को use करते हैं तब details को भरने के लिए user को prompt करते हैं, इस object का use करके उन enter किये हुए details को दुसरे page(action page) में validation और अन्य उद्देश्य के लिए भेजता है.

  request object के कुछ methods:

  1) **getParameter() :**

     इस method का use करके request के parameter के value को प्राप्त किया जाता है. example लें तो login page पर user यदि user-id और password enter करता है और verification के बाद login page, user information page को redirect कर देता है, इसके बाद request.getParameter का use करके user-id और password के value को प्राप्त करते हैं जिसे user ने login page में input किया था. example:

String Uid= request.getParameter("user-id");
String Pass= request.getParameter("password");

2) **getParameterNames() :** यह request से सम्बंधित सभी parameter name के enumeration को return करता है.

Enumeration e= request.getParameterNames();

3) **getAttribute(String name):** इसका use attribute value को प्राप्त करने के लिए किया जाता है. request.getAttribute("admin") statement, admin attribute के value को प्रदान करेगा.
4) **getCookies()** – यह client से cookie object के array को return करता है.इस method का use मुख्य रूप से JSP में cookies में काम करने के लिए होता है.
5) **getQueryString() –** इसका use, JSP page URL से सम्बंधित query string को प्राप्त करने के लिए किया जाता है. यह URL में Question mark (?) के बाद का string है.
6) **getParameterValues(String name) –** यह parameter values का array return करता है.
String[] allpasswords = request.getParameterValues("password");

इनके अलावा और भी methods है: getAttribute(String Name), getAttributeNames(), setAttribute(String, Object), removeAttribute(String), getHeader(String Name), getHeaderNames(), getRequestURL(), getMethod().

Example: इस example में index.html page में user से input प्राप्त कर रहे हैं, और इन्हीं input data को request implicit object की सहायता से userinfo.jsp page में display कर रहे हैं.

index.jsp

```
<html>
<head>
<title>Enter UserName and Password</title>
</head>
<body>
<form action="userinfo.jsp">
Enter User Name: <input type="text" name="uname" /> <br><br>
Enter Password: <input type="text" name="pass" /> <br><br>
<input type="submit" value="Submit Details"/>
</form>
</body>
</html>
```

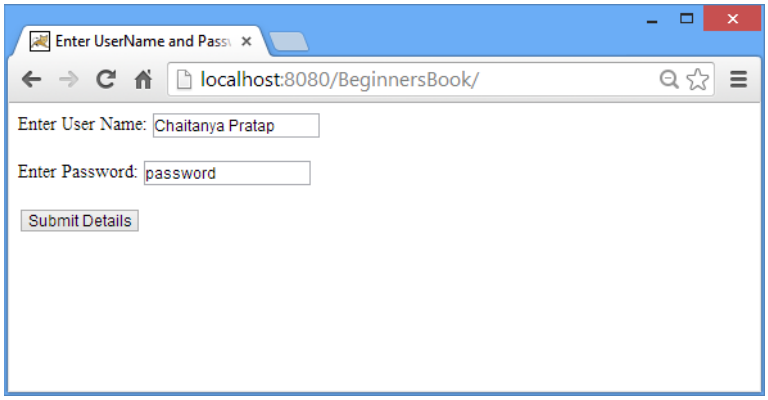
userinfo.jsp

```
<%@ page import = " java.util.* " %>
<html>
<body>
<%
String username=request.getParameter("uname");
String password=request.getParameter("pass");
out.print("Name: "+username+" Password: "+password);
```

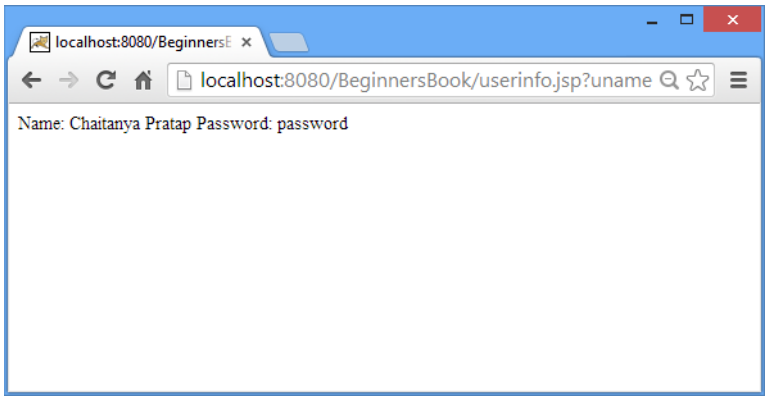
%>
</body>
</html>

output:

index.jsp



userinfo.jsp



○ response:

यह javax.servlet.http.HttpServletRequest का instance है और मुख्य रूप से response implicit object का use, client request को process करने के बाद browser को भेजे जाने वाले response को modify करने के लिए होता है. इसमें निम्न methods use होते हैं:

1. void setContentType(String type)
2. void sendRedirect(String address)
3. void addHeader(String name, String value)
4. void setHeader(String name, String value)
5. boolean containsHeader(String name)
6. void addCookie(Cookie value)
7. void sendError(int status_code, String message)
8. boolean isCommitted()
9. void setStatus(int statuscode)

1. **void setContentType(String type)** – यह method, browser को MIME type और character encoding setting के अनुसार response data को बताता है. इस method के द्वारा सेट किये गए यह information, response को interpret करने में help करता है.
   Example –

   response.setContentType("text/html");

   response.setContentType("image/gif");

   response.setContentType("image/png");

   response.setContentType("application/pdf");

2. **void sendRedirect(String address)** – यह new jsp page में control को redirect करता है.
   इसके लिए current page में इस method को use करके argument के रूप में जिस new jsp page में redirect होना है उसका URL/address को pass करना है.

   response.sendRedirect("www.8080/pages/xyz.jsp");

3. **void addHeader(String name, String value)** – addHeader method, response में header को add कर देता है. जिसमें header का नाम और इसकी value होती है. For example – The below statement will include a header “Site” in the response with value “BeginnersBook.com”.

   response.addHeader("Site", "www.abc.com");

4. **void setHeader(String name, String value)** – यह header value को set कर देता है. यह method, header के current value को new value से override कर देता है.

   response.setHeader("Site", "BB.com");

5. **void addCookie(Cookie cookie)** – यह method response में cookie add कर देता है. cookies name/value के रूप में small information होते हैं.
   response.addCookie(Cookie Author);
   response.addCookie(Cookie Siteinfo);
6. **boolean isCommitted()** – यह method check करता है कि Http response, client को भेज दिया गया है या नहीं. यदि हाँ, तब यह true return करेगा नहीं तो false.

```
<% if(response.isCommited())
{
 <%--do something --%>
}else
 {
<%--do something else --%>
} %>
```

- ## pageContext:

यह object, javax.servlet.jsp.PageContext का instance है. JSP pageContext implicit object का use करके get attribute, set attribute या remove attribute operation दिए scope के अन्दर किया जा सकता है.

JSP Page – Scope: PAGE_CONTEXT

HTTP Request – Scope: REQUEST_CONTEXT
HTTP Session – Scope: SESSION_CONTEXT
Application Level – Scope: APPLICATION_CONTEXT

इसमें use होने वाले methods:

1. **Object findAttribute (String AttributeName):** यह method, सभी 4 level page, request, session और application के क्रम में specific attribute के लिए search करता है. यदि किसी भी level में attribute न मिले तब यह NULL return करता है.
2. **Object getAttribute (String AttributeName, int Scope)**: यह method, findAttribute() की तरह ही है. अंतर यही है कि यह सिर्फ specified level या scope में ही attribute को search करता है.

Object obj = pageContext.getAttribute("ABC", PageContext.SESSION_CONTEXT);
इसी प्रकार दुसरे scope के लिए इस method को use किया जा सकता है.
Object obj = pageContext.getAttribute("ABC", PageContext. REQUEST_CONTEXT);
Object obj = pageContext.getAttribute("ABC", PageContext. PAGE_CONTEXT);
Object obj = pageContext.getAttribute("ABC", PageContext. APPLICATION_CONTEXT);

3. **void removeAttribute(String AttributeName, int Scope):** इस method का use करके दिए scope से किसी attribute को remove किया जाता है.
   pageContext.removeAttribute("MyAttr", PageContext. PAGE_CONTEXT);
   यह JSP statement, page scope से "MyAttr" attribute को हटा देगा.
4. **void setAttribute(String AttributeName, Object AttributeValue, int Scope):** यह method किसी दिए scope में attribute को write कर देता है.
   pageContext.setAttribute("mydata", "This is my data", PageContext. APPLICATION_CONTEXT);

   यह statement, application scope में "mydata" attribute स्टोर करेगा जिसकी value "This is my data" होगी.

- ## config:

JSP config implicit object, javax.servlet.ServletConfig का instance है. इसका use किसी perticular JSP page के लिए configuration information प्राप्त करना है. application implicit object का use करके पुरे application के लिए initialization parameter प्राप्त कर सकते हैं.फिर भी config का use करके individual servlet mapping के initialization parameter को प्राप्त कर सकते हैं.

config implicit method के methods:

1. **String getInitParameter(String name):** यह दिए parameter name के लिए initialization parameter के value को return करता है.
   <web-app>
   …
   <context-param>
   <param-name>parameter1</param-name>

<param-value>ValueOfParameter1</param-value>
</context-param>
</web-app>
यह web.xml file है.

String s=application.getInitParameter("parameter1");

2. **Enumeration getInitParameterNames():** initialization parameter के enumeration को Return करता है.
3. **ServletContext getServletContext():** यह method Servlet context को reference return करता है.
4. **String getServletName():** यह method, servlet के name को return करता है, जिसे web.xml file में <servlet-name> tag के अन्दर define किया जाता है.

Unit-5

Development in JSP

Action Element:

JSP में action elements विशेष tags होते है। इन tags के माध्यम से JSP, special features provide करती है। उदाहरण के लिए useBean action element को use करके java bean class को JSP page में use कर सकते है।

ये tags JSP engine द्वारा specially अलग तरह से process किये जाते है। जब इन tags को use किया जाता है तब code तो कम लिखना पड़ता ही है साथ ही task भी ज्यादा आसानी से perform कर पाते है। ये tags, common functionality provide करते है जिसकी जरुरत लगभग सभी web applications में होती है।

JSP, निचे दिए गए action tags provide करती है। सभी tags को jsp: keyword से start करते है।

jsp:include

jsp:forward

jsp:param

jsp:useBean

jsp:getProperty

jsp:setProperty

jsp:plugin

jsp:body

jsp:text

jsp:element

attribute

इन सभी action tags के बारे में detail से बताया जा रहा है।

jsp:include

यदि किसी दूसरे JSP page को main page में include करना चाहते है तो इसके लिए include action element use कर सकते है। हालाँकि ऐसा JSP include directive के द्वारा भी किया जा सकता है लेकिन include action element उससे advanced functionality provide करता है। इन दोनों tags में क्या main differences है।

Include Directive	Include Action
जब include directive use करते है तो file translation के समय ही main page में include हो जाती है।	जब include action use करते है तो file run time पर main page में include होती है।
जब include directive use करते है तो include की गयी file का content main file में copy हो जाता है।	जब include action use करते है तो file को refer (point) किया जाता है। ये किसी pointer की तरह होता है।
Include directive को static content के संदर्भ में प्रयोग किया जाता है।	Include action को dynamic content के संदर्भ में प्रयोग किया जाता है।
Include directive के द्वारा include की जाने वाली file को parameters नहीं pass कर सकते है।	Include action के द्वारा include की गयी file को parameters pass किये जा सकते है।

JSP action element का basic syntax:

```
<jsp:include page="URL" flush="true" />
```

या निचे दिया गया syntax भी use कर सकते है।

```
<jsp:include page="URL" flush="true">
</jsp:include>
```

Include action tag के 2 attributes होते है।

1. **page** – इस attribute के द्वारा उस page का URL define करते है जिसे include करना चाहते है।
2. **flush** – इस attribute की value true या false में दी जाती है। ये attribute बताता है की add की गई file buffer flush की गयी है या नहीं की गयी है। ये जरुरी नहीं है की इस attribute को include ही करे only flush attribute के साथ भी इस tag को use कर सकते है।

इसे निचे एक उदाहरण के माध्यम से समझाया जा रहा है।

```
<jsp:include page="sales.jsp" flush="true" />
```

jsp:forward

सभी requests किसी web application के main page को भेजी जाती है। लेकिन ये जरुरी नहीं है की सभी requests को main page में ही handle करें। यदि current page के लिए की गयी request को किसी दूसरे JSP page को forward करना चाहते है तो इसके लिए JSP forward action tag use कर सकते है।

ध्यान रखने योग्य बात ये है की के current page का content pass नहीं होता है केवल request pass होती है। इस tag में

page attribute define करते है जो की उस page का URL define करता है जिसे request forward की जानी है। JSP forward tag का general syntax निचे दिया जा रहा है।

```
<jsp:forward page="URL“>
```

चाहे तो निचे दिया गया syntax भी use कर सकते है।

```
<jsp:forward page="URL">
</jsp:forward>
```

इसे निचे उदाहरण के माध्यम से समझाया जा रहा है।

```
<jsp:forward page="sales.jsp" />
```

jsp:param

यदि include किये गए JSP page को या forward किये गए JSP page को parameters pass करना चाहते है तो इसके लिए JSP param action tag use कर सकते है। ये JSP include और forward action tags के sub tag के रूप में use किया जाता है।

JSP param tag में parameters name और value attributes के द्वारा define किये जाते है। इन parameters को दूसरे page में getParameter() method द्वारा access करते है। इस tag का general syntax निचे दिया जा रहा है।

```
<jsp:param name="Name" value-"Value-Of-Parameter” />
```

इसे निचे उदाहरण के माध्यम से समझाया जा रहा है।

```
<jsp:forward page="sales.jsp"> <jsp:param name="UserName" value="YrName" /> </jsp:forward>
```

ऊपर दिए गए उदाहरण में request sales.jsp page को forward की जा रही है। साथ ही एक UserName नाम से एक parameter भी pass किया गया है जिसकी value YrName है। इस parameter की value को sales.jsp page में इस प्रकार access किया जायेगा।

```
<%= request.getParameter("UserName");
```

यँहा पर expression directive में getParameter() method को call किया गया है। जैसा की को expression directive में लिखा गया java code directly execute होता है। इसलिए ये statement execute होने पर इस parameter की value द्वारा replace कर दिया जायेगा जो की YrName है।

jsp:useBean

JSP useBean action tag को JSP page में Java bean classes use करने की capability provide करता है। इस tag के साथ 2 tags <jsp:getProperty> और <jsp:setProperty> होते है। इन tags की मदद से Java bean class में properties को set और access करते है। JSP useBean tag का general syntax निचे दिया जा रहा है।

```
<jsp:useBean
id="UniqueName"                scope="page|request|session|application"          class="package.class"
 type="package.class"              beanName="package.class"> </jsp:useBean>
```

JSP useBean action tag में 5 attributes define करते है। इनके बारे में निचे बताया जा रहा है।

1. **id** – इस attribute में java bean class के object का नाम देते है। इस नाम के द्वारा bean class को दिए गए object में uniquely identify किया जाता है।
2. **scope** – इस attribute के द्वारा include की गयी bean class का scope define करते है। इसे page, request, session और application values के रूप में define करते है।
3. **class** – इस attribute के द्वारा bean class को define करते है। Class को उसके package के साथ define किया जाता है।
4. **type** – इस attribute द्वारा define करते है की object किस type को होगा।
5. **beanName** – ये attribute bean class को instantiate() method द्वारा instantiate करता है।

जरुरी नहीं की इन सभी attributes को define करें। चाहे तो सिर्फ id और class attribute के साथ भी इस tag को use कर सकते है।

jsp:setProperty

इस tag द्वारा bean class में property की value set करते है। इस tag में 3 property define करते है।

1. **name** – ये bean class का नाम होता है जिसमे property set करने के लिए होती है।
2. **property** – ये उस property का नाम होता है जिसे set करना चाहते है।
3. **value** – ये वो value होती है जो property के लिए set करना चाहते है।

इसका syntax निचे दिया जा रहा है।

```
<jsp:setProperty name="Bean_Class-Name" property="propertyName" value="val" />
```

jsp:getProperty

इस action tag के द्वारा bean class से किसी भी property की value access कर सकते है। इस tag के 2 attributes होते है।

1. **name** – ये उस bean class का नाम होता है जिसकी property को access करना चाहते है।
2. **property** – ये उस property का नाम होता है जिसकी value retrieve करना चाहते है।

इसका syntax निचे दिया जा रहा है।

```
<jsp:getProperty name="Name_Of_Bean_Class" property="name-of-property" />
```

JSP में java bean class के use को एक complete उदाहरण द्वारा निचे समझाया जा रहा है।

Java Bean Class	JSP page
package myPackage; public class BeanDemo { private int num; public void setValue(int n) { this.num=n; } public int getValue() { return num; }	<html> <head> <title>JSP Bean Demo</title> </head> <body> <jsp:useBean id="MyBean" class="MyPackage.BeanDemo"> <jsp:setProperty name="BeanDemo" property="num" value="5" /> <jsp:getProperty name="BeanDemo" property="num" /> </body> </html>

jsp:plugin

JSP का plugin action tag JSP page में applet को insert करने के लिए use किया जाता है। Applet को execute करने के लिए ये tag client machine पर plugin download करता है। इस tag के द्वारा bean भी JSP page में insert कर सकते है। ये tag आवश्यकता के अनुसार object और embed tag को use करता है। इस action tag का general syntax निचे दिया जा रहा है।

```
<jsp:plugin type="bean or applet" code="name of java class file" codebase="directory name fo java class file">

</jsp:plugin>
```

jsp:element

ये tag XML elements को dynamically define करने के लिए use किया जाता है। इस action tag के अंदर 3 और action tags define करते है।

1. <jsp:body>
2. <jsp:attribute>

इस tag का general syntax निचे दिया जा रहा है।

```
<jsp:element name="element-name"> attribute & body action tags </jsp:element>
```

jsp:body

ये tag XML file में body elements define करने के लिए use किया जाता है। इस tag <jsp:element> tag के अंदर define किया जाता है। इसका general syntax निचे दिया जा रहा है।

```
<jsp:body>              body here  </jsp:body>
```

jsp:attribute

इस action tag के द्वारा attribute और उसकी value define करते है। इसका general syntax निचे दिया जा रहा है।

```
<jsp:attribute name="attibute-name">          value of defined attribute </jsp:attribute>
```

Client request

request implicit object का मुख्य उद्देश्य है jsp page पर data प्राप्त करना जिसे पहले के jsp page में user द्वारा enter किया गया है. JSP में जब भी login और signup format को use करते हैं तब details को भरने के लिए user को prompt करते हैं, इस object का use करके उन enter किये हुए details को दुसरे page(action page) में validation और अन्य उद्देश्य के लिए भेजता है.

request object के कुछ methods:

1) **getParameter() :**
   इस method का use करके request के parameter के value को प्राप्त किया जाता है. example लें तो login page पर user यदि user-id और password enter करता है और verification के बाद login page, user information page को redirect कर देता है, इसके बाद request.getParameter का use करके user-id और password के value को प्राप्त करते हैं जिसे user ने login page में input किया था.
   example:
   String Uid= request.getParameter("user-id");
   String Pass= request.getParameter("password");
2) **getParameterNames() :** यह request से सम्बंधित सभी parameter name के enumeration को return करता है.

   Enumeration e= request.getParameterNames();

3) **getAttribute(String name):** इसका use attribute value को प्राप्त करने के लिए किया जाता है. request.getAttribute("admin") statement, admin attribute के value को प्रदान करेगा.
4) **getCookies()** – यह client से cookie object के array को return करता है.इस method का use मुख्य रूप से JSP में cookies में काम करने के लिए होता है.
5) **getQueryString() –** इसका use, JSP page URL से सम्बंधित query string को प्राप्त करने के लिए किया जाता है. यह URL में Question mark (?) के बाद का string है.
6) **getParameterValues(String name) –** यह parameter values का array return करता है.
   String[] allpasswords = request.getParameterValues("password");

इनके अलावा और भी methods है: getAttribute(String Name), getAttributeNames(), setAttribute(String, Object), removeAttribute(String), getHeader(String Name), getHeaderNames(), getRequestURL(), getMethod().

Example: इस example में index.html page में user से input प्राप्त कर रहे हैं, और इन्हीं input data को request implicit object की सहायता से userinfo.jsp page में display कर रहे हैं.

index.jsp

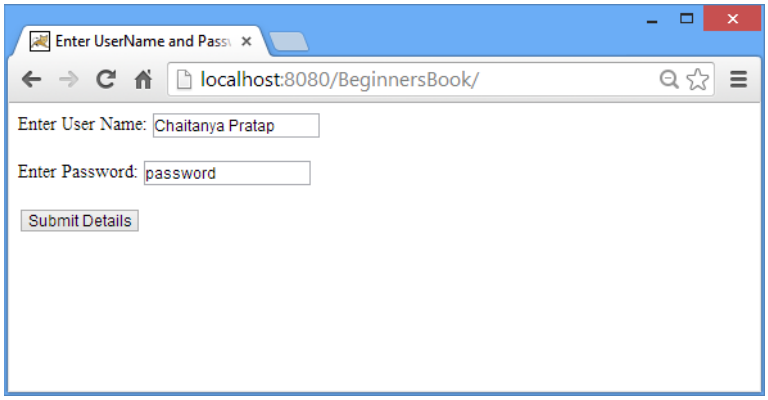
```
<html>
<head>
<title>Enter UserName and Password</title>
</head>
<body>
<form action="userinfo.jsp">
Enter User Name: <input type="text" name="uname" /> <br><br>
Enter Password: <input type="text" name="pass" /> <br><br>
<input type="submit" value="Submit Details"/>
</form>
</body>
</html>
```

userinfo.jsp

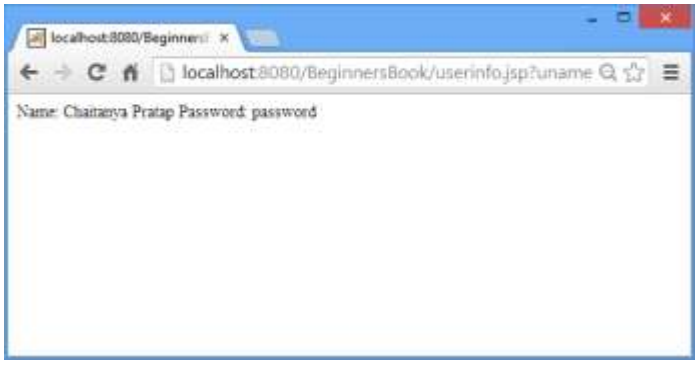
```
<%@ page import = " java.util.* " %>
<html>
<body>
<%
String username=request.getParameter("uname");
String password=request.getParameter("pass");
out.print("Name: "+username+" Password: "+password);
%>
</body>
</html>
```

output:

index.jsp



userinfo.jsp



Server response

यह javax.servlet.http.HttpServletRequest का instance है और मुख्य रूप से response implicit object का use, client request को process करने के बाद browser को भेजे जाने वाले response को modify करने के लिए होता है. इसमें निम्न methods use होते हैं:

1. void setContentType(String type)
2. void sendRedirect(String address)
3. void addHeader(String name, String value)
4. void setHeader(String name, String value)
5. boolean containsHeader(String name)
6. void addCookie(Cookie value)
7. void sendError(int status_code, String message)
8. boolean isCommitted()
9. void setStatus(int statuscode)

1. **void setContentType(String type)** – यह method, browser को MIME type और character encoding setting के अनुसार response data को बताता है. इस method के द्वारा सेट किये गए यह information, response को interpret करने में help करता है.
   Example –

   ```
   response.setContentType("text/html");

   response.setContentType("image/gif");

   response.setContentType("image/png");

   response.setContentType("application/pdf");
   ```

2. **void sendRedirect(String address)** – यह new jsp page में control को redirect करता है. इसके लिए current page में इस method को use करके argument के रूप में जिस new jsp page में redirect होना है उसका URL/address को pass करना है.

   ```
   response.sendRedirect("www.8080/pages/xyz.jsp");
   ```

3. **void addHeader(String name, String value)** – addHeader method, response में header को add कर देता है. जिसमें header का नाम और इसकी value होती है. For example – The below statement will include a header “Site” in the response with value “BeginnersBook.com”.

response.addHeader("Site", "www.abc.com");

4. **void setHeader(String name, String value)** – यह header value को set कर देता है. यह method, header के current value को new value से override कर देता है.

   response.setHeader("Site", "BB.com");

5. **void addCookie(Cookie cookie)** – यह method response में cookie add कर देता है. cookies name/value के रूप में small information होते हैं.
   response.addCookie(Cookie Author);
   response.addCookie(Cookie Siteinfo);
6. **boolean isCommitted()** – यह method check करता है कि Http response, client को भेज दिया गया है या नहीं. यदि हाँ, तब यह true return करेगा नहीं तो false.
   <% if(response.isCommited())
   {
   <%--do something --%>
   }else
   {
   <%--do something else --%>
   } %>

Form Processing

Forms सामान्य methods होते है जिनकी मदद से web pages के साथ interact कर सकते है.

कभी कभी information को ब्राउज़र से वेब सर्वर को भेजना होता है. तो ब्राउज़र information को वेब सर्वर को भेजने के लिए दो methods का प्रयोग करता है:- GET method तथा POST method.

JSP form processing methods:-

1:- GET method:-

GET method ब्राउज़र से वेब सर्वर को सूचना भेजने के लिए default मेथड होती है.

GET method जो है वह URL पेज के साथ यूजर की encoded information को एक साथ जोड़कर भेजती है. पेज url तथा encoded information को character '?' अलग करता है. नीचे देखिये-
https://ehindistudy.com/hello?key1=value1&key2=value2

जब से GET method में url के साथ plain text string जुड़ा है, GET method का प्रयोग sensitive information जैसे:- पासवर्ड तथा अन्य संवेदनशील सूचना को सर्वर को भेजने में नहीं करना चाहिए.

GETmethod में size का limitation भी है. केवल 1024 characters ही request में भेज सकते है.

information को request ऑब्जेक्ट के getQueryString() तथा getParameter() methods का प्रयोग करके एक्सेस किया जाता है.

2:-POST method:-

POST method सर्वर को इनफार्मेशन भेजने की सबसे ज्यादा विश्वसनीय मेथड है. यह method जो है वह GET method की तरह ही इनफार्मेशन को भेजती है, GET method में character '?' के बाद text string को भेजा जाता है जबकि POST method में इनफार्मेशन को एक अलग मैसेज की तरह भेजा जाता है.

यह ज्यादातर ऐसी सूचना को भेजने में प्रयोग किया जाता है जो sensitive होती है जैसे:- password इत्यादि.

jsp इस प्रकार की request को getParameter() तथा getInputStream() method का प्रयोग करके हैंडल करता है. getParameter() का प्रयोग सरल पैरामीटर्स को read करने के लिए तथा getInputStream() का प्रयोग बाइनरी डेटा स्ट्रीम को read करने के लिए किया जाता है.

JSP डेटा प्रोसेसिंग को निम्नलिखित methods की सहायता से हैंडल करता है:-

1:- getParameter():- इसका प्रयोग पैरामीटर से वैल्यू प्राप्त करने के लिए किया जाता है.

2:- getParameterValue():- इसका प्रयोग पैरामीटर्स की बहुत सारी values को return करने के लिए किया जाता है.

3:- getParameterNames():- इसका प्रयोग पैरामीटर के names को प्राप्त करने के लिए किया जाता है.

4:- getInputStream():– इसका प्रयोग क्लाइंट के द्वारा भेजे गये बाइनरी डेटा को read करने के लिए किया जाता है.

get method example:- नीचे इसका उदाहरण दिया गया है**:-**

jsp form processing

```
<html>
<head>
<title>example of GET method</title>
</head>
<body>
<h1>GET method</h1>
<ul>
<li><p><b>Name:</b>
<%= request.getParameter("name")%>
</p></li>
<li><p><b>address:</b>
<%= request.getParameter("city")%>
</p></li>
<li><p><b>college:</b>
<%= request.getParameter("prof")%>
</p></li>
</ul>
</body>
</html>
```

POST method example:- नीचे दिए गये उदाहरण में को form का प्रयोग करके post method को implement करना दिखाया गया है.

file name:-hello.html

```
<html>
<head>
<title>example of post method</title>
</head>
<body>
<h2>example of post method</h2>
<form action = "welcome.jsp" method = "POST">
Name: <input type = "text" name = "name"/>
address: <input type = "text"  name = "city"/>
college:  <input type = "text" name = "job">
<input type = "submit" value = "Submit">
</form>
</body>
</html>
```

file name:- simple.jsp

file name:- simple.jsp

```
<html>
<head>
<title>simple page of jsp by post method</title>
</head>
<body>
Name: <%= request.getParameter ("name") %>
address: <%= request.getParameter ("city") %>
college: <%= request.getParameter ("job") %>
</body>
</html>
```

JSP - Cookies Handling

Cookies, client computer में store होने वाले text files होते हैं और इन्हें विभिन्न information को track करने के लिए रखे जाते हैं. ये छोटे textual information होते हैं, जो client computer में store होते हैं. इसे server से client को और फिर client से server को अन्य request के साथ वापस भेजा जाता है. JSP भी HTTP cookies को servlet technology का use करके support करता है.

Cookies use करने के लाभ: 1) username और password जैसे information को याद रखना

2) प्राथमिकता को याद रखना

3) विज्ञापन (advertisement)

पुराने user को पहचानने के लिए 3 step शामिल होते हैं:

- Server script, browser को cookies का set भेजता है. जैसे name, age, ID आदि.
- Browser इन information को भविष्य में use करने के लिए local machine में store करता है.
- अगली बार, browser कोई request web server को भेजता है तब उस request के साथ cookies information को भी server को भेजा जाता है, और server इन information को user को पहचानने या अन्य उद्देश्य के लिए use करता है.

JSP script, request method ***request.getCookies()*** की सहायता से cookies को access कर सकता है. जो कि cookies object का Array return करता है. हालाँकि browser, user को यह option देता है की cookies को delete या disable कर सकता है.

Servlet Cookies Methods

Following table lists out the useful methods associated with the Cookie object which you can use while manipulating cookies in JSP −

S.No.	Method & Description
1	**public void setMaxAge(int expiry)** This method sets how much time (in seconds) should elapse before the cookie expires. If you don't set this, the cookie will last only for the current session.
2	**public int getMaxAge()** This method returns the maximum age of the cookie, specified in seconds, By default, **-1** indicating the cookie will persist until the browser shutdown.
3	**public String getName()** This method returns the name of the cookie. The name cannot be changed after the creation.
4	**public void setValue(String newValue)** This method sets the value associated with the cookie.
5	**public String getValue()** This method gets the value associated with the cookie.
6	**public void setComment(String purpose)** This method specifies a comment that describes a cookie's purpose. The comment is useful if the browser presents the cookie to the user.
7	**public String getComment()**

	This method returns the comment describing the purpose of this cookie, or null if the cookie has no comment.
8	**addCookie()** add cookie in response and it is sent to the browser.
9	**getCookie()** To read the cookie sent by client along with request.

Setting Cookies with JSP

JSP specification javax.servlet.http package में एक cookie class प्रदान करता है. यह cookie class का एक constructor है जिसमें 2 argument, cookie की name और value की आवश्यकता होती है.

Syntax:

```
Cookie(String name, String value);
```

Cookies setting के लिए 3 step शामिल है:

Step 1: एक Cookie object create करना

cookie name और cookie value देकर Cookie constructor को call किया जाता है. ये दोनों argument string होते हैं:

```
Cookie cookie = new Cookie("key","value");
```

Step 2: maximum age को set करना

setMaxAge() method का use करके cookie की validity अवधि को specify किया जाता है.

```
cookie.setMaxAge(60*60*24);     // set up a cookie for 24 hours.
```

Step 3: Cookie को HTTP response headers में भेजना

response.addCookie का use करें.

```
response.addCookie(cookie);
```

Example

Let us modify our Form Example to set the cookies for the first and the last name.

```
<%
  // Create cookies for first and last names.
  Cookie firstName = new Cookie("first_name", request.getParameter("first_name"));
  Cookie lastName = new Cookie("last_name", request.getParameter("last_name"));

  // Set expiry date after 24 Hrs for both the cookies.
  firstName.setMaxAge(60*60*24);
  lastName.setMaxAge(60*60*24);
```

```
   // Add both the cookies in the response header.
   response.addCookie( firstName );
   response.addCookie( lastName );
%>

<html>
   <head>
      <title>Setting Cookies</title>
   </head>

   <body>
      <center>
         <h1>Setting Cookies</h1>
      </center>
      <ul>
         <li><p><b>First Name:</b>
            <%= request.getParameter("first_name")%>
         </p></li>
         <li><p><b>Last  Name:</b>
            <%= request.getParameter("last_name")%>
         </p></li>
      </ul>

   </body>
</html>
```

Let us put the above code in **main.jsp** file and use it in the following HTML page −

```
<html>
   <body>

      <form action = "main.jsp" method = "GET">
         First Name: <input type = "text" name = "first_name">
         <br />
         Last Name: <input type = "text" name = "last_name" />
         <input type = "submit" value = "Submit" />
      </form>

   </body>
</html>
```

OUTPUT: **hello.jsp**

Main.jsp

Setting cookies

First Name: JOHN

Last Name: KUMAR

Reading Cookies with JSP

Cookies को read करने के लिए, javax.servlet.http.cookie object का array, HttpServletRequest के method getCookies() को call करके create करते हैं. इसके बाद, **getName()** and **getValue()** का use करके प्रत्येक cookie और सम्बंधित value को access करते हैं.

Example

```
<html>
  <head>
    <title>Reading Cookies</title>
  </head>

  <body>
    <center>
      <h1>Reading Cookies</h1>
    </center>
    <%
      Cookie cookie = null;
      Cookie[] cookies = null;

      // Get an array of Cookies associated with the this domain
      cookies = request.getCookies();

      if( cookies != null ) {
        out.println("<h2> Found Cookies Name and Value</h2>");

        for (int i = 0; i < cookies.length; i++) {
          cookie = cookies[i];
          out.print("Name : " + cookie.getName( ) + ",  ");
          out.print("Value: " + cookie.getValue( )+" <br/>");
        }
      } else {
        out.println("<h2>No cookies founds</h2>");
      }
    %>
  </body>

</html>
```

OUTPUT:

Found Cookies Name and Value

Name : first_name, Value: JOHN

Name : last_name, Value: KUMAR

Delete Cookies with JSP

Cookie को delete करने के लिए निम्न 3 step को follow करें:

1. सभी उपलब्ध cookie को read करें, और cookie object में store करें.
2. cookie को delete करने के लिए setMaxAge() method का use करके cookie के age को zero (0) करें.
3. response header में इस cookie को वापस add करें.

example:

```
<html>
  <head>
    <title>Reading Cookies</title>
  </head>

  <body>
    <center>
      <h1>Reading Cookies</h1>
    </center>
    <%
      Cookie cookie = null;
      Cookie[] cookies = null;

      // Get an array of Cookies associated with the this domain
      cookies = request.getCookies();

      if( cookies != null ) {
        out.println("<h2> Found Cookies Name and Value</h2>");

        for (int i = 0; i < cookies.length; i++) {
          cookie = cookies[i];

          if((cookie.getName( )).compareTo("first_name") == 0 ) {
            cookie.setMaxAge(0);
            response.addCookie(cookie);
            out.print("Deleted cookie: " +
            cookie.getName( ) + "<br/>");
          }
          out.print("Name : " + cookie.getName( ) + ",  ");
          out.print("Value: " + cookie.getValue( )+" <br/>");
        }
      } else {
        out.println(
        "<h2>No cookies founds</h2>");
      }
    %>
  </body>
```

Cookies Name and Value

Deleted cookie : first_name

Name : first_name, Value: JOHN

Name : last_name, Value: KUMAR

Found Cookies Name and Value

Name : last_name, Value: KUMAR

Introduction to JSP File Uploading

File upload करने का अर्थ है, local system से किसी file को server पर store करना। अलग अलग scripting languages स्वयं की file upload करने की facility provide करती है। JSP के द्वारा भी किसी file को server पर upload कर सकते है। JSP के द्वारा text file से लेकर binary file और image files upload की जा सकती है।

JSP को use करते हुए file upload करने के लिए HTML form tag का इस्तेमाल करते है। इस tag के द्वारा file upload करने के लिए user interface provide करते है। File upload करने के लिए user इसी form से interact करता है।

Server से JSP file के माध्यम से interact किया जाता है। JSP file को end user द्वारा form submit किये जाने पर execute कराया जाता है।

Requirement For File Uploading Using JSP

JSP को use करते हुए file server पर upload करने के लिए को निचे दी गयी चार files की आवश्यकता होगी।

1. **Commons.io.jar File** – ये एक Apache library होती है। इस library में के use के लिए utility classes, file filters, file comparators आदि classes available रहती है।
2. **Commons.fileupload.jar File** – ये भी एक Apache library होती है। ये library web applications के लिए file uploading को बहुत ही आसान बना देती है।
3. **User Interface JSP File** – ये वो JSP file होती है जिसमें file uploading के लिए HTML form design करते है। ये user interface होता है।
4. **Controller JSP File** – ये वो JSP file होती है जो form के submit होने पर file handling को handle करती है।

Steps to Upload Files Using JSP

JSP के द्वारा file upload करने के 2 steps होते है।

1. सबसे पहले HTML form tag के द्वारा user interface create किया जाता है।
2. Form submit होने पर file को server पर upload करने के लिए controller JSP file execute करवाई जाती है।

Creating User Interface (HTML form)

Files को upload करने के लिए HTML forms में एक special file type का input element होता है। इस element को file uploading के लिए ही use किया जाता है। इसका general syntax:

```
<input type="file" name="name-of-file">
```

File uploading के लिए create किये गए form में अनिवार्य रूप से 3 attributes define करते है।

1. **action** – इस attribute में उस JSP file का path देंगे जो form के submit होने पर execute होगी।
2. **method** – Form का method POST ही होना चाहिए। यदि ये method GET होता है तो file uploading properly work नहीं करेगी।
3. **enctype** – ये encode type attribute होता है। इसकी value हमेशा multipart/form-data देना ही अनिवार्य है।
4. File uploading के लिए user interface का उदाहरण निचे दिया जा रहा है।

```
<html> <head> <title>JSP File Upload Demo</title> </head> <body> <h1>Select file to upload :</h1> <form action="FileHandling.jsp" method="POST" enctype="multipart/form-data"> <input type="file" name="file1"> <input type="submit" value="Upload"> </form> </body> <html>
```

Create Controller JSP File

Controller JSP file में scriptlet tag के अंदर file को server पर upload करने के लिए java code लिखते है। सबसे पहले निचे दिए गए packages को import करते है।

1. java.io.*;
2. java.util.*;
3. javax.servlet.*;
4. javax.servlet.http.*;
5. org.apache.commons.fileupload.*;
6. org.apache.commons.fileupload.disk.*;
7. org.apcahe.commons.fileupload.servlet.*;
8. org.apache.commons.io.output.*;

2. Java code में सबसे पहले एक file type का object create करते है। ये object file को hold करने के लिए create किया जाता है।
3. इसके बाद upload की जाने वाली file की disk size limit define करने के लिए एक variable create करते है और उसमे file की maximum size bytes के रूप में बताते है।
4. इसके बाद upload की जाने वाली file की memory size define करने के लिए एक variable create करते है और उसमे file की maximum memory size bytes के रूप में define करते है।
5. इसके बाद file को जहाँ store करना चाहते है उस path को string variable करके define करते है।
6. इसके बाद form के द्वारा submit की गयी file का content type check करते है। इसके लिए पहले तो request object पर getContentType method call करते है और उसे एक string variable में store करवाते है।
7. इसके बाद if statement के द्वारा content type check करते है। यदि content type multipart/form-data है तो if statement block execute होगा।
8. इसके बाद DiskFileItemFactory class का object create करते है और इस object पर setSizeThreshold method call करते है जिसमें maximum memory limit variable pass किया जाता है ।
9. इसके बाद इसी object पर setRepository method call करते है और उसमे जिस directory में file को temporarily upload करना चाहते है उसका path दिया जाता है।
10. इसके बाद ServeltFileUpload class का variable create करते है और उसमे DiskFileItemFactory class का object pass करते है।

11. इसके बाद ServletFileUpload class के object पर setSizeMax() method call करते है और उसमे maximum file size define करने के लिए पहले create किया गया variable pass करते है।
12. इसके बाद try block start करते है और उसमें FileItem class का object create करते है।
13. इसके बाद ServletFileUpload class के object पर parseRequest method call करते है। इस method में request object pass करते है। इसके बाद इसे FileItem class के object को assign कर दिया जाता है।
14. इसके बाद FileItem class के object पर isFormField() method call करते है और check करते है की क्या ये request किसी form field से आयी है।
15. इसके बाद FileItem class के object पर getFieldName() method call करते है और इसके result को एक string variable में store करते है।
16. इसके बाद FileItem class के object पर getName() method call करते है और इसके result को भी एक string variable में store करवाते है।
17. इसके बाद इसी objcet पर isInMemory() और getSize() methods call करते है और उनके result को respective variables में store करवाते है।
18. इसके बाद file class का नया object create करते है और उसे file path का variable argument के रूप में pass करते है।
19. इसके बाद FileItem class के object पर write() method call करते है और उसमे argument के रूप में file class का object pass करते है।
20. इसके बाद out.println() method द्वारा एक message print करवाते है।
21. इसके बाद catch block में exceptions को handle करते है।
22. इसके बाद else part में appropriate message print करवाते है ।

```
<%
File                                                                    file;
int maxFsize = 5000*1024; //Give maximum file size in bytes
int maxMsize = 5000*1024; //Give maximum memory size in bytes
String filePath = "c:/apache-tomcat/webapps/data/";
String contentTypeObj = request.getContentType();
if((contentTypeObj.indexOf("multipart/form-data") >= 0))
{
      DiskFileItemFactory factoryObj = new DiskFileItemFactory();
                                    factoryObj.setSizeThreshold(maxMsize);
                        factoryObj.setRepository(new File(c://temp));
  servletFileUpload servletUploadObj = new ServletFileUpload(factoryObj);
                                    servletUploadObj.setSizeMax(maxFsize);
try{
              FileItem fiObj = upload.parseRequest(request);
                                                  if(!fiObj.isFormField())
                                                                        {
                        String fieldName = fiObj.getFieldName();
                            String fileName = fiObj.getName();
                      boolean isInMemory = fiObj.isInMemory();
                                long sizeInBytes = fi.getSize();
                    file = new File(filePath + "File Name");
                                                        fiObj.write(file);
                     out.println("File was uploaded successfully!")
                                                                        }
}
catch(Exception                                                         e)
{
                                                    system.out.println(e);
}
```

```
else{
out.println("File was not uploaded!")
}
}
%>
```

JSP - Database Access

Create Table

To create the **Employees** table in the EMP database, use the following steps −

Step 1

Open a **Command Prompt** and change to the installation directory as follows −

```
C:\>
C:\>cd Program Files\MySQL\bin
C:\Program Files\MySQL\bin>
```

Step 2

Login to the database as follows −

```
C:\Program Files\MySQL\bin>mysql -u root -p
Enter password: ********
mysql>
```

Step 3

Create the **Employee** table in the **TEST** database as follows − −

```
mysql> use TEST;
mysql> create table Employees
  (
    id int not null,
    age int not null,
    first varchar (255),
    last varchar (255)
  );
Query OK, 0 rows affected (0.08 sec)
mysql>
```

Create Data Records

Let us now create a few records in the **Employee** table as follows − −

```
mysql> INSERT INTO Employees VALUES (100, 18, 'Zara', 'Ali');
Query OK, 1 row affected (0.05 sec)

mysql> INSERT INTO Employees VALUES (101, 25, 'Mahnaz', 'Fatma');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO Employees VALUES (102, 30, 'Zaid', 'Khan');
```

```
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO Employees VALUES (103, 28, 'Sumit', 'Mittal');
Query OK, 1 row affected (0.00 sec)

mysql>
```

SELECT Operation

Following example shows how we can execute the **SQL SELECT** statement using JTSL in JSP programming −

```
<%@ page import = "java.io.*,java.util.*,java.sql.*"%>
<%@ page import = "javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix = "c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix = "sql"%>

<html>
  <head>
    <title>SELECT Operation</title>
  </head>

  <body>
    <sql:setDataSource var = "snapshot" driver = "com.mysql.jdbc.Driver"
      url = "jdbc:mysql://localhost/TEST"
      user = "root"  password = "pass123"/>

    <sql:query dataSource = "${snapshot}" var = "result">
      SELECT * from Employees;
    </sql:query>

    <table border = "1" width = "100%">
      <tr>
        <th>Emp ID</th>
        <th>First Name</th>
        <th>Last Name</th>
        <th>Age</th>
      </tr>

      <c:forEach var = "row" items = "${result.rows}">
        <tr>
          <td><c:out value = "${row.id}"/></td>
          <td><c:out value = "${row.first}"/></td>
          <td><c:out value = "${row.last}"/></td>
          <td><c:out value = "${row.age}"/></td>
        </tr>
      </c:forEach>
    </table>

  </body>
</html>
```

Access the above JSP, the following result will be displayed −

Emp ID	First Name	Last Name	Age
100	Zara	Ali	18
101	Mahnaz	Fatma	25
102	Zaid	Khan	30
103	Sumit	Mittal	28

INSERT Operation

Following example shows how we can execute the SQL INSERT statement using JTSL in JSP programming −

```
<%@ page import = "java.io.*,java.util.*,java.sql.*"%>
<%@ page import = "javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix = "c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix = "sql"%>

<html>
  <head>
    <title>JINSERT Operation</title>
  </head>

  <body>
    <sql:setDataSource var = "snapshot" driver = "com.mysql.jdbc.Driver"
      url = "jdbc:mysql://localhost/TEST"
      user = "root"  password = "pass123"/>
      <sql:update dataSource = "${snapshot}" var = "result">
      INSERT INTO Employees VALUES (104, 2, 'Nuha', 'Ali');
    </sql:update>

    <sql:query dataSource = "${snapshot}" var = "result">
      SELECT * from Employees;
    </sql:query>

    <table border = "1" width = "100%">
      <tr>
        <th>Emp ID</th>
        <th>First Name</th>
        <th>Last Name</th>
        <th>Age</th>
      </tr>

      <c:forEach var = "row" items = "${result.rows}">
        <tr>
          <td><c:out value = "${row.id}"/></td>
          <td><c:out value = "${row.first}"/></td>
```

```
            <td><c:out value = "${row.last}"/></td>
            <td><c:out value = "${row.age}"/></td>
          </tr>
        </c:forEach>
      </table>

    </body>
</html>
```

Access the above JSP, the following result will be displayed −

Emp ID	First Name	Last Name	Age
100	Zara	Ali	18
101	Mahnaz	Fatma	25
102	Zaid	Khan	30
103	Sumit	Mittal	28
104	Nuha	Ali	2

DELETE Operation

Following example shows how we can execute the **SQL DELETE** statement using JTSL in JSP programming −

```
<%@ page import = "java.io.*,java.util.*,java.sql.*"%>
<%@ page import = "javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix = "c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix = "sql"%>

<html>
  <head>
    <title>DELETE Operation</title>
  </head>

  <body>
    <sql:setDataSource var = "snapshot" driver = "com.mysql.jdbc.Driver"
       url = "jdbc:mysql://localhost/TEST"
       user = "root" password = "pass123"/>

    <c:set var = "empId" value = "103"/>

    <sql:update dataSource = "${snapshot}" var = "count">
       DELETE FROM Employees WHERE Id = ?
       <sql:param value = "${empId}" />
    </sql:update>
```

```
   <sql:query dataSource = "${snapshot}" var = "result">
      SELECT * from Employees;
   </sql:query>

   <table border = "1" width = "100%">
      <tr>
         <th>Emp ID</th>
         <th>First Name</th>
         <th>Last Name</th>
         <th>Age</th>
      </tr>

      <c:forEach var = "row" items = "${result.rows}">
         <tr>
            <td><c:out value = "${row.id}"/></td>
            <td><c:out value = "${row.first}"/></td>
            <td><c:out value = "${row.last}"/></td>
            <td><c:out value = "${row.age}"/></td>
         </tr>
      </c:forEach>
   </table>

  </body>
</html>
```

Access the above JSP, the following result will be displayed −

Emp ID	First Name	Last Name	Age
100	Zara	Ali	18
101	Mahnaz	Fatma	25
102	Zaid	Khan	30

UPDATE Operation

Following example shows how we can execute the **SQL UPDATE** statement using JTSL in JSP programming −

```
<%@ page import = "java.io.*,java.util.*,java.sql.*"%>
<%@ page import = "javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri = "http://java.sun.com/jsp/jstl/core" prefix = "c"%>
<%@ taglib uri = "http://java.sun.com/jsp/jstl/sql" prefix = "sql"%>

<html>
  <head>
    <title>DELETE Operation</title>
  </head>
```

```
  <body>
    <sql:setDataSource var = "snapshot" driver = "com.mysql.jdbc.Driver"
       url = "jdbc:mysql://localhost/TEST"
       user = "root" password = "pass123"/>

    <c:set var = "empId" value = "102"/>

    <sql:update dataSource = "${snapshot}" var = "count">
       UPDATE Employees SET WHERE last = 'Ali'
       <sql:param value = "${empId}" />
    </sql:update>

    <sql:query dataSource = "${snapshot}" var = "result">
       SELECT * from Employees;
    </sql:query>

    <table border = "1" width = "100%">
       <tr>
          <th>Emp ID</th>
          <th>First Name</th>
          <th>Last Name</th>
          <th>Age</th>
       </tr>

       <c:forEach var = "row" items = "${result.rows}">
          <tr>
             <td><c:out value = "${row.id}"/></td>
             <td><c:out value = "${row.first}"/></td>
             <td><c:out value = "${row.last}"/></td>
             <td><c:out value = "${row.age}"/></td>
          </tr>
       </c:forEach>
    </table>

  </body>
</html>
```

Access the above JSP, the following result will be displayed −

Emp ID	First Name	Last Name	Age
100	Zara	Ali	18
101	Mahnaz	Fatma	25
102	Zaid	Ali	30