بسم الله الرحمن الرحيم

مقدمة نحو چافاسكريبت
Introduction to JavaScript

مـــن W3Schools

www.w3schools.com

ترجمــة وإعـــداد

عـــبـــدالله مـــحـــمـــد الـــغـــامـــدي

apc1424@yahoo.com

التوزيع مجاني *

---

* مع الإشارة لـ W3Schools كمصدر

جـــدول المـــحـــتـــويـــــات

مـــقـــدمـــــة

تستخدم جافاسكريبت في ملايين الصفحات على الإنترنت لتحسين التصميم، أو تصميم نماذج الإدخال، أو غير ذلك. تم تطوير جافاسكريبت بواسطة شركة نيتســكيب **Netscape** وهي واحدة من أشهر اللغات النصية على الإنترنت.

تعمل جافاسكريبت على معظم المتصفحات الرئيسية الموجودة اليوم مثل **Navigator** من شركة نيتسكيب و **Internet Explorer** من شركة مايكروسوفــت.

ما الذي يجب أن تعرفه الآن؟!

قبل الاستمرار، عليك الإلمام بالمفاهيم الأساسية في:

- صفحات الويب WWW ، لغة النص المترابط HTML ، وبعض الأساسيات في بناء صفحات الإنترنت.

إذا رغبت في تعلم المزيد عن المواضيع السابقة، يمكنك زيارة www.w3schools.com .

مـــا هـــــي جافاسكريبت؟

- صممت جافاسكريبت لإضافة التفاعلية لصفحات HTML.
- جافاسكريبت هي لغة برمجة نصية – لغة البرمجة النصية هي لغة خفيفة.
- جافاسكريبت هي عبارة عن أسطر قابلة للتنفيذ.
- جافاسكريبت يتم تضمينها –عادة- مباشرة في صفحات HTML.
- جافاسكريبت هي لغة مفسرة.
- يمكن لأي شخص استخدام جافاسكريبت بدون شراء ترخيص استخدام.
- جافاسكريبت مدعومة من غالبية متصفحات الإنترنت الشهيرة.

هل جــافا Java هي نفسهــا جافاسكريبت ؟

لا!

جافا و جافاسكريبت هما لغتين مختلفتين تماماً...

جافا (طورت بواسطة شركة صن مايكروسيستيمز) هي لغة برمجة قوية ومعقدة –وفي نفس التصنيف مع لغات برمجة مثل C و C++ .

ما الذي يمكن لــ جافاســكريبت أن تفــعله؟

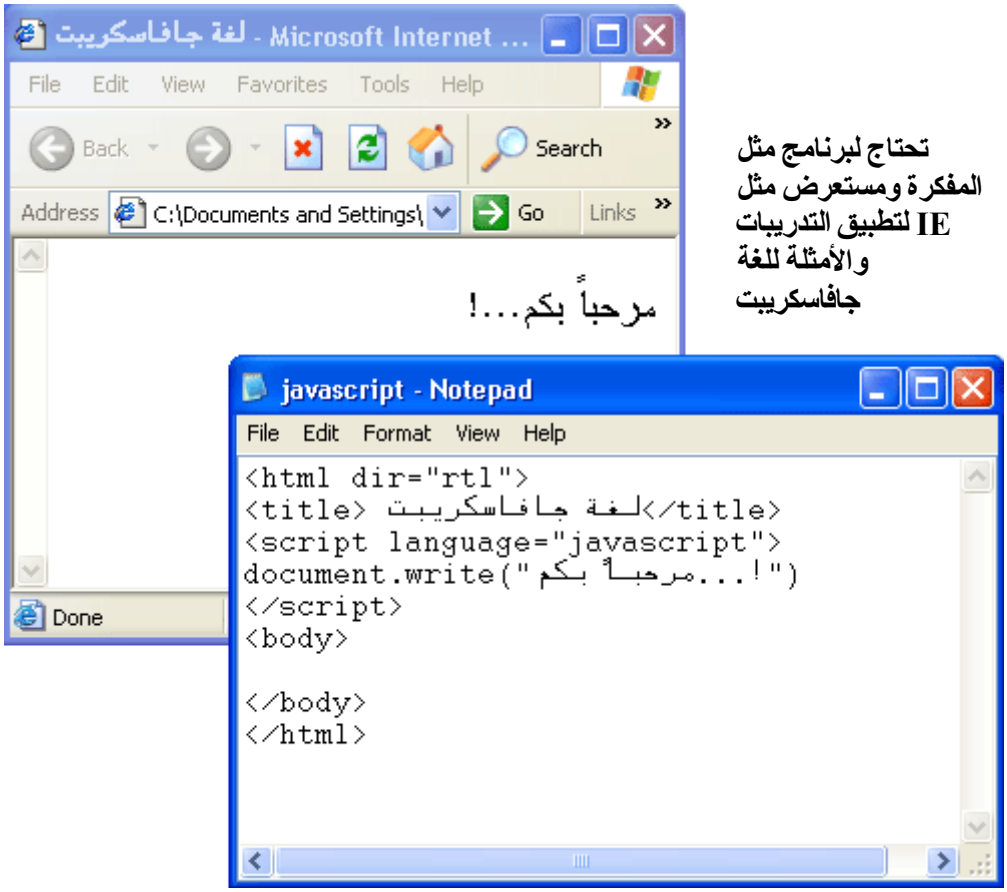
- **جافاسكريبت تعطي مؤلفي HTML أداة برمجية** –مؤلفي HTML ليسوا مبرمجين، ولكن جافاسكريبت هي لغة برمجة بسيطة التركيب! فبإمكان أي شخص أن يضع نصاً صغيراً لصفحات HTML الخاصة به.
- **يمكن لـ جافاسكريبت أن تضع نصاً متغيراً في صفحات HTML** – جملة جافاسكريبت مثل هذه document.write("<h1>"+name+"</h1>") يمكن أن تكتب نصاً متغيراً لصفحة HTML.
- **يمكن لـ جافاسكريبت أن تستجيب للأحداث** – يمكن لـ جافاسكريبت أن تنفذ أمراً عندما يحدث شيء ما، مثل عند انتهاء تحميل الصفحة أو عندما يضغط المستخدم على عنصر HTML.
- **جافاسكريبت يمكنها أن تتعرف على أو تكتب عناصر HTML** –يمكن لـ جافاسكريبت أن تكتب وتغير محتوى صفحات HTML.
- **يمكن أن تستخدم جافاسكريبت للتحقق من البيانات** –يمكن لـ جافاسكريبت أن تستخدم للتحقق من البيانات المدخلة في النماذج –في مواقع الإنترنت- قبل اعتمادها، وذلك يوفر الكثير من وقت المعالجة بالنسبة للخوادم Servers.

ما هي البرامـــج التي أحتاجهـــا؟

لا تحتاج –في الحقيقة –سوى إلى محرر نصوص! وذلك لكتابة النص البرمجي (الكود) الخاص بـHTMLومن ثم تضمين جمل جافاسكريبت فيها...

إذا كنت تصمم صفحاتك ببرنامج مثل FrontPage فبإمكانك وضع جمل جافاسكريبت ضمن محرر HTML في البرنامج...

إذا رغبت في تطبيق التمرينات في هذا الكتاب، استخدم برنامج المفكرة Note Pad المرفق بنظام التشغيل، واحفظ ملفات التدريبات بالامتداد file.htm أو file.html ، ويمكنك بعد ذلك استعراضها بمتصفح الإنترنت لديك، سواءً IE أو Navigator.



تحتاج لبرنامج مثل المفكرة ومستعرض مثل IE لتطبيق التدريبات والأمثلة للغة جافاسكريبت

كيف أضع نص جافاسكريبت داخل صفحـــة HTML؟

بالشكل التالي:

```
<html>
<body>
<script type="text/javascript">
document.write("Hello World!")
</script>
</body>
</html>
```

لاحظ: علامات HTML باللون الأسود – نص جافاسكريبت باللون الأزرق – دالة جافاسكريبت باللون الأحمر

بعد حفظ الملف السابق باسم ex1.html مثلا وفتحه باستخدام متصفح إنترنت إكسبلورير ، سيظهر في الصفحة:

Hello World!

شرح المثال أعلاه:

لإدراج النص البرمجي في صفحة HTML ، نستخدم علامة <script>، ولتعريف لغة جافاسكريبت كلغة برمجة نصية نكتب:

```
<script type="text/javascript">
```

بعد ذلك يبدأ نص جافاسكريبت: أمر جافاسكريبت لكتابة النص على الصفحة هو document.write

```
document.write("Hello World!")
```

بعد ذلك يجب إقفال العلامة <script>

```
</script>
```

إنـــهاء الجمل بفاصلة منقوطـــة؟

مع لغات البرمجة التقليدية مثل C++ وجافا، كل جملة (تعليمة) برمجية تنتهي بفاصلة منقوطة " ; "..
معظم المبرمجين استمروا في هذه العادة عند الكتابة بلغة جافاسكريبت؛ لكن عموماً، كتابة الفاصلة المنقوطة أمر **اختياري!** على الرغم من ذلك؛ الفاصلة المنقوطة مطلوبة عند كتابة أكثر من جملة برمجية في سطر واحد.

كيفيـــة التعـــامل مع المتصفحات القديمـــة

المتصفحات التي لا تدعم جافاسكريبت، ستظهر النص البرمجي كمحتوى صفحة – بمعنى أنه سيظهر كما كتبته ضمن HTML - . لمنع ظهور النص البرمجي في المتصفحات التي لا تدعم جافاسكريبت؛ نستخدم علامة التعليقات في HTML كما يلي:

```
<script type="text/javascript">
<!--
    some statements
//-->
</script>
```

الشرطتين المائلتين للخلف (//) في نهاية سطر التعليقات هي رمز التعليقات في جافاسكريبت، وتمنع مترجم جافاسكريبت من ترجمة السطر البرمجي.
ملاحظة: لا يمكنك وضع // في بداية سطر التعليقات (<!--//) لأن المتصفحات القديمة سوف تعرض النص.

جافاسكريبت: ... أين أضع النـــص البرمجي؟

النصوص البرمجية في القسم body ستنفذ عندما يتم تحميل الصفحة... (يتم التنفيذ أولاً بأول).
النصوص البرمجية في القسم head ستنفذ عندما يتم استدعاؤها...

في القسم HEAD:
نضع النصوص البرمجية التي تحتوي على دوال functions في القسم head من الصفحة. بذلك نضمن أن النصوص البرمجية قد تم تحميلها قبل استدعاء الدوال.
في القسم BODY:
يتم تنفيذ النصوص البرمجية الموضوعة في القسم body.
النص البرمجي الخارجي:
كيفية الوصول للنص البرمجي الموجود في ملف آخر External File.

مكـــان وضـــع جافاسكريبت

النصوص البرمجية في الصفحة سيتم تنفيذها مباشرة أثناء تحميلها في المتصفح. وهذا ليس ما نتمناه دائماً؛ أحياناً نريد أن ينفذ النص عند تحميل الصفحة وأحياناً أخرى نريد أن يتم تنفيذ النص عندما يضغط المستخدم على عنصر ما.
النصوص البرمجية في القسم head : النصوص البرمجية التي تحتاج لاستدعاء أو التي تنفذ عند ما ينقر المستخدم على عنصر ما تكون في القسم head ، وبذلك تكون متأكداً من أن النص البرمجي سيتم تحميله قبل استخدامه.

```
<html>
<head>
<script type="text/javascript">
     some statements
</script>
</head>
```

النصوص البرمجية في القسم body: هنا يتم وضع النصوص التي تنفذ عند تحميل الصفحة (أولاً بأول) وهي التي ستكون محتوى الصفحة.

```
<html>
<head>
</head>
<body>
<script type="text/javascript">
     some statements
</script>
</body>
```

النصوص البرمجية في كلا القسمين head و body: يمكنك وضع عدد لا نهائي من النصوص البرمجية في القسمين head و body.

```
<html>
<head>
<script type="text/javascript">
     some statements
</script>
</head>
<body>
<script type="text/javascript">
     some statements
</script>
</body>
```

كيفية تنفيذ نص جافاسكريبت موجود في ملف خارجي

أحياناً تحتاج لتنفيذ نص جافاسكريبت ضمن أكثر من صفحة. بدون كتابة النص كل مرة في كل صفحة.
لتبسيط ذلك، يمكنك كتابة نص جافاسكريبت فيملف خارجي وذلك بحفظه بالامتداد js. (مثال: file.js)
كأن تكتب النص البرمجي التالي:

```
document.write("This script is external")
```

واحفظه بالاسم example.js .

ملاحظة: لا يمكن أن يحتوي النص البرمجي الخارجي على علامة <script>.

الآن ، يمكنك استدعاء النص البرمجي المحفوظ في الملف الخارجي من خلال صفة "scr" من أي مكان في صفحة HTML:

```
<html>
<head>
</head>
<body>
<script src="example.js"></script>
</body>
</html>
```

تذكر أن تضع النص البرمجي في المكان الصحيح الذي أردت أن تكتبه فيه.

المتغـــيرات

المتغير هو "حاوية" أو مكان يتم تخزين البيانات فيه. قيمة المتغير (البيانات المخزنة فيه) يمكن أن تتغير خلال النص البرمجي. يمكنك الإشارة للمتغير بواسطة اسمه وذلك لمشاهدة قيمته أو تغيير هذه القيمة.

ضوابط لأسماء المتغيرات:

- اسم المتغير حساس لحالة الحرف case sensitive ، اسم المتغير Book يختلف عن book يختلف عن bOoK وهكذا...
- يجب أن يبدأ الاسم بحرف أو شرطة سفلية _ underscore .

الإعلان عن المتغيرات

يمكنك إنشاء متغير من خلال جملة var :

```
var strname = some value
```

كما يمكنك إنشاء المتغيرات بدون جملة var:

```
strname = some value
```

تعيين القيم للمتغيرات

يمكنك تعيين قيمة للمتغير بهذه الطريقة:

```
var strname = "Ahmed"
```

أو بهذه الطريقة:

```
strname = "Ahmed"
```

اسم المتغير في الجانب الأيسر من التعبير ، والقيمة التي تود تعيينها للمتغير تقع في اليمين. (بعد علامة =)، في المثال أعلاه؛ اسم المتغير strname ويحمل القيمة Ahmed.

مـدة حيـاة المتغيرات

عندما تعلن عن متغير داخل دالة، فإن المتغير يمكن الوصول إليه داخل هذه الدالة فقط، وعندما تخرج من هذه الدالة فإنه لا وجود لهذا المتغير تسمى هذه المتغيرات بالمتغيرات المحلية Local Variables.
يمكنك إنشاء متغيرات محلية بنفس الاسم في دوال مختلفة؛ لأن كل واحد منها سيكون معرفاً داخل الدالة التي أنشئ فيها.
إذا أنشأت متغير خارج الدالة؛ فإن كل الدوال في الصفحة قادرة على الوصول إليه. تسمى هذه المتغيرات بالمتغيرات العامة Global Variables ، وتبدأ مدة حياة المتغير العام منذ الإعلان عنه وحتى نهاية الصفحة.

المعامـــــلات

المعاملات أو العمليات الحسابية و المنطقية تستخدم لإجراء الحسابات على قيم المتغيرات.
توجد خمسة أنواع من المعاملات كالتالي:

المعاملات الرياضية

المعـامل	الـوصـــــف	مـثـــال	النتيجة
+	الجمع	x=2 x+2	4
-	الطرح	x=2 5-x	3
*	الضرب	x=4 x*5	20
/	القسمة	15/5 5/2	3 2.5
%	باقي القسمة	5%2 10%8 10%2	1 2 0
++	الزيـادة بمقدار...	x=5 x++	x=6
--	النقـص بمقدار ...	x=5 x--	x=4

معاملات التعيين

المعـامل	مـثـــال	المثال بصيغة أخرى...
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

معاملات المقارنة

المعامل	الوصف	مثال
==	يـســـاوي	8==5 تعيد (خاطئ)
=!	لا يســـاوي	8=!5 تعيد (صحيح)
>	أكـبـــر من	8>5 تعيد (خاطئ)
<	أصغـــر من	8<5 تعيد (صحيح)
>=	أكـبـــر من أو يســـاوي	8>=5 تعيد (خاطئ)
<=	أصغـــر من أو يســـاوي	8<=5 تعيد (صحيح)

المعاملات المنطقية

المعامل	الوصف	مثال
&&	and (و)	x=6 y=3 (x < 10 && y > 1) يعيد (صحيح)
||	or (أو)	x=6 y=3 (x==5 || y==5) يعيد (خاطئ)
!	not (نفي)	x=6 y=3 !(x==y) يعيد (صحيح)

معاملات السلسلة

في الغالب أن السلسلة هي عبارة عن نص، على سبيل المثال: "Hello World!".
للصق مصفوفتين نصيتين أو أكثر، نستخدم المعامل + .

```
txt1="What a very"
txt2="nice day!"
txt3=txt1+txt2
```

المتغير txt3 يحتوي الآن على "What a verynice day!".
لإضافة مسافة بين السلسلتين؛ أضف مسافة في التعبير الحسابي نفسه أو في إحدى السلسلتين.

```
txt1="What a very"
txt2="nice day!"
txt3=txt1+" "+txt2
أو
txt1="What a very "
txt2="nice day!"
txt3=txt1+txt2
```

المتغير txt3 يحتوي الآن على "What a very nice day!".

الـــــدوال

الدالة هي وحدة برمجية قابلة لإعادة الاستخدام (يمكن أن تستخدم أكثر من مرة خلال البرنامج) وتنفذ الدالة إما عند حدوث حدث event معين أو عندما يتم استدعاؤها.
بتعريف آخر: الدالة هي مجموعة من الجمل البرمجية تنفذ باستدعائها أو بحدوث حدث ما، والدالة قابلة لإعادة الاستخدام، بحيث يمكنك استخدامها أكثر من مرة داخل الصفحة، ويتم تعريف الدالة في بداية الملف (في القسم head تحديداً) ويمكن استدعاؤها فيما بعد.

هذه هي طريقة في جافاسكريبت لتحذير المستخدم:

```
alert("This is a message")
```

كيف تعرّف دالة؟

لإنشاء دالة يجب أن تحدد لها اسماً، ومعاملات argument وبعض الجمل البرمجية.

```
function myfunction(argument1,argument2,etc)
{
some statements
}
```

الدالة التي ليس لديها معاملات، لا بد من أن تحتوي أقواس:

```
function myfunction()
{
some statements
}
```

المعاملات هي متغيرات تستخدم في الدالة. قيم المتغيرات هي قيم تمرر للدالة عند استدعاؤها.
بوضعك للدالة في القسم head من الصفحة تكون متأكداً من أن الدوال قد تم تحميلها قبل استدعائها.
بعض الدوال تعيد قيمة إلى التعبير الذي تم استدعاؤها منه:

```
function result(a,b)
{
c=a+b
return c
}
```

كيفية استدعاء دالة

لا تنفذ الدالة قبل استدعائها.
يمكنك استدعاء الدالة مع معاملاتها:

```
myfunction(argument1,argument2,etc)
```

أو بدون معاملاتها:

```
myfunction()
```

جملة الإرجاع return

الدوال التي سوف تعيد نتيجة بعد تنفيذها يجب أن تستخدم جملة return . هذه الجملة تحدد القيمة التي يجب أن تعاد للمكان الذي استدعيت منه.

على سبيل المثال: هذه الدالة تعيد مجموع عددين:

```
function total(a,b)
{
result=a+b
return result
}
```

عندما تستدعي هذه الدالة، يجب أن ترسل معاملين (تمرير قيم للدالة) :

```
sum=total(2,3)
```

القيمة المعادة من الدالة هي 5 . وسوف تخزن في متغير يسمىsum.

الجمـــــل الشـــرطية في جافاسكريبت

تستخدم الجمل الشرطية في جافاسكريبت (وفي كافة لغات البرمجة) لإنجاز مهام مختلفة بناءاً على شروط أو قرارات معينة.

لدينا في جافاسكريبت ثلاثة أنواع من الجمل الشرطية:

- **جملة if** – استخدم هذه الجملة عندما تريد تنفيذ نصاً برمجياً إذا تحقق شرط معين (بمعنى أن تكون قيمته true).
- **جملة if...else** – استخدم هذه الجملة عدما تريد تنفيذ نصاً برمجياً من اثنين إذا تحقق شرط معين.
- **جملة switch** – استخدم هذه الجملة عندما تريد تنفيذ نصاً برمجياً من مجموعة من النصوص البرمجية إذا تحقق شرط معين.

جملـة If و If...else

وتستخدم جملة if –كما ذكرنا- لتنفيذ نص برمجي عند تحقق شرط معين؛ إما إذا لم يتحقق الشرط (كانت قيمته false) فإنه سيتم تجاوز النص البر مجي ، ولن يتم تنفيذه.

الصيغة

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

مثـال (**شرح المثال**: إذا كان الوقت في متصفحك أقل من ١٠، ستظهر لك عبارة الترحيب Good Morning)

```
<script type="text/javascript">
//If the time on your browser is less than 10,
//you will get a "Good morning" greeting.
var d=new Date()
var time=d.getHours()

if (time<10)
{
document.write("<b>Good morning</b>")
}
</script>
```

لاحظ أنه لا توجد else في المثال، فقط يتم تنفيذ الأمر (طباعة رسالة الترحيب) إذا تحقق الشرط (الوقت أقل من ١٠).

إما إذا أردت تنفيذ نصاً برمجياً عند تحقق الشرط، ونصاً آخر عند عدم تحققه، فعليك استخدام If...else .

صيغة If...else

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

مثال (**شرح المثال**: إذا كان الوقت في متصفحك أقل من ١٠، ستظهر لك عبارة الترحيب !Good Morning؛ و إلا ستظهر عبارة (Good day!

```
<script type="text/javascript">
//If the time on your browser is less than 10,
//you will get a "Good morning" greeting.
//Otherwise you will get a "Good day" greeting.
var d = new Date()
var time = d.getHours()

if (time < 10)
{
document.write("Good morning!")
}
else
{
document.write("Good day!")
}
</script>
```

جملــة Switch

وتستخدم تنفيذ واحد من مجموعة من النصوص البرمجية عند تحقق شرط معين.

الصيغة

```
switch (expression)
{
case label1:
  code to be executed if expression = label1
  break
case label2:
  code to be executed if expression = label2
  break
default:
  code to be executed
  if expression is different
  from both label1 and label2
}
```

كيف يعمل؟ - أولاً لدينا تعبير expression (في الغالب متغير)، يتم اختباره لمرة واحدة، ثم تقارن قيمة هذا التعبير مع قيم كل حالة case في التركيب البرمجي (أنظر الصيغة أعلاه) ؛ فإذا حصل تطابق مع أي من الحالات، يتم تنفيذ النص البرمجي الموجود معها. وتستخدم الحالة default عندما لا يوجد تطابق مع جميع الحالات، وتستخدم break لمنع التسلسل التنفيذي من الذهاب للحالة التالية عند وجود تطابق.

مثال: (**شرح المثال**: ستتلقى عبارة ترحيب مختلفة بحسب اليوم)

```
<script type="text/javascript">
//You will receive a different greeting based
//on what day it is. Note that Sunday=0,
//Monday=1, Tuesday=2, etc.
var d=new Date()
theDay=d.getDay()
switch (theDay)
{
case 5:
  document.write("Finally Friday")
  break
case 6:
  document.write("Super Saturday")
  break
case 0:
  document.write("Sleepy Sunday")
  break
default:
  document.write("I'm looking forward to this weekend!")
}
</script>
```

المعامـــلات الشرطيـــة

تحتوي جافاسكريبت أيضاً على معاملات شرطية؛ وذلك لتعيين قيمة إلى متغير بناءاً على تحقق شرط معين.

الصيغة

```
variablename=(condition)?value1:value2
```

مثـــل

```
greeting=(visitor=="PRES")?"Dear President ":"Dear "
```

شرح المثال: إذا كان المتغير visitor يساوي PRES ، يتم تخزين النص " Dear President" في المتغير greeting ؛ إما إذا كان المتغير visitor لا يساوي PRES ، فسيتم تخزين النص " Dear" في المتغير greeting.

التـــكـــــرار

تستخدم جمل التكرار في جافاسكريبت لتنفيذ مجموعة جمل برمجية لعدد محدد من المرات.
لدينا ثلاثة أنواع من جمل التكرار في جافاسكريبت:

- **while** –يتم تكرار مجموعة من الجمل البرمجية ما دام الشرط صحيحاً.
- **do…while** –يتم تكرار مجموعة من الجمل البرمجية ثم اختبار الشرط؛ فإذا كان صحيحاً يتم تكرار التنفيذ.
- **for** –تكرار تنفيذ مجموعة جمل برمجية لعدد محدد من المرات.

جملة while

وهي تنفذ مجموعة جمل برمجية ما دام الشرط صحيحاً (يتم اختبار الشرط أولاً)؛ وهذه صيغتها:

```
while (condition)
{
    code to be executed
}
```

جملة do…while

يتم فيها تنفيذ مجموعة جمل برمجية ثم اختبار الشرط؛ فإذا كان صحيحاً يتم تكرار التنفيذ.

```
do
{
    code to be executed
}
while (condition)
```

جملة for

هذه الجملة تنفذ مجموعة جمل برمجية عدداً محدداً من المرات.

```
for (initialization; condition; increment)
{
    code to be executed
}
```

بعــض كائنـــات جافاسكريبت

سنلقي الضوء على ثلاثة من كائنات جافاسكريبت، وهي كائن التاريخ Date Object وكائن الدوال الرياضية Math Object وكائن المصفوفة Array Object.

كائـن التاريخ Date Object

ويستخدم للعمل مع التواريخ والأوقات (الساعات، الدقائق و الثواني).
يمكنك إنشاء نسخة من كائن التاريخ، باستخدام كلمة "new" المحجوزة في جافاسكريبت.
لتخزين التاريخ الحالي في متغير، على سبيل المثال المتغير my_date :

```
var my_date=new Date()
```

بعد إنشاء نسخة من كائن التاريخ، يمكنك الوصول للطرق methods الخاصة بكائن التاريخ من خلال المتغير my_date. على سبيل المثال؛ لو أردت الحصول على التاريخ من كائن التاريخ، قم بكتابة:

```
my_date.getDate()
```

يمكنك أيضاً كتابة التاريخ داخل أقواس، كالتالي:

```
new Date("Month dd, yyyy hh:mm:ss")
new Date("Month dd, yyyy")
new Date(yy,mm,dd,hh,mm,ss)
new Date(yy,mm,dd)
new Date(milliseconds)
```

هنا، كيف تنشئ كائن التاريخ لكل طريقة من الطرق أعلاه:

```
var my_date=new Date("October 12, 1988 13:14:00")
var my_date=new Date("October 12, 1988")
var my_date=new Date(88,09,12,13,14,00)
var my_date=new Date(88,09,12)
var my_date=new Date(500)
```

كائن الدوال الرياضية Math Object

وهو عبارة عن مجموعة من الدوال والثوابت الرياضية المعرفة مسبقاً والجاهزة للاستخدام.
لا تحتاج لتعريف نسخة من كائن الدوال الرياضية قبل استخدامه.
لتخزين رقم عشوائي بين (٠) و (١)، في متغير اسمه –على سبيل المثال- "r_number" نكتب التالي:

```
r_number=Math.random()
```

لتخزين الرقم المقرب بالنسبة للعدد ٨,٦ في متغير اسمه "r_number" :

```
r_number=Math.round(8.6)
```

هناك العديد من الثوابت في هذا الكائن مثل: E لإعادة أساس اللوغاريتم الطبيعي، و وهناك العديد من الدوال الجاهزة مثل abs(x) لإعادة القيمة المطلقة للعدد x.

في نهاية هذا الكتاب، هنا مجموعة من النصائح والتلميحات والتي هي بمثابة خطوط إرشاد للتمكن من جافاسكريبت:

جافاسكريبت حساسة لحالة الأحرف
الدالة المسماة myfunction تختلف عن myFunction ، لذلك انتبه لهذا الأمر عند إنشاء أو استدعاء المتغيرات، الكائنات والدوال.

الرموز (الأقواس)
الرموز والأقواس المفتوحة ({ [" ' يجب أن تغلق بنفس الأقواس والرموز وبنفس الترتيب ' "] }) .

المسافات الفارغة
تتجاهل جافاسكريبت المسافات الفارغة، يمكنك إضافة المزيد من المسافات لجعل النص البرمجي واضحاً ومقروءاً.
السطرين التاليين هما نفس الشيء:

```
name="Hege"
name = "Hege"
```

قطع السطر البرمجي
يمكنك قطع السطر البرمجي داخل السلسلة النصية بشرطة مائلة للخلف \ . هذا المثال سيعرض بشكل صحيح:

```
document.write("Hello \
World!")
```

ملاحظة: لا يمكنك قطع السطر البرمجي بهذه الطريقة:

```
document.write \
("Hello World!")
```

سيسبب المثال أعلاه خطأ –عند تنفيذه-.

إدراج رموز خاصة
يمكنك إدراج رموز خاصة (مثل & ; ' ") باستخدام الشرطة المائلة للخلف \ :

```
document.write ("You \& I sing \"Happy Eid\".")
```

سوف يظهر الأمر أعلاه بعد تنفيذه كالتالي:

```
You & I sing "Happy Eid".
```

التعليقات
يمكنك إضافة التعليقات للنص البرمجي لجافاسكريبت ، وذلك بابتداء التعليق بشرطتين مائلتين // :

```
sum=a + b  //calculating the sum
```

يمكنك أيضاً أن تبتدئ التعليقات بـ " */ "وتنهيها بـ " /* "مثل:

```
sum=a + b  /*calculating the sum*/
```

وتفيد هذه الطريقة في إنشاء أكثر من سطر للتعليقات:

```
/* This is a comment
block. It contains
several lines*/
```

ونختم أخيراً بمجموعة من **مصطلحـــات جافاسكريبت** وشرح موجز لكل منها:

المصطلح الإنجليزي	المصطلح العربي	شرح موجز
Script Block	**وحدة نص برمجي**	كل النص البرمجي والتعليقات المحصورة بين علامتي <script> و </script>
Variable	**متغير**	مكان يستخدم لتخزين الأرقام والنصوص والقيم البوليانية true/false
Initialize	**أولي**	تعيين قيمة أولية للمتغير، سواء عند إنشائه أو فيما بعد.
Call	**استدعاء**	استدعاء دالة بواسطة اسمها.
Function	**دالة**	جزء من النص البرمجي لجافاسكريبت وتعرف لمرة واحدة، ويمكن أن تستخدم أكثر من مرة.
Argument	**معامل**	قيمة يمكن تمريرها للدالة عند الاستدعاء لاستخدامها في التنفيذ.
Expression	**تعبير**	نص برمجي يجمع بين متغيرين أو قيم حرفية مع معامل جافاسكريبت وذلك لإنتاج نتيجة للتعبير.
Statement	**جملة**	تعليمة جافاسكريبت واحدة كاملة تنتهي بفاصلة منقوطة.
Statement Block	**وحدة جمل**	مجموعة جمل جافاسكريبت تنفذ بواسطة دالة عند استدعائها.
For Loop	**التكرار For**	جملة تنفذ مع التكرار جزء من نص جافاسكريبت لعدد معطى من المرات.
While Loop	**التكرار While**	جملة تنفذ مع التكرار جزء من نص جافاسكريبت إذا تحقق شرط معين.
Conditional Branching	**التفرع المشروط**	جملة تنفذ جزء من نص جافاسكريبت إذا تحقق شرط معين؛ وإذا لم يتحقق الشرط، تنفذ جزءاً آخر.
Multi-way Branching	**التفرع المتعدد**	جملة تختبر تعبيراً معطى، وبناءاً على النتيجة تنفذ جزءاً معيناً من النص البرمجي، و إلا فإنها تنفذ النص البرمجي الافتراضي default
Return	**العودة**	النتيجة النهائية لدالة جافاسكريبت تمرر إلى المكان الذي استدعى الدالة.

■ من كتاب جافاسكريبت في خطوات سهلة JavaScript in easy steps للمؤلف مايك ماك جراث.

ملاحـــظات مهمـــة:

- كل عبارة (نص برمجي) أو (جملة برمجية) ورادة في هذا الكتاب تقابل المصطلح الإنجليزي "كود" Code.
- كل عبارة (نص برمجي) أو (جملة برمجية) ورادة في هذا الكتاب قد تجدها في المواقع العربية على الإنترنت بمصطلحات مثل (سكريبت - سكريبتات – أكواد)
- المفاهيم البرمجية الواردة في هذا الكتاب مثل المتغيرات، الدوال، التكرار Loops ... الخ؛ هي مفاهيم متكررة في جميع لغات البرمجة تقريباً.
- عند الرغبة في الحصول على (سكريبتات) أو (أكواد) جافاسكريبت جاهزة، يمكنك البحث عن هذه المواقع في محركات البحث بكتابة "سكربتات جافاسكريبت".
- عند وجود أخطاء إملائية أو برمجية (معلومة خاطئة مثلاً)، الرجاء إرسال رسالة بذلك إلى apc1424@yahoo.com
- التوزيع مجاني لهذا الكتاب الإلكتروني، مع الإشارة –عند النقل أو الاقتباس –إلى W3Schools.

وفي النهاية تقبلوا خالص تحياتي

عبدالله محمد الغامدي

apc1424@yahoo.com