WEB DESIGNING

PGDCA

2nd

JavaScript

JavaScript Introduction (जावा स्क्रिप्ट का परिचय)

जावास्क्रिप्ट का विकास नेटस्केप कम्युनिकेशन (Netscape Communication) नामक कंपनी के Brendan Eich द्वारा किया गया था इसे पहली बार 1995 में नेटस्केप नेविगेटर 2.0 नामक ब्राउज़र प्रोग्राम के साथ जारी किया गया था और प्रारंभ में इसका नाम लाइव स्क्रिप्ट (LiveScript) था| लेकिन Java नाम की लोकप्रियता के कारण इसका नाम बाद में बदलकर जावास्क्रिप्ट रखा गया| Java सन माइक्रोसिस्टम (Sun Micro system) नामक कंपनी द्वारा सभी प्लेटफार्म पर चलने वाली ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग (OOP) भाषा के रूप में विकसित की गई है इसने शीघ्र ही मान्यता प्राप्त कर ली थी इसलिए लाइव स्क्रिप्ट का नाम बदलकर जावास्क्रिप्ट रख दिया गया|

माइक्रोसॉफ्ट कॉरपोरेशन नामक कंपनी ने जावा स्क्रिप्ट के महत्व को पहचाना और इससे मिलती-जुलती दो स्क्रिप्टिंग भाषाओं को प्रस्तुत किया – एक jscript जो जावास्क्रिप्ट से बहुत समानता रखती है और दूसरी VBScript जो विज़ुअल बेसिक का ही एक भाग या सब सेट है इन एक जैसी कई भाषाओं ने वेब डेवलपर के लिए बहुत समस्या पैदा की क्योंकि किसी भी ब्राउज़र में इन सभी के कोड को interprit करने की क्षमता नहीं हैं | इसलिए अंत में नेटस्केप, माइक्रोसॉफ्ट तथा अन्य कंपनियां एक सामान्य स्क्रिप्टिंग भाषा के रूप में जावास्क्रिप्ट को स्वीकार करने को तैयार हो गए| तत्पश्चात यूरोपीय कंप्यूटर निर्माता संघ ECMA (EUROPEAN COMPUTER MANUFACTURER ASSOCIATION) ने इसका मानक रूप जुलाई 1997 में जारी किया जो आज प्रचलित है हालांकि अभी भी कोई भी ब्राउजर इसके साथ 100% complaint नहीं है|

जावास्क्रिप्ट एक स्क्रिप्टिंग भाषा है इसका तात्पर्य है कि यह एक ऐसी प्रोग्रामिंग भाषा है जो सीखने और प्रयोग करने में सरल है और जिसका प्रयोग छोटे-छोटे रूटीन या उपयोगों को लिखने में किया जाता है छोटे-छोटे प्रोग्रामों को ही स्क्रिप्ट कहा जाता है जावास्क्रिप्ट का विकास वेबपेजों में वार्तालाप (interactivity) संभव करने के लिए किया गया था|

इस समय जावा स्क्रिप्ट 3 रूपों में मिलता है –

1. Core JavaScript
2. Client Side JavaScript
3. Server Side JavaScript

1. इसमें कोर जावा स्क्रिप्ट (Core JavaScript) मौलिक जावास्क्रिप्ट भाग है इसमें ऑपरेटर (Operator),कंट्रोल संरचनाएं (Control Structure), बिल्ट इन फंक्शन (Built in function), तथा ऑब्जेक्ट (Object) शामिल है जिनसे मिलकर जावास्क्रिप्ट एक प्रोग्रामिंग भाषा बनती है|

2. क्लाइंट साइड जावास्क्रिप्ट (Client Side JavaScript) कोर जावास्क्रिप्ट का एक विस्तार है जो किसी ब्राउज़र को नियंत्रित करने के लिए तैयार किया गया है यह जावास्क्रिप्ट का सबसे लोकप्रिय रूप है|

3. सर्वर साइड जावास्क्रिप्ट (Server Side JavaScript) भी कोर जावा स्क्रिप्ट का एक अन्य विस्तार है जो डेटाबेस के उपयोग के लिए तैयार किया गया है यह कहीं अधिक जटिल है और सभी ब्राउज़र इसको सपोर्ट नहीं करते|

आजकल लगभग सभी ब्राउज़र क्लाइंट साइड जावास्क्रिप्ट को सपोर्ट करते हैं|

JavaScript Basic and Element of JavaScript

(JavaScript बेसिक्स या JavaScript के तत्व(

JavaScript में लगभग वे सभी प्रोग्रामिंग क्षमताएं होती हैं जो अधिकांश प्रोग्रामिंग भाषाओं में पाई जाती हैं जैसे variable, control structure, constants, user define functions आदि इन सभी Programming तकनीकों का प्रयोग किसी भी HTML डॉक्यूमेंट में डाले गए JavaScript कोड में किया जा सकता है JavaScript की इन तकनीकों के कारण HTML की क्रियाशीलता बढ़ जाती है और वेब पेज interactive बन जाते हैं|

JavaScript के चरों (Variable),नियतांको (Constants), फ़ंक्शन (Function) आदि को परिभाषित करने के लिए HTML डॉक्यूमेंट का <head> section सबसे आदर्श स्थान है| इसका कारण यह है कि यह सेक्शन हमेशा <body> section से पहले प्रोसेस किया जाता है JavaScript code में उपयोग किए जा रहे Variables को <head> सेक्शन में परिभाषित करने से उनका आगे स्वतंत्रतापूर्वक उपयोग किया जा सकता है यह बहुत महत्वपूर्ण है, क्योंकि किसी तत्व को बिना घोषित या परिभाषित किए उपयोग करने पर गलत संदेश प्राप्त होगा|

1. Data type and Literals/Constant (डेटा टाइप और अचर)

JavaScript में किसी Variable का Data type पहले से घोषित नहीं किया जाता अतः आप एक ही Variable को अलग अलग समय पर अलग-अलग प्रकार का डाटा स्टोर करने के लिए भी उपयोग कर सकते हैं| वैसे JavaScript में Variables को 4 प्राथमिक प्रकार का Constant Data दिया जा सकता है जो निम्न प्रकार है-

- Number (संख्या)

संख्याएं सामान्यता पूर्णांक (integer) अथवा फ्लोटिंग पॉइंट (floating point) हो सकती हैं पूर्णांक में कोई दशमलव बिंदु नहीं होता जबकि फ्लोटिंग पॉइंट संख्या में दशमलव बिंदु का प्रयोग किया जाता है फ्लोटिंग पॉइंट संख्याओं में घात (exponent) भी हो सकता है जो E अक्षर के बाद दिया जाता है उदाहरण के लिए 12, 0, -4, 333 आदि सभी पूर्णांक संख्याएं हैं और 1.0, 5.345, -56.3, 24.4E4 यह सभी फ्लोटिंग पॉइंट संख्याएं हैं इनके अतिरिक्त JavaScript में एक विशेष NaN (not a Number) मान भी होता है|

- Boolean (बुलियन)

बुलियन चर या अचर के दो मान हो सकते हैं true and false बूलियन व्यंजको में लॉजिकल ऑपरेटर जैसे AND, OR, NOT आदि का प्रयोग किया जा सकता है JavaScript बुलियन मानो true और false को संख्यात्मक व्यंजनों में प्रयोग किए जाने वाले अपने आप क्रमशः 1 और 0 में बदल देता है|

ध्यान रहे कि जावा स्क्रिप्ट में 1 और 0 को बुलियन मान नहीं माना जाता|

- String (स्ट्रिंग)

सिंगल (Single) या डबल (Double) कोटेशन चिन्हों में रखे गए शून्य या अधिक चिन्हों को स्ट्रिंग कहा जाता है उदाहरण के लिए "Ashok", 'Ram' यह सभी स्ट्रिंग हैं यदि किसी संदर्भ चिह्न को स्ट्रिंग में शामिल करना है तो उससे पहले एक backslash (\) लगाना चाहिए| उदाहरण के लिए, यदि किसी string में आप Don't रखना चाहते हैं तो इसे या तो "Don't" लिखें या 'Don\'t' इस तरह लिखें|

- Null (नल)

यह केवल एक मान null को व्यक्त करता है जो खालीपन दिखाता है जावा स्क्रिप्ट में सामान्यतया इसका उपयोग चरों को प्रारंभ में घोषित करते समय प्रारंभिक मान रखने में किया जाता है यदि ऐसा ना किया जाए तो किसी variable में अनपेक्षित मान भी भरा हो सकता है|

2. Variable (चर)

Variable (चर) विभिन्न मानो को स्टोर करने के लिए प्रयोग किए जाते हैं और उनका कोई नाम भी रखा जाता है, जिनके द्वारा उनका संदर्भ दिया जाता है चरों के नाम ऐसे रखे जाने चाहिए कि उनसे इसका पता चलता हो कि किस चर का क्या उपयोग किया जाएगा अर्थात उस variable में कौन सा मान स्टोर किया जाएगा variables के नाम अंग्रेजी वर्णमाला के अक्षरों a से z तथा A से Z अथवा अंडरस्कोर (__ से प्रारंभ हो सकते हैं|

JavaScript में अंग्रेजी वर्णमाला के छोटे और बड़े अक्षरों को अलग-अलग माना जाता है अर्थात JavaScript केस सेंसिटिव है वैसे सुविधा की दृष्टि से हम चरों के नाम हमेशा छोटे अक्षरों में रखते हैं यदि नाम बड़ा है तो प्रत्येक शब्द का पहला अक्षर केपिटल कर देते हैं और शेष अक्षर छोटे रहते हैं जैसे FirstName, DateOfBirth, TotalMarks आदि आप इनमे से किसी भी परंपरा को अपना सकते हैं HTML में space का कोई महत्व नहीं है परंतु JavaScript में इनका महत्व होता है|

Creating Variable (चर बनाना)

जावा स्क्रिप्ट में चरों को पहले से बना लेना अर्थात घोषित करना अनिवार्य नहीं है परंतु ऐसा करना अच्छी प्रोग्रामिंग परंपरा है वैसे हम बिना घोषित किए भी किसी चर का उपयोग कर सकते हैं चर घोषित करने के लिए जावा स्क्रिप्ट में var आदेश का उपयोग किया जाता है इसका सामान्य रूप निम्न प्रकार है-

Var<variable nam>=value;

जहां <variable name> उस चर का नाम है और Value उसका प्रारंभिक मान है बरबा चिन्ह (=) का प्रयोग उस चर का मान निर्धारित करने के लिए किया जाता है इसलिए इस ऑपरेटर को एक असाइनमेंट ऑपरेटर (assignment operator) कहा जाता है चरो का प्रारंभिक मान रखना वैकल्पिक (optional) है|

var first_name = "Ashish Kumar";

var roll_no;

var phone_no = 256485;

Learn JavaScript in Hindi Tutorial (Define and Declare Variable using VAR and LET)

जावास्क्रिप्ट में वेरिएबल कैसे बनाते हैं और वेरिएबल बनाते समय क्या बातें याद रखनी पड़ती है जानने के लिए नीचे दिए गए विडियो को जरुर देखें

3. Operator and Expressions (ऑपरेटर और व्यंजक)

किसी ऑपरेटर का प्रयोग एक या अधिक मानो को केवल एक मान में परिवर्तित करने के लिए किया जाता है जिन मानो पर ऑपरेटर को लागू किया जाता है उन्हें Operands कहा जाता है किसी ऑपरेटर और उसके operands के संयोग को मुद्रा या व्यंजक (expression) कहा जाता है किसी व्यंजक का मान निकालने के लिए उसमें दिए गए ऑपरेटरों को उनके operands के न्यूनतम मान पर लागू किया जाता है और अंत में एक परिणामी मान निकाला जाता है|

JavaScript में उपयोग किए जाने वाले ऑपरेटर निम्न प्रकार हैं-

- Arithmetic Operators (अंकगणितीय ऑपरेटर)

इनका प्रयोग गणितीय क्रियाएं अर्थात गणनाएं करने के लिए किया जाता है JavaScript के अंकगणितीय ऑपरेटर निम्न सारणी में दर्शाए गए हैं-

Operator	Result
+	Addition (also unary plus)
–	Subtraction (also unary minus)
*	Multiplication
/	Division
%	Modulus
++	Increment
+=	Addition assignment
– =	Subtraction assignment
*=	Multiplication assignment
/=	Division assignment
%=	Modulus assignment
– –	Decrement

जिस ऑपरेटर के लिए केवल एक operand की आवश्यकता होती है उसे यूनेरी (unary) ऑपरेटर कहा जाता है और जिसके लिए दो operands की आवश्यकता होती है उसे Binary Operator कहते हैं सभी प्रचलित अंकगणितीय ऑपरेटर Binary हैं, जबकि ++ और — यूनेरी ऑपरेटर है|

वृद्धि (++) और कमी (–) ऑपरेटरों को दो प्रकार से उपयोग में लाया जा सकता है operand पहले और operand के बाद उदाहरण के लिए ++X देने पर x का मान पहले एक से बढ़ाया जाएगा फिर परिणाम लौटाया जाएगा जबकि x++ देने पर पहले x का मान घटाया जाएगा फिर उसे एक से बढ़ा दिया जाएगा इसी प्रकार – – X देने पर x का मान पहले एक से घटाया जाएगा और फिर परिणाम लौटाया जाएगा जबकि X– देने पर पहले x का मान लौटाया जाएगा फिर उसे एक से घटा दिया जाएगा उदाहरण के लिए निम्नलिखित निर्धारण कथनों पर ध्यान दीजिए-

```
X=3;
Y=x++;
Z=++x;
```

यहां पहले कथन के कारण Variable x का मान 3 रख दिया जाएगा| दूसरे कथन से Variable Y का मान पहले Variable X के बराबर अर्थात 3 रखा जाएगा| फिर X को एक से बढ़ा दिया जाएगा अर्थात अब x का मान 4 हो जाएगा| तीसरे कथन से पहले variable X का मान 1 से बढ़ाया जाएगा अर्थात x का मान 5 हो जाएगा फिर वह मान variable Z में लौटाया जाएगा अर्थात Z का मान 5 होगा|इस प्रकार कथनों के परिणाम स्वरुप X का मान 3, Y का मान 3 और Z का मान 5 हो जायेगा |

- Logical Operators (तार्किक ऑपरेटर)

इन ऑपरेटर का उपयोग Boolean Operands पर Boolean Operations करने के लिए किया जाता हैं | Java Script में केवल तीन तार्किक ऑपरेटर होते हैं, जो निम्न सारणी में दर्शाए गए हैं |

Logical Operator	Java Operator
AND	&&
OR	||
NOT	!

- Comparison Operators (तुलनात्मक ऑपरेटर)

इन ऑपरेटर्स का प्रयोग दों मानो की तुलना करने के लिए किया जाता हैं और इनका परिणाम Boolean मानो अर्थात True अथवा False में होता हैं| JavaScript में उपयोग किये जाने वाले तुलना ऑपरेटर निम्न सारणी में दर्शाए गए हैं-

Operators	Meaning	Example	Result
<	Less than	5<2	False
>	Greater than	5>2	True
<=	Less than or equal to	5<=2	False
>=	Greater than or equal to	5>=2	True
==	Equal to	5==2	False
! =	Not equal to	5! =2	True
===	Equal value and same type	5 === 5	True
		5 === "5"	False
! ==	Not Equal value or Not same type	5 ! == 5	False
		5 ! == "5"	True

- Assignment Operators (निर्धारण ऑपरेटर)

किसी निर्धारण ऑपरेटर का उपयोग किसी चर का मान बदलने अर्थात उसका नया मान रखने के लिए किया जाता हैं | मूल निर्धारण ऑपरेटर केवल एक हैं -'=', जिसे कुछ अंकगणितीय ऑपरेटर्स के साथ मिलकर अन्य निर्धारण ऑपरेटर बनाये गए हैं| जावास्क्रिप्ट में उपलब्ध निर्धारण ऑपरेटर निम्न सारणी में दर्शाए गए हैं –

- String Operator (स्ट्रिंग ऑपरेटर)

इस ऑपरेटर का प्रयोग केवल स्ट्रिंगो पर क्रियाये करने के लिए किया जाता हैं | जावास्क्रिप्ट में ऐसा केवल एक ऑपरेटर हैं +, जिसे स्ट्रिंग योग (String concatenation) ऑपरेटर कहा जाता हैं | इस ऑपरेटर का उपयोग दो स्ट्रिंगो को मिलाकर एक स्ट्रिंग बनाने के लिए किया जाता हैं| उदाहरण के लिए, "abc"+"opq" का परिणाम "abcopq" होगा|

- Special Operator (विशेष ऑपरेटर)

जावास्क्रिप्ट में कई ऑपरेटर भी हैं , जो ऊपर बताई गई किसी श्रेणी में नहीं आते | इन्हें विशेष ऑपरेटर कहा जाता हैं| ऐसे तीन प्रमुख ऑपरेटर हैं, जिनका परिचय नीचे दिया गया हैं-

Delete इस ऑपरेटर का प्रयोग किसी array के किसी तत्व को हटाने के लिए किया जाता हैं |

new इस ऑपरेटर का प्रयोग किसी array के किसी तत्व को जोड़ने के लिए किया जाता हैं|

void यह ऑपरेटर कोई मान नहीं लौटाता| सामान्यतया इसका उपयोग किसी URL को खली मान देने के लिए किया जाता हैं|

Learn JavaScript in Hindi Tutorial (Expression and Types of Operator in JavaScript)

जावास्क्रिप्ट में वेरिएबल कैसे बनाते हैं और वेरिएबल बनाते समय क्या बातें याद रखनी पड़ती है

Control Flow statements Looping in Javascript

Control statements program के flow को control करते है। जैसे की आप control statements की मदद से आप चुन सकते है की आप कौन सा स्टेटमेंट execute करवाना चाहते है और कौन सा नहीं करवाना चाहते है। Control statements की मदद से logic perform किया जाता है।

सरल शब्दों में कह सकते है की ,आप चुन सकते है की कौन-सा स्टेटमेंट किस situation में execute होगा,और साथ ही आप control statements की मदद से एक statement को कई बार execute कर सकते है।

Types of Control Flow Statements

Control statements को तीन केटेगरी में बांटा गया है। ये केटेगरी control statements के tasks को भी define करती है।

- Selection statements/Conditional statements
- Looping statements
- Jump statements

Selection statements / Conditional statements

इस category के control statements program में statements को situation के अनुसार सेलेक्ट करके execute करने के लिए प्रयोग किये जाते है। ये निम्न प्रकार के होते है-

- If
- If-Else
- Nested-If
- Switch case

If Statement

If statement, condition को test करता है यदि condition true होती है तो brackets में दिए हुआ statement execute होता है और यदि condition false है तो control बाहर चला जाता है।

Example:1

```
if(10>3)
{
document.write("This will be displayed");
}
```

जैसा की आप ऊपर दिए हुए उदाहरण में देख रहे है दी हुई condition true है इसलिए brackets के अंदर का statement execute होगा। आइये इसका एक और उदाहरण देखते है।

```
if(3>10)
{
document.write("This will not be displayed");
}
```

इस उदाहरण में condition false है इसलिए brackets में दिया हुआ statement execute नहीं होगा।

If else statement

If else statement भी if statement की तरह ही होता है। बस इसमें else वाला भाग भी जोड़ दिया जाता है। Else part में आप वो statements लिखते है जो condition false होने पर execute होना चाहिए। आइये इसका उदाहरण देखते है।

```
if(num%2==0)
{
document.write("Number is Positive");
}
else
{
document.write("Number is Negative");
}
```

Else If Statement

यदि आप चाहते है की एक condition के false होने पर else part को execute ना करके किसी दूसरी condition को check किया जाये तो इसके लिए आप else if statements use कर सकते है।

Else if statements के द्वारा आप एक से अधिक conditions को check कर सकते है और सभी condition के false होने पर else part को execute करवा सकते है।इसके लिए आप elseif keyword यूज़ करते है। First condition को normal if else statement की तरह execute किया जाता है। इसके अलावा आप जितनी भी conditions add करना चाहते है उन्हें if और else part के बीच elseif keyword के द्वारा डिफाइन करते है।इसका सिंटेक्स निम्नानुसार है –

```
if(condition 1)
{
// Will be executed if above condition is true
}
else if(condition 2)
{
// Will be executed if 1st condition is false and this condition is true.
}

....
....
....
else if(condition N)
{
// Will be executed if all the conditions above it were false and this condition is true.
}
else
{
// Will be executed if all the above conditions are false
}
```

Example :2

```
if(num>0)
{
document.write("Number is Positive");
}
elseif( num==0)
{
document.write("Number is Zero");
}
else
{
document.write(" Number is Negative");
}
```

आइये अब इसे एक उदाहरण से समझने का प्रयास करते है।

Nested If Statement

यदि आप आप चाहे तो एक if condition में दूसरी if condition भी डाल सकते है। इसका structure नीचे दिया जा रहा है।

```
if(condition)
{
if(condition)
{
// Statement to be executed
}
}
else
{
// Statements to be executed
}
```

जैसा की आप ऊपर दिए गए syntax में देख सकते है एक if condition के अंदर दूसरी if condition define की गयी है। आप चाहते तो nested if में else part भी add कर सकते है। आइये अब इसे एक उदाहरण से समझने का प्रयास करते है।

Example:3

```
if(5>3)
{
if(5>6)
{
document.write("This will not be executed");
}
else
{
document.write("5 is greater than 3 but not 6");
}
}
else
{
document.write("5 is not greater than 3");
}
```

Switch Case

Switch case बिलकुल if statement की तरह होता है। लेकिन इसमें आप एक बार में एक से ज्यादा conditions को check कर सकते है। Switch case में cases define किये जाते हैं। बाद में एक choice variable के द्वारा ये cases execute करवाए जाते है। Choice variable जिस case से match करता है वही case execute हो जाता है। इसका उदाहरण नीचे दिया जा रहा है।

Example :4

```
var day=2;
// Passing choice to execute desired case
switch(day)
{
case 1:
document.write("Monday");
break;
case 2:
document.write("Tuesday");
break;
case 3:
document.write("Wednesday");
break;
case 4:
document.write("Thursday");
break;
case 5:
document.write("Friday");
break;
case 6:
document.write("Saturday");
break;
default:
document.write("Wrong input");
break;
}
```

जैसे की आप देख सकते है हर case के बाद में break statement यूज़ किया गया है। यदि आप break statement यूज़ नहीं करते है तो सभी cases one by one execute हो जाते है। इस उदाहरण में variable की value 2 है इसलिए second case execute होगा और Tuesday display हो जायेगा।

Looping Statements

Looping statements particular statement को बार-बार execute करने के लिए यूज़ किये जाते है। ये 3 प्रकार के होते है। इनके बारे में नीचे दिया जा रहा है।

While Loop Statement

इस loop में आप एक condition देते है जब तक condition true होती है block में दिए गए statements execute होते रहते है। Condition false होते ही loop terminate हो जाता है और program का execution continue रहता है।

Example :5

```
var num = 0;
// While loop iterating until num is less than 5
while(num <5)
{
document.write("Hello");
num++;
}
```

इस उदाहरण में जब तक num 5 से कम है तब तक loop का block execute होगा। एक चीज़ यहां पर नोटिस करने की ये है की हर बार

num को increment किया जा रहा है ताकि कुछ स्टेप्स के बाद loop terminate हो जाये। यदि यहां पर ऐसा नहीं किया जाये तो loop कभी terminate ही नहीं होगा infinite time तक चलेगा। इसलिए इस situation से बचने के लिए किसी भी प्रकार के loop में loop control variable को increment किया जाता है।

Do-While Loop Statement

Do while loop भी while loop की तरह ही होता है। बस ये first time बिना condition check किये execute होता है और बाद में हर बार condition check करता है। यदि condition true होती है तो do block के statements execute कर दिए जाते है। आइये इसे एक उदाहरण से समझते है।

Example :6

```
var num=0;
// Do-while loop
do{document.write("hello");
num++;}
while(num>5);
```

जैसा की आप देख सकते है पहले do block execute होगा और उसके बाद condition check की जाएगी। इस loop की विशेषता ये है की चाहे condition true हो या false loop एक बार तो जरूर execute होगा। यदि condition true होती है तो loop further execute होता है नहीं तो terminate हो जाता है।

For Loop Statement

सभी loops में for loop सबसे सरल है और सबसे ज्यादा प्रयोग किया जाने वाला loop है। इसमें आप single line में ही पूरे loop को define कर देते है। यदि condition true होती है तो block में दिए गए statements execute हो जाते है। इस loop का उदाहरण नीचे दिया गया है।

```
// For loop running until i is less than 5
for(var i=0;i<5;i++)
{
document.write("This will be printed until condition is true");
}
```

For loop में condition और increment दोनों एक साथ ही define किये जाते है। साथ ही इसमें loop control variable भी define किया जाता है। Condition के false होते ही loop terminate हो जाता है।

Jump Statements

Jump statements program के execution को एक जगह से दूसरी जगह transfer करने के लिए यूज़ किये जाते है। इन statements को special cases में यूज़ किया जाता है। इनके बारे में नीचे दिया जा रहा है।

Continue Statement

Continue statement के द्वारा आप किसी भी loop की कोई iteration skip कर सकते है। जैसे की आप चाहते है की 3rd iteration skip हो जाये और compiler कोई action ना ले।

Example :7

```
for(var i=0; i<5;i++)
{
if(i==2)
```

```
{
// Skipping third iteration of loop
continue;
}
document.write("This will be displayed in iterations except 3rd");
}
```

Continue statement को प्रयोग करने से compiler 3rd iteration को skip कर देगा और कोई भी statement execute नहीं किया जायेगा। इसके बाद next iteration शुरू हो जायेगी।

Break Statement

Break statement compiler के execution को stop करने के लिए यूज़ किया जाता है। Break statement आने पर compiler execution को उस block से बाहर ला देता है। इसको एक loop के example से आसानी से समझा जा सकता है।

```
for(var i=0;i<5;i++)
{
if(i==2)
{
//Breaking 3rd iteration of loop
break;
}
document.write("This will be displayed 2 times only");
}
```

उपर दिए गए उदाहरण में जैसे ही loop की 3rd iteration आती है तो break statement के द्वारा loop terminate हो जाता है और program का execution loop के बाहर से शुरू हो जाता है।

Functions in JavaScript (जावास्क्रिप्ट में Function)

Function जावास्क्रिप्ट कोड के ऐसे समूह है जो किसी विशेष कार्य को करते हैं और प्रायः कोई मान लौटाते हैं किसी Function में 0 या अधिक पैरामीटर हो सकते हैं पैरामीटर ऐसी मानक तकनीक है, जिसके द्वारा हम किसी Function को दिए जाने वाले डाटा को नियंत्रित कर सकते हैं किसी Function को पास किए जाने वाले डाटा के आधार पर ही वह Function प्रायः कोई मान लौटाता है|

जावास्क्रिप्ट में Function दो प्रकार के होते हैं-

- Built in functions
- User define functions

Built in functions

यह ऐसे Function है, जो जावास्क्रिप्ट में पहले से उपलब्ध है इनका उपयोग विशेष प्रकार के टाइप कन्वर्शन (type conversion) के लिए किया जाता है ऐसे कुछ Functions का परिचय आगे दिया जा रहा है-

Eval () function

इस Function का उपयोग किसी स्ट्रिंग व्यंजक को संख्यात्मक मान में बदलने के लिए किया जा सकता है| उदाहरण के लिए,

Var g_total = eval ("3 *4 + 5")

का परिणाम यह होगा कि चर g_total में संख्या 17 Store हो जाएगी|

ParseInt () function

इस Function का उपयोग किसी स्ट्रिंग व्यंजक को पूर्णांक संख्या में बदलने के लिए किया जा सकता है| यदि इसको दी गई स्ट्रिंग किसी पूर्णांक से प्रारंभ होती है तो वह संख्या लौटाई जाएगी नहीं तो 0 लौटाया जाएगा| उदाहरण के लिए,

Var a_num = parseInt ("12.3abcd")

के परिणाम स्वरुप variable a_num में संख्या 12 स्टोर हो जाएगी तथा

Var a_num = parseInt ("abc12.3xyz")

के परिणाम स्वरुप variable a_num में "NaN" स्टोर होगा, जिसका अर्थ है "not a number"

ParseFloat() function

इस Function का उपयोग किसी स्ट्रिंग व्यंजक को अपूर्णांक संख्या में बदलने के लिए किया जा सकता है| यदि इसको दी गई स्ट्रिंग किसी प्लॉटिंग संख्या से प्रारंभ होती है तो वह संख्या लौटाई जाएगी नहीं तो 0 लौटाया जाएगा|
उदाहरण के लिए,

Var a_num = parsefloat ("12.3abcd")

के परिणाम स्वरुप variable a_num में संख्या 12.3 स्टोर हो जाएगी|

User defined functions

यह ऐसे Function है जिन्हें कोई उपयोगकर्ता अपनी आवश्यकता और सुविधा के अनुसार परिभाषित कर सकता है और उनका built-in-functions की तरह ही उपयोग कर सकता है| ऐसे किसी Function को उपयोग में लाने से पहले उसे घोषित करना अनिवार्य है| Function को घोषित करने, उनको कॉल करने, उनको पैरामीटर के मान पास करने तथा उनके द्वारा लौटाए गए मानों को स्वीकार करने के लिए उचित व्याकरण नियमों का पालन करना अनिवार्य है|

Function घोषित करना(Declaration of functions)

Function का निर्माण और घोषणा Function कीवर्ड के द्वारा की जाती है किसी Function के निम्नलिखित तीन भाग होते हैं-

1. Function का नाम
2. उसके parameter या argument की सूची, जो Function को कॉल करते समय उसको भेजे गए मानों को स्वीकार करेंगे और
3. उस Function को परिभाषित करने वाले जावास्क्रिप्ट के आदेश अर्थात कोड|

किसी Function की घोषणा का सामान्य रूप निम्न प्रकार है-

Function function_name (parameter1,parameter2,….)
{
जावास्क्रिप्ट के कथन
}

जहां function_name उस Function का नाम है और कोष्टक में उसके Parameters की सूची है जिसके विभिन्न Parameters को कॉमा (,) द्वारा अलग अलग किया जाता है| Function का नाम किसी अक्षर से प्रारंभ होना चाहिए उसमें अंडरस्कोर (_) का प्रयोग किया जा सकता है Function का नाम केस सेंसिटिव होता है अर्थात उसमें छोटे और बड़े अक्षरों में अंतर किया जाता है इसलिए उसको कॉल करते समय उसका नाम ठीक वैसा ही दिया जाना चाहिए जैसा उसकी परिभाषा में दिया गया है|

किसी Function को घोषित करने से ही उसका पालन नहीं किया जाता जब Function को कॉल किया जाता है तभी उसके कथनों का पालन किया जाता है|

जावास्क्रिप्ट में Objects

जावास्क्रिप्ट में लगभग सभी Elements Objects कहलाते हैं |ऑब्जेक्ट किसी स्पेशल टाइप के Data या Variable होते है जिसकी Property और Method होती है | जावास्क्रिप्ट में हम जो भी इन्सर्ट करते हैं जैसे – Array, Functions, Date, Maths आदि सभी Objects के अंतर्गत आते हैं | हम हमेशा जावास्क्रिप्ट में स्क्रिप्टिंग करते समय OBJECTS का इस्तेमाल करते ही हैं|

साधारण अर्थो में जावास्क्रिप्ट में सब कुछ ऑब्जेक्ट होता है। एक Object कुछ Properties और Methods को एक जगह Bind करने के लिए प्रयोग किया जाता है। ये Properties और Methods किसी Single Entity को Represent करती है। जावास्क्रिप्ट आपको built in objects provide करती है। जैसे की JavaScript में Strings Objects है। आप कोई भी String Variable Create करके उससे Related Properties और Methods प्रयोग कर सकते है।

जैसे – document.write(“Hello World”) ;

यहाँ document एक object है और write() एक method उपरोक्त उदाहरण के आधार पर हम यह कह सकते हैं कि मेथड को एक्सेस करने के लिए ऑब्जेक्ट का प्रयोग किया जाता है |

Creating Objects in JavaScript

अधिकतर प्रोग्रामिंग लैंग्वेज में ऑब्जेक्ट्स एक ही तरीके से क्रिएट किये जाते है। जावास्क्रिप्ट में आप ऑब्जेक्ट्स तीन तरह से क्रिएट कर सकते है।

- Creating Objects in Single Statement
- Creating Objects with New Keyword
- Creating Objects Using Constructor

First Method

Creating Objects in Single Statement

पहले तरीके में आप Object का नाम लिखते है और Curly Brackets में Properties और उनकी Values को Colon से Separate करते है। आप चाहे तो JavaScript में Objects किसी Variables की तरह एक Single Statement में Create कर सकते है।

Example :
var obj1 = {Name:”yourName”,Age:24};

इस उदाहरण में Object किसी Variable की तरह Create किया गया है। Curly brackets में Properties के नाम और Values को Colon से Separate किया गया है। और हर Pair को Comma (,) से Separate किया गया है। किसी भी Property की Value को Access Object के नाम के आगे dot (.) Operator लगाकर और फिर Property का नाम लिखकर Create कर सकते है।

document.write(obj1.Name);

Single Statement में Object Create करना सबसे आसान तरीका है।

Second Method

Creating Objects with New Keyword

दूसरे तरीके में New Keyword के द्वारा एक Object instance create किया जा सकता है।इस तरीके में पहले New Keyword के साथ Object Create कर लिया जाता है। बाद में Object के नाम के आगे dot (.) Operator लगाकर Property का नाम और Values Define की जाती है। अधिकतर Programming languages में इसी तरीके से Objects Create किये जाते है। इसका उदाहरण नीचे दिया जा रहा है।

```
<html>
<head>
<title>JavaScript New Keyword Demo</title>
</head>
<body>

<script type="text/javascript">
// Creating object
var obj1 = new Object();
// Creating object variables

obj1.Name="computerhindinotes";
obj1.Age=29;
// Printing object variables
document.write("Name is "+Name+"and age is "+Age);
</script>

</body>
</html>
```

Output

Third Method

Creating Objects Using Constructor

तीसरे तरीके में object constructor प्रयोग किया जाता है। इन सभी तरीकों के बारे में निचे detail से दिया जा रहा है। JavaScript में Object Create करने का ये तरीका थोड़ा जटिल (Complicated) है। इस तरीके में एक Function Create किया जाता है जो Properties की Values को

Argument की तरह लेता है। ये Function C language के किसी Structure या C++ की Classes की तरह काम करता है और Object Create होने के लिए एक Structure Provide करता है।

इस फंक्शन का नाम ऑब्जेक्ट के नाम जैसा ही होता है। जब New Keyword के साथ Object Create किया जाता है तो साथ ही Properties की Values भी Pass की जाती है। ये Values this keyword के द्वारा Original Properties को Apply की जाती है। इसका उदाहरण नीचे दिया जा रहा है।

```
<html>
<head>
<title>JavaScript Objects Demo</title>

<script type="text/javascript">
function Employee(name,age)

{
this.Name = name
this.Age = age;
}
</script>

</head>
<body>

<script type="text/javascript">
// Creating object using constructor
var obj1 = new Employee("Computerhindinotes",20);
document.write("Name is :"+obj1.Name+"<br>");
document.write("Age is :"+obj1.Age);
</script>

</body>
</html>
```

Output

Object Properties

- किसी भी जावास्क्रिप्ट ऑब्जेक्ट का सबसे महत्वपूर्ण हिस्सा Properties हैं। Properties जावास्क्रिप्ट ऑब्जेक्ट से जुड़ीं वैल्यू हैं। जावास्क्रिप्ट ऑब्जेक्ट Unordered Properties का एक संग्रह है।
- Properties आमतौर पर change, add और delete किये जा सकते हैं, लेकिन कुछ केवल Read किये जाते हैं।
- जावास्क्रिप्ट ऑब्जेक्ट्स में Named Value को Properties कहा जाता हैं।

Property	Value
firstName	Rohit

lastName	Sharma
age	50
eyeColor	blue

Add New Properties in JavaScript

- आप किसी मौजूदा Object को केवल एक Value देकर नई Properties जोड़ सकते हैं।
- मान लें कि person object पहले से मौजूद है – आप फिर उसे नई Properties दे सकते हैं:

Example – person.nationality = "English";

Delete Properties in JavaScript

डिलीट कीवर्ड किसी प्रॉपर्टी को किसी ऑब्जेक्ट से हटा देता है:

Example – var person = {firstName:"Rohit", lastName:"Sharma", age:50, eyeColor:"blue"};

delete person.age; // or delete person["age"];

- डिलीट कीवर्ड प्रॉपर्टी की वैल्यू और प्रॉपर्टी दोनों को ही डिलीट कर देता है।
- हटाए जाने के बाद, प्रॉपर्टी को दोबारा वापस जोड़ने से पहले इसका इस्तेमाल नहीं किया जा सकता है।
- हटाए गए ऑपरेटर को ऑब्जेक्ट प्रॉपर्टी पर उपयोग करने के लिए डिज़ाइन किया गया है। इसका वेरिएबल या फंक्शन पर कोई प्रभाव नहीं पड़ता है।
- हटाए गए ऑपरेटर को पूर्वनिर्धारित जावास्क्रिप्ट ऑब्जेक्ट प्रॉपर्टी पर उपयोग नहीं किया जाना चाहिए। यह आपके एप्लिकेशन को क्रैश कर सकता है।

Property Attributes in JavaScript

- सभी प्रॉपर्टीज का एक नाम है। इसके अलावा उनकी एक वैल्यू भी है।
- वैल्यू, प्रॉपर्टी की attributes में से एक है।
- अन्य attributes हैं: enumerable, configurable, और writable|
- ये attributes परिभाषित करती हैं कि प्रॉपर्टी तक कैसे पहुँचा जा सकता है (क्या यह readable है ?, क्या यह writable है?)
- जावास्क्रिप्ट में, सभी attributes को पढ़ा जा सकता है, लेकिन केवल वैल्यू एट्रिब्यूट को बदला जा सकता है

Prototype Properties in JavaScript

जावास्क्रिप्ट ऑब्जेक्ट्स उनके प्रोटोटाइप की प्रॉपर्टी को प्राप्त करते हैं। डिलीट कीवर्ड inherited properties को नहीं हटाता है, लेकिन यदि आप एक प्रोटोटाइप प्रॉपर्टी हटाते हैं, तो यह प्रोटोटाइप से object inherited को प्रभावित करेगा।

JavaScript Object Methods

Example –

```
var person = {
firstName: “Rohit”,
lastName : “Sharma”,
id     : 4488,
fullName : function() {
return this.firstName + ” ” + this.lastName;
}.
};
```

फ़ंक्शन की परिभाषा में, this फ़ंक्शन के “owner” को संदर्भित करता है।

उपरोक्त उदाहरण में, this वह पर्सनल ऑब्जेक्ट है जो Fullname फ़ंक्शन का “owns” है।

दूसरे शब्दों में, this.firstName का अर्थ है firstName की प्रॉपर्टी this object है|

JavaScript Methods

- जावास्क्रिप्ट मेथड्स ऐसी क्रियाएं हैं जो ऑब्जेक्ट्स पर की जा सकती हैं।
- जावास्क्रिप्ट मेथड एक प्रॉपर्टी है जिसमें फ़ंक्शन परिभाषा होती है।

Property	Value
firstName	Rohit
lastName	Sharma
age	50
eyeColor	blue
fullName	function() {return this.firstName + ” ” + this.lastName;}

Using Built-In Methods

किसी टेक्स्ट को अपरकेस में कनवर्ट करने के लिए स्ट्रिंग ऑब्जेक्ट के toUpperCase() मेथड का उपयोग करता है:

```
var message = “Hello world!”;
var x = message.toUpperCase();
```

उपरोक्त कोड के Execution के बाद x का मान होगा:
HELLO WORLD!

Adding a Method to an Object

ऑब्जेक्ट में एक नई मेथड जोड़ना आसान है:

Example –

```
person.name = function () {
```

```
return this.firstName + " " + this.lastName;
};
```

Document Object Model क्या है ?

Document Object Model ,W3C (World Wide Web) का एक स्टैंडर्ड है | यह किसी डॉक्यूमेंट को एक्सेस करने के लिए स्टैंडर्ड Define करता है| अन्य शब्दों में डॉक्यूमेंटऑब्जेक्ट मॉडल किसी डॉक्यूमेंट के Structure, Style तथा Content को Dynamically Access और Update करने के लिए Programs और Scripts के लिए प्लेटफार्म उपलब्ध कराता है |

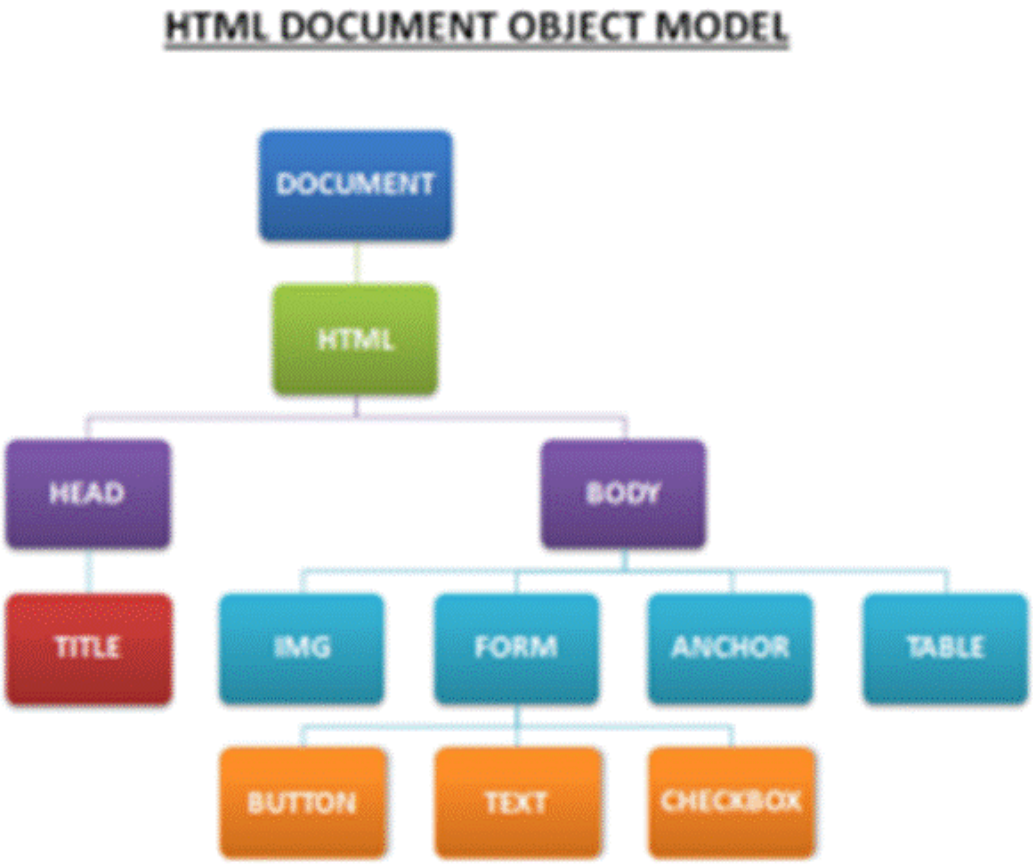
जब कोई वेब पेज लोड होता है तो वेब ब्राउजर उस पेज का Document Object Model बना देता है | JavaScript Document Object Model (DOM) का सबसे बड़ी विशेषता यह है की इसकी मदद से हम HTML tags को Dynamically Access कर सकते हैं और जरूरत पड़ने पर उनमें परिवर्तन (Changes)भी कर सकते हैं ।

किसी HTML पेज को डायनामिक)Dynamic) बनाने के लिए Document Object Model में जावास्क्रिप्ट की महत्वपूर्ण भूमिका होती है क्योकि –

- Javascript webpage के सभी tags को access और change कर सकता है |
- Javascript webpage के सभी attributes को access और change कर सकता है |
- Javascript webpage के पुराने html elements जैसे – Tags , Attributes को remove कर सकता है |
- Javascript द्वारा webpage में नए tags और attributes को add किया जा सकता है।
- Javascript webpage की सभी CSS styles को change कर सकता है |
- Javascript द्वारा नए events create किये जा सकते है।

Document Object Model को हम एक Object tree के द्वारा समझ सकते हैं –

- Document मतलब Webpage |
- Object मतलब विभिन्न Elements जैसे Tags, attributes इत्यादि |
- Model मतलब बनावट या संरचना |



DOM Accessible Basic HTML Elements

कुछ बेसिक HTML Elements जिन्हें आप DOM के द्वारा हैंडल कर सकते है।

- Anchor
- Form
- Textbox
- Textarea
- Checkbox
- Radio
- Select
- Option
- Reset
- Button
- Link

एक ब्राउज़र सभी Tags को Nested list की तरह देखता है। जैसा की नीचे दी गयी लिस्ट में दिया गया है। HTML Tag Top पर होगा और बाकि Tags उसके Nested Order में होंगे।

HTML
———-HEAD
————TITLE
—-BODY
————-P
————H1

————FORM
————TABLE

इन सभी Tags को Access करने के लिए आपको पहले उनके Parent Tags को Access करना होता है। Parent Tags को Node भी कहते है। इन Tags को Access करने के लिए आप Document Object को यूज़ करते है। जैसे की Form को Access करने के लिए आप Document.Form Statement यूज़ कर सकते है।

यदि आप Form में किसी Field की Value Access करना चाहते है तो उस Field का नाम लिखकर उसके आगे Dot Operator लगाकर Value लिख देंगे।जैसे की आप किसी Text-Box की Value Access करना चाहते है जिसका नाम firstname है तो ऐसा आप इस तरह कर सकते है।

var firstName = document.form.firstName.value;

JavaScript Document Object Model (DOM) का सबसे बड़ा फीचर यह है की इसकी मदद से आप सभी Tags को Dynamically Access कर सकते है और Situation के According उनमें Changes कर सकते है।

Functions of JavaScript DOM

getElementById(id)
ये Method एक Element Return करता है। इस मेथड में Id argument की तरह Pass की जाती है। वो Id जिस Element की होती है वो Element ये Method Return कर देता है।

getElementsByName(name)
आप बहुत से Name पास कर सकते है। ये Name जिन Elements से Match होते है वो Elements ये Method Return करता है।

getElementsByTagName(tagName)
एक Tag Name Pass किया जाता है। ये Method उस Tag के नाम वाले सभी Tags को Return करता है।

getElementsByClassName(className)
एक Class Name Pass किया जाता है। ये Method Class Name वाले सभी Tags Return करता है।

What is Array in JavaScript

Array जावास्क्रिप्ट की ऐसी वस्तुएं (objects) हैं, जो मानो की किसी श्रंखला को स्टोर कर सकते हैं ये मान Array में इंडेक्स किए गए स्थानों पर स्टोर किए जाते हैं किसी Array में तत्वों की कुल संख्या को उस Array की लंबाई कहा जाता है किसी Array के किसी विशेष तत्व को उस Array के नाम और बड़े कोष्टक में उसके इंडेक्स के मान (index value) को रखकर उपयोग किया जा सकता है Array तत्वों के Index शून्य(0) से प्रारंभ होते हैं और अंतिम इंडेक्स उसे array की लंबाई से एक कम संख्या के बराबर होता है|

उदाहरण के लिए, हम 10 संख्याओं का एक Array numb उपयोग कर सकते हैं ऐसी स्थिति में हम कहेंगे कि इस Array की लंबाई 10 है इसमें पहली संख्या का इंडेक्स 0 और अंतिम अर्थात दसवीं संख्या का इंडेक्स 9 होगा हम इस array के पांचवें तत्व का संदर्भ numb[4] लिख कर देंगे| इसी प्रकार अन्य तत्वों के बारे में समझा जा सकता है|

JavaScript Array Video Tutorial in Hindi

अगर आप ऐरे क्या होता है और इसे स्क्रिप्ट बनाने में किस प्रकार से प्रयोग किया जाता है सीखना चाहते हैं तो आप हमारे द्वारा बनाया गया ये विडियो जरुर देखिये, इस विडियो में आप निम्नलिखित कांसेप्ट समझ पाएंगे

- What is JavaScript array
- How to Declare array

- how to initialize array
- Array length property

Declaring Array (Array घोषित करना)

जावास्क्रिप्ट में किसी Array का उपयोग करने से पहले उसको घोषित करना अनिवार्य है Array को हम निम्नलिखित में से किसी भी विधि से घोषित कर सकते हैं-

Array_name=new array [array_length];
Array_name=new array[ ] ;

जहां array_name उस Array का नाम है और array_length उसकी लंबाई है पहली विधि में Array की लंबाई स्पष्ट घोषित की गई है जबकि दूसरी विधि में लंबाई घोषित नहीं है वैसे जावास्क्रिप्ट आवश्यकता होने पर किसी भी Array की लंबाई अपने आप बढ़ा लेता है भले ही उसे निश्चित लंबाई के साथ घोषित किया गया हो|

उदाहरण के लिए, ऊपर बताए Array numb को निम्न प्रकार घोषित कर सकते हैं

Numb = new array [10];

अब यदि आप किसी व्यंजक में numb[15] देते हैं, तो जावास्क्रिप्ट उस Array की लंबाई को उसी के अनुसार 16 तक बढ़ा देगा|

Dense array

Dense array एक ऐसा Array होता है जिसका निर्माण उसके सभी तत्वों को कोई विशेष मान देकर किया जाता है, दूसरे शब्दों में उसके तत्वों का प्रारंभिक मान उसकी घोषणा के समय ही रख दिया जाता है अन्य सभी बातों में Dense array का उपयोग साधारण Array की तरह ही किया जाता है|
किसी Dense array के निर्माण या घोषणा का सामान्य रूप निम्न प्रकार है

Array_name=new array[value0, value1,……,value n];

जहां value0, value1…. आदि कोई अच्छा है यहां क्योंकि इंडेक्स 0 से प्रारंभ होकर n तक है इसलिए इस Array में तत्वों की संख्या n+1 होगी|

जावास्क्रिप्ट इवेंट्स)JavaScript Event)

जावास्क्रिप्ट इवेंट की मदद से आप अपने वेबपेज को इस तरह डिज़ाइन कर सकते है की आपका वेबपेज, यूज़र की गतिविधि (activity) को प्रतिक्रिया (respond) कर सके और उसके अनुसार जरुरी बदलाव कर सके। इवेंट का प्रयोग करने से आपका वेबपेज गतिशील और इंटरेक्टिव (dynamic and interactive) बन जाता है।

इवेंट्स यूज़र के एक्शन पर क्रियान्वित होते है)Execute Script on User Action)

अधिकतर इवेंट्स यूज़र के द्वारा उत्त्पन्न (generated) होते है। सामान्यतः हम जो वेबपेज बनाते है वो एक बार में ही पूरी तरह लोड हो जाते है। मतलब वेबपेज में ऐसा कोई हिस्सा नहीं होता जो लोड होना बाकी हो। लेकिन ऐसा हो सकता है की आप कुछ जानकारी (information) यूज़र के हिसाब से लोड करना चाहते है। जैसे की यूज़र किसी लिंक पर क्लिक करे या मेन्यू में से कोई आइटम सेलेक्ट करे तो उसके अनुसार पेज में बदलाव आ जाये। ऐसा आप जावास्क्रिप्ट इवेंट्स की मदद से कर सकते है।

हर इवेंट को आप HTML में एट्रिब्यूट की तरह प्रयोग कर सकते है और उस टैग से सम्बंधित सभी इवेंट हैंडल कर सकते है। जैसे की बटन से सम्बंधित onClick()event प्रयोग कर सकते है। जैसे ही यूज़र बटन पर क्लिक करें तो कोई एक्शन ले सकते है।

उदाहरण के लिए यदि आप वेबपेज में कोई अलर्ट बॉक्स (Alert-box)जोड़ते है तो वह वेबपेज के लोड होते ही प्रदर्शित हो जाता है। लेकिन आप चाहते है की वो अलर्ट बॉक्स तब प्रदर्शित हो जब यूज़र किसी बटन या लिंक पर क्लिक करे तो इसके लिए आप जावास्क्रिप्ट इवेंट्स को प्रयोग करेंगे।

JavaScript Event Categories

जावास्क्रिप्ट इवेंट्स को अलग-अलग केटेगरी में बांटा गया है। जैसे की माउस से संबंधित इवेंट माउस इवेंट की श्रेणी में आते है और फॉर्म से संबंधित इवेंट्स फॉर्म इवेंट्स की श्रेणी में आते है। जवास्क्रिप्ट्स इवेंट्स की सामान्यतः पांच श्रेणी होती है।

Mouse Events

onclick, onmouseover, onmouseout, ondblclick. onmousedown,

Form Events

onblur, onchange, onfocus, onselect, onsubmit, onreset

Keyboard Events

onkeypress, onkeydown, onkeyup

Frame Events

onload, onunload, onresize, onscroll, onunload, onerror

Media Events

onplay, onpause, onerror, onprogress, onplaying

Common JavaScript Events

Javascript के कुछ Common Events जिनका प्रयोग बहुत ज्यादा होता है।

onclick= “ ”

ये एक Mouse Event है। आप इसे Clickable Components (Link, Button) के साथ प्रयोग कर सकते है। Component पर Click होते ही Define किया गया Javascript Function Call हो जाता है।

onfocus=” ”

ये एक Form Event है। इससे Form को जैसे ही Focus मिलता है Script Execute हो जाती है।

Onblur=” ”

ये भी एक Form Event है जो जैसे ही Form से Focus हटता है Script को Execute करता है।

onchange=” ”

जैसे ही Component में कोई परिवर्तन होता है ये Event Script को Execute कर देता है। जैसे की लिस्ट में से कोई दूसरा Item Select किया जाये।

onSelect=” “

जब यूज़र Text को Select करता है तो ये Event Define किये गए Function को Call करता है।

onmouseover=” ”

जब Component पर Mouse को ले जाया जाता है Script Execute हो जाती है।

onmouseout=” ”

जैसे ही Component से Mouse को हटाया जाता है ये Event Script को Execute कर देता है।

onload=” ”

जैसे ही Page की Loading Complete होती है Script Execute हो जाती है।

onunload=” ”

जैसे ही Browser में कोई नयी Window खोली जाती है ये Event Script को Execute कर देता है।

onsubmit=” ”

जैसे ही Form को Submit किया जाता है ये Javascript Event Defined Function को Call कर देता है।

---

ntroduction of JavaScript Popup

पॉपअप (Popup)एक छोटी दूसरी विंडो होती है जो current window के अंदर ही जावास्क्रिप्ट द्वारा उत्पन्न (generate) होती है। इन्हें popup boxes भी कहते है। Popup के द्वारा 3 तरह के काम किये जा सकते है।

- आप कोई भी अलर्ट प्रदर्शित कर सकते है।
- कोई भी कार्य करने से पहले यूजर से पुष्टि)confirm) कर सकते है।
- इनपुट की आवश्यकता होने पर यूज़र से इनपुट ले सकते है।

Types of Popup in JavaScript

JavaScript में पॉपअप उनके काम के अनुसार 3 तरह के होते है।

- Alert Dialog Box
- Confirm Dialog Box
- Prompt Dialog Box

Learn JavaScript in Hindi (Popup Boxes, Alert Box, Prompt Box, Confirm Box)

अगर आप जावास्क्रिप्ट में पॉपअप बॉक्स जैसे अलर्ट बॉक्स, प्रोम्पट बॉक्स और कन्फर्म बॉक्स कैसे प्रयोग करना है सीखना चाहते हैं तो नीचे दिया गया विडियो जरुर देखिये|

Alert Dialog Box

Alert box का प्रयोग यूज़र को महत्पूर्ण मेसेज (important message) प्रदर्शित करने के लिए किया जाता है। जब आप चाहते है की यूज़र आपके मेसेज को जरूर पढे ऐसी परिस्थिति (situation) में आप alert box प्रयोग कर सकते है।

ये बॉक्स current window से focus हटा देता है और सिर्फ मेसेज प्रदर्शित होता है। साथ ही alert box में एक ok का button भी होता है जिस पर क्लिक करने से alert box हट जाता है। JavaScript में alert dialog generate करने के लिए आप alert function call करते है। इस फंक्शन में आप जो मेसेज प्रदर्शित करना चाहते है उसे आर्गुमेंट (argument) की तरह लिखते है।

Example1: Displaying Alert

```
<html>
<head>
<title>Alert box demo</title>
</head>
<body>

<script type="text/javascript">
alert("This is an alert box");
</script>
</body>
</html>
```

Output

Confirm Dialog Box

Confirm dialog box यूज़र से किसी टास्क (task) के बारे में पुष्टि (confirmation) लेने के लिए प्रयोग किया जाता है। ये एक छोटी सी विंडो होती है जिसमे Yes और No बटन होते है। जिसमें से Yes बटन यूज़र की सहमती दर्शाता है और No बटन प्रदर्शित करता है की यूज़र आगे बढना (proceed) नहीं चाहता है।

JavaScript में Confirm dialog box generate करने के लिए आप confirm method को call करते है। इस मेथड में आप एक message argument की तरह pass करते है, जो यूज़र से पुष्टि (confirmation) मांगता है। जैसे की आप यूज़र से पूछते है की आप ये पुस्तक खरीदना चाहते है?

Example2: Confirm user action

```
<html>
<head>
<title>Confirm dialog demo</title>
</head>
<body>
<script type="text/javascript">
```

```
// Setting confirm box
if(confirm("Do you want to buy this book?"))
{
// Print message when if user clicks OK
document.write("User wants to buy the book");
}
</script>
</body>
</html>
```

उपर दिये गए उदाहरण में यदि user confirm box में Yes सेलेक्ट करता है तो if statement में दिया गया message web page पर प्रिंट हो जाता है।

OUTPUT

यदि यूजर OK पर click करता है तो message show होता है नहीं तो कुछ भी show नहीं होता है।

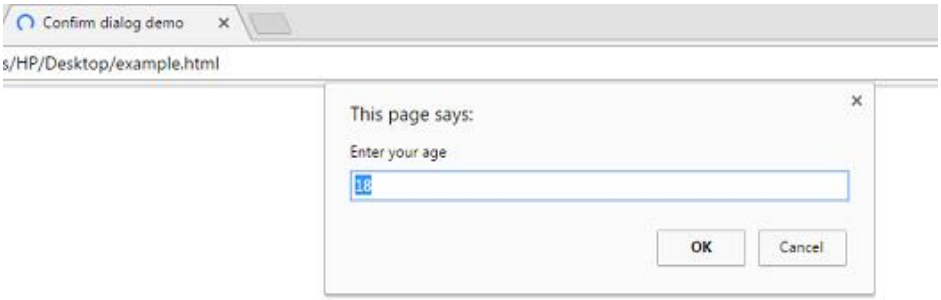
Prompt Dialog Box

यदि आप यूजर से कोई इनपुट लेना चाहते है तो आप prompt dialog box यूज़ कर सकते है। Prompt dialog box में एक text-box होता है और एक ok button होता है। Text-box में user value input करता है। Prompt box create करने के लिए आप prompt() method को call करते है। इस method में 2 parameters pass किये जाते है। पहला parameter वो label होता है जो text-box में क्या वैल्यू डालनी है ये प्रदर्शित करता है। दूसरे parameter में text-box की default value pass की जाती है।

Example3: Taking user input

```
<html>
<head>
<title>prompt box demo</title>
</head>
<body>
<script type="text/javascript">
// Creating a prompt dialog
var age = prompt("Enter your age", 18);
document.write("Age is :",age);
</script>
</body>
</html>
```

Output



यूज़र जो value input के रूप में देगा OK पर क्लिक करने पर उसे appropriate message के साथ प्रिंट किया जायेगा।