سلسلة الموسوعة العربية للكتب الإلكترونية

http://www.c4arab.com

تعلّم PHP بسهولة

مع العديد من الأكواد و المشاريع الجاهزة !

HTML, CSS, JavaScript, MySQL, XML

تهاني السبيت
مشرفة الموسوعة العربية للكمبيوتر و الإنترنت

الطبعة الأولى
2002

!!نسخة تجريبية غير مكتملة!!

التواصل مع القراء

إلى القارئ العزيز ،،،

حرصت الموسوعة العربية للكمبيوتر و الإنترنت – و من منطلق اهتمامها العام بعلوم الحاسب و التقنية و اهتمامها الخاص بتقديم هذه العلوم باللغة العربية – على تقديم هذه السلسلة من الكتب الإلكترونية التي نتمنى أن تحقق طموحات القارئ العربي الذي اعتاد على قراءة أجود المطبوعات بكافة اللغات العالمية .

إن الموسوعة العربية – من خلال هذه السلسلة – تطمح لتقديم سلسلة من الكتب بمستوى عالٍ من الجودة ، الشيء الذي لن يتحقق بدون ملاحظاتكم و اقتراحاتكم حول السلسلة – طريقة الكتابة ، الأخطاء الإملائية و النحوية ، التنظيم و الترتيب ، طريقة نشر الكتاب و توزيعه ، الإخراج الفني ... الخ

ننتظر سماع أراءكم على البريد الإلكتروني المخصص لذلك :
ebooks@c4arab.com
نرجو ذكر اسم الكتاب و الكاتب و الطبعة مع ملاحظاتكم لنا .

تـــــــهاني الســـــــــبيت
مشرفة الموسوعة العربية للكمبيوتر و الإنترنت

شكر و تقدير

أن تكتب كوداً برمجياً ... أو حتى عشرات الأكواد ، وحدك ، فهذا شيء ممكن . لكن أن تؤلف كتاباً عن لغة برمجة بدون دعم و مساندة حقيقية ممن تثق بهم فهذا شيء غير ممكن .

أتقدم بكلمة شكر بسيطة لكل من ساعد على إخراج هذا الكتاب ... كلمة شكر بسيطة لأعضاء مجلسي إدارة و تطوير الموسوعة العربية للكمبيوتر و الإنترنت ... كلمة شكر بسيطة لأعضاء و زوار الموسوعة بكافة أقسامها .

لولا الله ، ثم لولا اهتمامكم بظهور هذا الكتاب بصورة جيدة ، لما وجدت هذه الصفحات طريقها إلى قارئنا العزيز .

كلمة شكر مقدمة لكل من يفكر في تقديم أي اقتراح أو مساعدة لتطوير هذا الكتاب حتى يكون مرجعاً عربياً شاملاً للغة "بي اتش بي" .

شكر خاص للوالد و الوالدة .. لم يكن بإمكاني التقدم خطوة واحدة بدون الدعم و المساندة الدائمة من قبلكما حفظكما الله من كل سوء .

عن الكاتبة

تهاني السبيت

تاريخ الميلاد : 4- 4 -1401 هـ الموافق 8-2-1981م في الرياض

طالبة في كلية الحاسب و المعلومات – جامعة الملك سعود بالرياض (السنة الأخيرة)

تولت منصب رئاسة مجلس إدارة الموسوعة العربية منذ نشأتها عام 2000 و حتى الآن

تعمل حالياً كمصممة و مطورة مواقع ويب مع شهادة خبرة في نفس المجال كما تعمل في فريق الدعم الفني التابع لأحد شركات الاستضافة العربية .

عدد أسطر بي اتش بي التي كتبتها حتى الآن يزيد عن الـ 20,000 سطر برمجي .

من بين اللغات الأخرى التي تجيدها : باسكال ، سي\سي ++ ، بيسك\ فيجوال بيسك ، أسمبلي .

و في مجال قواعد البيانات : أكسس ، اوراكل ، MySQL .

فهرس المحتويات باختصار

فهرس المحتويات بالتفصيل

مقدمة

بسم الله الرحمن الرحيم

طالما أنك حصلت على هذا الكتاب و طالما أنك تقرأ هذه الكلمات الآن فإنني أفترض أن لديك اهتمام حقيقي لتعلم لغة "بي اتش بي" . إذا كان هذا الافتراض صحيحاً فإننا – القارئ العزيز و كاتبة هذه السطور – ننتمي إلى مجال اهتمام واحد و سنسير في طريق واحد يساعد أحدنا الآخر حتى نصل إلى هدفنا النهائي – إتقان لغة "بي اتش بي" . المساعدة التي أطلبها منك بسيطة ، كل ما أريده أن تؤمن بأن هناك الكثيرين اللذين تمكنوا من إتقان هذه اللغة بل و الإبداع و الابتكار فيها مما يعني أن حلمك بإتقانها ليس بعيد المنال فكن على ثقة بنفسك و بقدراتك و ستصل بإذن الله !
المهم أن تركز على الفصول الأولى و تتقن الأساسيات أولاً و لا تحاول أبداً أن تتخطى هذه الفصول لتعجل بالنتيجة فهذا الطريق السريع الخاطئ هو أكثر الأخطاء شيوعاً عند تعلم هذه اللغة . نريد أن نصل لمستوى عالٍ من الإتقان و لا أعتقد أن القارئ الكريم يختلف معي في أن هذا لا يمكن أن يحدث بدون إتقان الأساس أولاً ثم التقدم إلى الأمام بخطوات واثقة .
إذن .. نحن متفقون على هذا المبدأ ؟!

أما أولئك الأشخاص اللذين يقرؤون هذا الكتاب ليس لتعلم "بي اتش بي" بل لزيادة معلوماتهم حولها فأيضاً أنصح بقراءة الكتاب كاملاً و لو على عجالة فقد تجد معلومة جديدة بين سطر و آخر .

دعونا نستعرض أبواب الكتاب بشكل سريع ..
الباب الأول – مداخل للغات أخرى – تبرز أهمية هذا الباب لدى مطور الويب الذكي الذي يستخدم اللغة المناسبة في الوقت المناسب .
قد توفر لك لغة "بي اتش بي" حلولاً كاملة لمختلف طلباتك – أو طلبات عملاءك و لكن لماذا تستخدم "بي اتش بي" مع وجود بديل أسهل و أفضل ؟
حاول دائماً أن تكون دقيقاً في هذا الجانب و كن ذكياً في اختيار ما يناسبك بالفعل !
الفصل الأول يقدم مدخل سريع للغة "اتش تي ام ال" (HTML) و في الواقع لا يمكنك تخطي هذا الفصل في أي حال من الأحوال – إلا إن كنت متقناً لهذه اللغة فعلاً . لغة "الاتش تي ام ال" هي اللغة التي يفهمها متصفح الإنترنت لديك و يستخدمها لعرض صفحات الويب . إذن هي الخطوة الأولى لكل مصمم و مطور ويب .
لغة "بي اتش بي" من اللغات التي تضع أكوادها بين أكواد لغة "الاتش تي ام ال" لذا لا أتصور أننا نستطيع فهم "بي اتش بي" دون فهم "اتش تي ام ال" .
في الغالب ستتولى "بي اتش بي" الجانب البرمجي و تتولى "اتش تي ام ال" جانب عرض النتيجة البرمجية .

الفصل الثاني يقدم مدخلاً سريعاً للغة "سي اس اس" و تستخدم لإضافة المزيد من المرونة على لغة "اتش تي ام ال" كما سنرى إن شاء الله .

الفصل الثالث يعطيك معلومات سريعة حول لغة الجافا سكريبت التي تعتبر حلاً برمجياً يتوسط الطريق بين (HTML) و (PHP) . السبب في تقديم هذا الفصل هو أن الجافا سكريبت تعتبر خياراً أفضل من بي اتش بي في بعض الجوانب مثل التأكد من مدخلات النماذج . سنتعرف على ذلك بالتفصيل لاحقاً إن شاء الله .

الباب الثاني – مدخل للغة بي اتش بي – يبدأ بنا المشوار لتعلم هذه اللغة مع هذا الباب . لن نبدأ في البرمجة هنا ! لكننا سنتعرف أكثر على هذه اللغة و كيفية تنصيبها على أجهزتنا تحضيراً لاستخدامها في الأبواب التالية .

الباب الثالث – أساسيات بي اتش بي – يتناول هذا الباب أساسيات البرمجة بلغة بي اتش بي . سنتعرف على الهيكل العام لأي برنامج ، على المتغيرات ، الجمل البرمجية المختلفة و غيرها .

الباب الرابع – تعمق في بي اتش بي – سينقلنا لمستوى أعلى في دراستنا لهذه اللغة . سنتمكن من التعامل مع المصفوفات بطرق شيقة و سهلة . سنتعرف بعمق على المتغيرات النصية (String) و بعض الدوال الخاصة بها . و سنتعرف في ختام هذا الباب على موضوع يشكل الاهتمام الأول لكل مهتم بلغة بي اتش بي ، ذلك هو التعرف على كيفية التعامل مع النماذج و مكوناتها المختلفة .

الباب الخامس – الدوال – تقدم لنا لغة بي اتش بي قائمة عملية لدوال يمكن استخدامها لأغراض متنوعة .. كما أنك تستطيع تعريف دوال إضافية خاصة بك . سنتناول موضوع الدوال بالتفصيل في هذا الباب مع أهمية إدراج ملحق خاص بالقائمة الكاملة للدوال الجاهزة في آخر الكتاب .

الباب السادس – التعامل مع الملفات – و كما يشير اسم الباب فإن الموضوع الذي سنلقي الضوء عليه هنا هو كيفية التعامل مع الملفات و الاستفادة الحقيقية من هذه الإمكانيات التي تقدمها لنا بي اتش بي .

الباب السابع – البرمجة الشيئية – أضافت بي اتش بي الكثير إلى نفسها كلغة برمجة معاصرة بسبب دعمها للبرمجة الشيئية مثلها في ذلك مثل أقوى لغات البرمجة الحالية كلغة السي بلس بلس و لغة الجافا . ستفيدك خبرتك في مثل هذه اللغات في تخطي هذا الباب بسهولة أكثر لكن هذه الخبرة غير مطلوبة بشكل إلزامي . تم تخصيص باب كامل للبرمجة الشيئية لتتمكن من الاستفادة الحقيقية منها .

الباب الثامن – قواعد البيانات – أصبح الانتقال من مواقع ويب الثابتة إلى المواقع المعتمدة على قواعد البيانات هاجساً يداعب أغلب مسئولي المواقع العربية منها و الأجنبية . هل حلمت يوماً ببرمجة سكريبت لإدارة موقعك ؟ مجلة عربية ؟ أو حتى منتدى حوار عربي خاص بموقعك ؟ إن كنت جاداً في حلمك فستجد مبتغاك في هذا الباب إن شاء الله . كما أن الباب يقدم مدخلاً للغة اس كيو ال (SQL) أيضاً . لم أقم بإدراج هذا المدخل في الباب الأول لأنك لن تحتاجه إلا مع هذا الباب بعكس مداخل الباب الأول التي احتجناها في أبواب متفرقة .

الباب التاسع – القوالب – إذا وضعنا رحالنا عند الباب التاسع فإننا سنكون حينها قد تمكنا من كتابة مشاريع يزيد عدد أسطرها عن الخمسة آلاف سطر تقريباً . عندها ستفكر جدياً في حل يسهل عليك إجراء تعديلات على طريقة عرض مشروعك و بتعديل أقل عدد ممكن من السطور فقط ! عليك حينها أن تقرأ هذا الباب و تمتع بما تقدمه بي اتش بي لك !

الباب العاشر – دعم الاكس ام ال – يبدأ الباب بمدخل للغة الاكس ام ال ، ثم يقدم تعريفا و شرحاً لكيفية تطوير مشاريع بي اتش بي متقدمة من خلال الاستفادة من هذه اللغة ! و كما ذكرنا بخصوص لغة الاس كيو ال فإنني فضلت عدم إدراج مدخل للغة الاكس ام ال في الباب الأول لأنه لا يهم جميع قراء هذا الكتاب كما أتوقع !

الباب الحادي عشر – ما تبقى عن بي اتش بي – يؤكد هذا الباب مبدأ أؤمن به كثيراً .. لن يأتي اليوم الذي تعتقد بأنك تعرف فيه كل شيء عن بي اتش بي ! دائماً هناك المزيد من التجديدات .. هناك المزيد من الأفكار البرمجية خفيفة الظل التي تعطي لبرامجك المزيد من القوة أو المرونة أو السهولة أو الاحترافية أو هذه الأمور معاً !
سنتعرف بدءاً على كيفية الاستفادة من الكوكيز . ثم سنشرح قائمة بمتغيرات جاهزة يمكن أن تفيدك كثيراً .

البابان الأخيران يقدمان لنا مشروعا بي اتش بي متكاملان ... سنبدأ المشروع من المستوى صفر مع بعضاً البعض .. و سنحتفل سوياً بإنجاز هذين المشروعين مع ختام الكتاب إن شاء الله .

الكتاب ملحق ببعض الملاحق المفيدة في آخره ... استخدمها كمراجع ترجع إليها في وقت الحاجة .
كما أعد بإضافة المزيد من الأبواب في الطبعات القادمة بإذن الله .

كيف تقرأ هذا الكتاب ؟

الكتاب معنون بالاسم "تعلم بي اتش بي بسهولة" ... مما يعني أن الكتاب يركز كثيراً على السهولة في تقديم المعلومات . أعتقد أن استخدام إشارات متفق عليها مسبقاً سيجعل الأمر أكثر سهولة علي و عليك عزيزي القارئ .
إذن فلنتفق على التالي :

كود بي اتش بي

إذا وجدت صندوقاً رمادي اللون كتبت عليه بعض الأكواد باللغة الإنجليزية على أسطر مرقمة فإن هذا يعني أنك تقرأ كود مكتوب بلغة بي اتش بي . و المثال الآتي يوضح ذلك :

php.php

```
1 <?php
2   echo ("PHP, the easy way");
3 ?>
```

كما نشاهد أعلاه ، قمنا بكتابة اسم الملف " php.php " في الجهة اليسرى العليا ثم بدأنا بكتابة كود بي اتش بي في أسطر مرقمة . إن الانتباه لاسم الملف سيساعدك كثيراً عندما نبدأ بكتابة مشاريع من عدة ملفات كما أن عدد السطر سيسهل علينا شرح سطر معين بذاته بالرجوع إلى رقم السطر بدلاً من إعادة كتابة السطر كاملاً .

معلومة إضافية

معلومة إضافية

ستجد بعض المعلومات المفيدة في صناديق شبيهه بهذه الصناديق خلال رحلتك مع هذا الكتاب . قراءة هذه الصناديق غير إلزامية لأنها عادة ستحتوي على بعض المعلومات التاريخية أو الإحصائية التي تهمك فقط إذا كنت مهتماً بهذا النوع من المعلومات .
لا بأس من الاطلاع السريع على المعلومة على الأقل !

تلميحة ذكية

تلميحة ذكية

ستجد الأفكار الذكية في صناديق شبيهه بهذه الصناديق خلال رحلتك مع هذا الكتاب . لا تفوت على نفسك فرصة قراءة هذه الأفكار الذكية لأنها ستساعدنا على الابتكار و التجديد من وقت لآخر .
عندما تفكر في كتابة أكوادك الخاصة ... لا تعطل الجانب الإبداعي و الابتكاري لديك .. هذا الجانب موجود و لكنه بحاجة لتنشيط من خلال إعطاءك بعضاً من هذه الأفكار في صناديقنا هذه .

تحذير !

تحذير !

انتبه جيداً لهذه الصناديق ! إنها تحتوي على تحذيرات هامة تحميك من القيام بشيء قد تندم على فعله إن لم تستمع إلى نصائح الكتاب .

و بعد أن اتفقنا على هذه الإشارات فإنه يكون قد تبقى لنا شيء واحد نذكره بخصوص كيفية قراءة الكتاب ... ألا و هو الأسلوب الذي تستخدمه لتحقيق الاستفادة القصوى من هذا الكتاب . نصيحتي في هذا الجانب هي أن تختار الرفيق المناسب في هذه الرحلة . فإما أن تطلب من شخص متمكن من هذه اللغة أن يشرح لك محتويات الكتاب في دورة مجزاة إلى عدة مستويات – بحسب الأبواب المختلفة في هذا الكتاب – بحيث يكون الكتاب هو المنهج الذي تسيران عليه . أو أن تختار صديقاً بمستوى خبرتك الحالية و تتعاونان لتسهيل الأمر عليكما ، بحيث يتولى أحد الأطراف شرح الباب الأول ثم يتولى الطرف الثاني شرح الباب الثاني و هكذا .

هذا كل شيء ! فلنبدأ باسم الله .

الباب الأول : مداخل للغات أخرى

مقدمة الباب الأول

يقدم لك هذا الباب مداخل للغات أخرى غير لغة بي اتش بي . كلمة مدخل تعني أننا لن نتعمق كثيراً في تفاصيل كل لغة ، بل سنأخذ من كل لغة ما يلزمنا لكتابة أكواد بي اتش بي واضحة و صحيحة .
سنتعرف في الفصل الأول على لغة الاتش تي ام ال .. اللغة الأولى على شبكة الويب . و في الفصل الثاني سنتعرف على لغة تضيف مرونة هائلة للغة الاتش تي ام ال(HTML) و هي لغة السي اس اس (CSS) . في الفصل الثالث سنتعرف على لغة الجافا سكريبت التي تعد خياراً ذكياً يضيف القوة و الجودة على مشاريع بي اتش بي .

الفصل الأول : مدخل للغة HTML

بالتأكيد سمعت عن لغة الاتش تي ام ال قبل ذلك ؟ أو كما يحب أن يسميها البعض لغة "هتمل" .
في الواقع فإن هذه الشهرة لهذه اللغة – التي لا يحب البعض أن يعتبروها لغة حقيقية – ليس بمستغرب، ذلك أن هذه اللغة شهدت بدايات ظهور شبكة الإنترنت حيث كانت و مازالت تستخدم لكتابة الصفحات التي يمكن استعراضها عن طريق متصفح مواقع إنترنت (مثل الإنترنت اكسبلورر ، النت سكيب ، الأوبرا أو غيرها) .
كانت البداية بسيطة جداً ... تمكنت لغة الهتمل في البداية من عرض النصوص الجامدة التي تتناسب مع طلبات مستخدمي الشبكة في ذلك الوقت و غالبيتهم من العسكريين أو الفيزيائيين .
مع انتشار شبكة الإنترنت الواسع في التسعينات من القرن الماضي كان لابد من أن تتطور هذه اللغة و تتطور معها مواقع شبكة الإنترنت . و كما تلاحظ بنفسك ، تطورت الطريقة التي تعرض بها المعلومات من الطريقة النصية الجامدة إلى الطريقة المرئية المتحركة .

تشير الأحرف في اسم اللغة إلى العبارة الإنجليزية التالية :
Hyper **T**ext **M**arkup **L**anguage
التي تشير إلى (لغة ترميز النص التشعبي)

تتكون لغة هتمل في مجملها من عدة وسوم (Tags) يحمل كل وسم من هذه الوسوم معنى خاص لدى المتصفح .
لنتخيل الأمر كالتالي : بما أنك مصمم مواقع ويب فإنك قد عقدت اتفاقاً مع المتصفح و أنشأت لغة خاصة للتفاهم بينك و بينه .
أول اتفاق تم عقده بينكما هو أن الأوامر التي سترسلها له ستكون دائماً بين قوسين من هذا الشكل :

<الأمر هنا>

إذا كان الأمر على هذا الشكل فإننا نسميه وسما .
هذا الأمر يعني أنك تطلب عرض صفحتك بتنسيق معين (لون أحمر ، خط عريض ، اتجاه من اليمين إلى اليسار أو العكس ... الخ) .
المتصفح ملتزم معك باستخدام هذا التنسيق – الأمر – حتى تخبره بالتوقف عن ذلك من خلال إرسال الأمر نفسه مع إضافة الشرطة المائلة (/) بعد القوس الأول مباشرة .
هكذا :

</الأمر هنا>
بطبيعة الحال أنت تأمر المتصفح باستخدام هذا التنسيق على جملة نصية معينة .
فأين يمكن أن نضع الجملة النصية التي نأمر المتصفح باستخدام هذا التنسيق معها ؟
توقعك صحيح !
نعم بالطبع ... يجب أن نضع الجملة المطلوبة بين وسم بداية الأمر ووسم نهاية الأمر .
هكذا :

<الأمر هنا>
الجملة النصية المراد تطبيق التنسيق عليها هنا
</الأمر هنا>

بالإضافة إلى وسوم تنسيق النص فإن لغة الهتمل تضم وسوماً لإدراج صورة أو خط أفقي أو جدول أو ملف فلاشي و كذلك وسوماً خاصة بإدراج مكونات النماذج (Form Elements) . كما أنها تضم وسوماً خاصة ببداية كل صفحة و نهايتها .
كما أن بعض الوسوم تقبل متغيرات إضافية ترسل مع الوسم و تعني معلومة إضافية للمتصفح .
أنت تقوم بكتابة هذه المتغيرات الإضافية داخل قوسي وسم البداية هكذا :
<الأمر هنا متغير="قيمة">
هذه المتغيرات ضرورية مع بعض الأوسمة – لن يفهمك المتصفح بكلمة واحدة ، يجب أن تعطيه المزيد من المعلومات ليفهمك أحياناً . و في أحيانٍ أخرى فإنها تكون اختيارية .
هذه أمثلة سريعة لبعض الوسوم حتى تعتاد على أشكالها :

```
<hr>
<hr width="50%">
<Font color="red">
<img src="images/php.gif">
```

إذن ، ما رأيك أن نبدأ بكتابة أول صفحة هتمل ؟
بداية نريد أن نخبر نظام التشغيل بأن الصفحة التي ننشئها حالياً هي صفحة يمكن عرضها من خلال المتصفح . نقوم بذلك من خلال تسمية الملف بالامتداد (htm. أو html.) .
إذن لنفتح معاً برنامج المفكرة (Notepad) .
ثم نقوم بتسمية الملف بهذا الاسم : 1.htm (يفضل أن تقوم بإنشاء مجلد خاص بأمثلة هذا الكتاب حتى لا تختلط عليك الملفات بكثرة التطبيقات)

تحذير !

عند اختيارك لأسماء ملفاتك فإن حالة الأحرف (كبيرة ـ صغيرة) غير هامة إذا كنت ستقوم بعرض صفحاتك على جهازك و بنظام التشغيل ويندوز و لكنها هامة جداً إذا أردت تحميل ملفاتك على مضيفات حقيقية على شبكة الإنترنت .
حاول أن تختار لنفسك منهجاً في كل شيء ، حتى في حالة الأحرف !
إذا كنت تفضل استخدام الأحرف الصغيرة فاستخدمها بشكل دائم ، و العكس صحيح بالطبع .
كذلك لا أنصح أبداً بتسمية الملفات بأسماء عربية حيث إن أغلبية المضيفات لا تتعرف عليها و بالتالي ستعتبر أن ملفك غير موجود !

ثم لنكتب هذه السطور في ملفنا :

1.htm

```
1 <html>
2 <head>
3         <title>عنوان الصفحة</title>
4 </head>
5 <body>
6
7 </body>
8 </html>
```

شكل 1

ماذا تعني هذه الوسوم ؟

لنتناولها واحداً تلو الآخر ...

<html>

بكتابتك لهذا الأمر فإنك تخبر المتصفح ببداية صفحتك . لاحظ أن نفس الأمر يستخدم في نهاية الصفحة أيضاً – مع الشرطة المائلة – لاختتام صفحتك .

هذا الأمر – بهذا الشكل – لا يقوم بأي تنسيق ... لكن يمكن أن نستفيد منه لتنسيق الصفحات العربية بحيث يكون الاتجاه الافتراضي لها من اليمين إلى اليسار (اتجاه الكتابة ، اتجاه شريط التمرير) .

كل ما عليك هو أن تمرر المتغير الإضافي التالي :

(Dir=”rtl”)

فيصبح الأمر هكذا :

```
<HTML dir=”rtl”>
```

<head>

تتكون كل صفحة هتمل من رأس و جسم ... الجسم يحتوي على المعلومات التي ستعرض لمتصفحي موقعك أم الرأس فإن الغرض منه هو تمرير معلومات أخرى للمتصفح .

الوسم الحالي (<head>) يعني بداية رأس الصفحة .

ستتعلم إضافة أشياء كثيرة داخل هذا الرأس منها وسم إثبات حقوقك في كتابة كود الصفحة و الذي يعتبر أحد الوسوم الوصفية (Meta tags). كذلك يمكنك أن تضيف بعض أكواد الجافا سكريبت أو الفيجوال بيسك سكريبت داخل الرأس .

و بالطبع فإن الوسم بحاجة لوسم إغلاق هو الوسم :

```
</head>
```

<title>

الق نظرة سريعة إلى أعلى نافذة المتصفح – بعد استعراض الصفحة التي أنشأناها قبل قليل ، ستلاحظ وجود العبارة "عنوان الصفحة" . إن هذه التجربة السريعة توضح لك الغرض من استخدام وسم <title> يقوم هذا الوسم بإرسال الجملة التي تكتبها

داخله إلى المتصفح ليقوم بدورة بعرضها على زائر الصفحة كعنوان للصفحة .
و بالطبع فإن الوسم بحاجة لوسم إغلاق هو الوسم :

```
</title>
```

<body>

أنت تعرف الآن أن كل صفحة هتمل تتكون من رأس و جسم .. فإذا كان الرأس يبدأ بالوسم <head> فإن الجسم يبدأ بالوسم <body>.
سنضع ما نريد عرضه داخلي وسمي الجسم .
هناك الكثير و الكثير من الأوسمة التي يمكن أن نضعها داخل جسم الصفحة .. سنتعرف على كل وسم بالتفصيل بعد قليل .
لا أنسى أن أذكرك بأن الوسم يحتاج لوسم ختامي هو :

```
</body>
```

إذن ! اكتب هذه السطور دائماً عند بداية أي ملف هتمل تنشئه .
فهو الشكل العام لكل صفحات هتمل ... و هاهو مرة أخرى للتذكير :

1.htm

```
1 <html>
2 <head>
3         <title>عنوان الصفحة</title>
4 </head>
5 <body>
6
7 </body>
8 </html>
```

شكل 2

تستطيع تذكر هذا الهيكل العام على أساس أنه يتكون من ثلاث مستويات رئيسية :
المستوى الأول هو مستوى وسم <html> .
هذا المستوى يضم المستوى الثاني المكون من جزأين و هما :
المستوى الثاني أ : مستوى الوسم <head>
المستوى الثاني ب : مستوى الوسم <body>
و كل مستوى من هذين المستوين يضم مستوى فرعي هو مستوى الأوسمة التي سنتعرف عليها الآن ...

وسوم رأس الصفحة :

سنبدأ بالتعرف على الأوسمة التي قد تصادفك في رأس الصفحة و هي :

<meta>

هذه هي الأوسمة الوصفية التي تحدثنا عنها قبل قليل . الأوسمة الوصفية ليست وسماً واحداً بل إنها مجموعة كبيرة من الأوسمة يمكنك أن تختار ما تشاء منها أو تتركها جميعاً إن أردت !

هذه الأوسمة لها صيغة عامة .. و تكتب بين وسمي <head> .
الصيغة العامة لها كالتالي :

```
<meta name="" content="">
```

الأمثلة عديدة و لا يمكن أن نسردها جميعاً هنا لذا سنكتفي بذكر أهم الأمثلة فقط :

```
<meta name="author" content="shahrazad@c4arab.com">
```

هذا المثال يوضح لك كيف يمكن أن تستخدم الوسوم الوصفية لإثبات حقوقك ككاتب لهذه الصفحة . يمكن أن يتعرف الزوار على كاتب الصفحة من خلال الاطلاع على الشفرة المصدرية للصفحة و ذلك من خلال الأمر (عرض <== المصدر) في نافذة متصفح إنترنت اكسبلورر .

```
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1256">
<meta http-equiv="Content-Language" content="ar-sa">
```

هذان السطران يطلبان من المتصفح أن يقوم بعرض الصفحة باستخدام التشفير العربي للأحرف . يستحسن أن تضيف هذان السطران في رأس كل صفحة من صفحاتك .
يوجد بدائل للسطر الثاني ، فكما نعلم فإن اللغة العربية هي اللغة الرسمية لـ 22 بلد عربي . مما يعني وجود حوالي 22 بديل لهذا السطر .
مثال :

```
<meta http-equiv="Content-Language" content="ar-ae">
```

كما نعرف .. و كما يثبت لنا استخدام أحد البدائل أعلاه ، فإن اللغة العربية واحدة في أي بلد عربي ! و لا يختلف استخدام أحد هذه البدائل عن غيرها . الخيار هنا لا يتعدى كونه تفضيل شخصي .

مثال آخر :

```
<meta http-equiv="Refresh"
content=0;URL="http://www.c4arab.com">
```

هذا المثال يقدم أحد الحلول التي ستحتاج لها يوماً ما . استخدامك لهذا السطر في رأس الصفحة يعني أنه سيتم تحويل الزائر مباشرة إلى صفحة أخرى هي الصفحة **http://www.c4arab.com** أو كما تحدد بنفسك في الأمر .
لا تتردد بتجربة هذا السطر بنفسك إن لم تكن الفكرة قد اتضحت لك بعد .

وسوم جسم الصفحة :

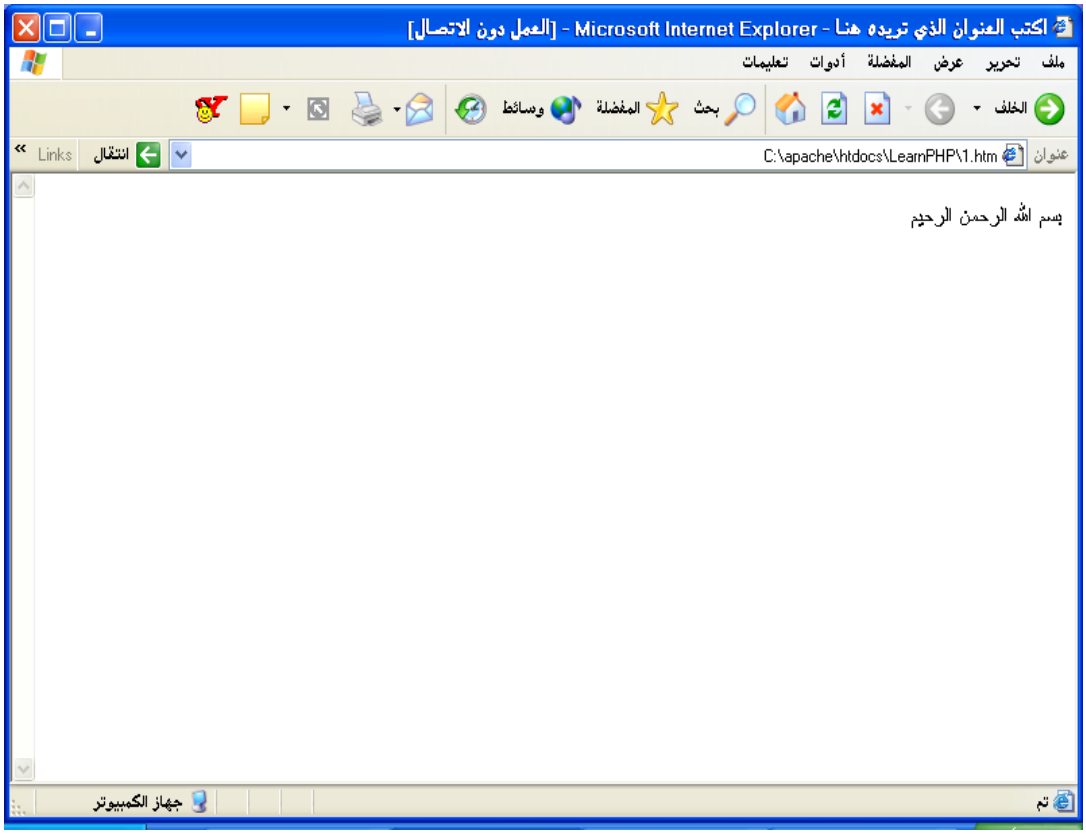
و الآن سننتقل إلى الوسوم التي تهمنا حقاً .. الوسوم التي ستظهر نتائجها لدى زوار صفحتك .
لنأخذ أولاً المثال التالي :

1.htm

```
1 <html dir="rtl">
2 <head>
3       <title>اكتب العنوان الذي تريده هنا</title>
4 </head>
5 <body>
6    بسم الله الرحمن الرحيم
7 </body>
8 </html>
```

شكل 3

كما لاحظت ، لقد قمنا بكتابة "بسم الله الرحمن الرحيم" في جسم الصفحة . عندما تحفظ الملف بالتغييرات الجديدة فإنك ستحصل على صفحة شبيهه للصفحة التالية :



شكل 4

نريد أن نضيف بعض التنسيقات على هذه العبارة .. تابع معي الوسوم التالية :

<font>

هذا الوسم هو وسم خاص بتنسيق الخط المستخدم للكتابة . يمكنك بواسطة هذا الوسم أن تغير الخط المستخدم ، اللون ، الحجم .
كما يوضح الجدول التالي :

المتغير	الوصف
Color	هذا المتغير يحتوي على المعلومات الخاصة باللون المستخدم في الكتابة .
Face	هذا المتغير يحتوي على المعلومات الخاصة باسم الخط المستخدم في الكتابة .
Size	هذا المتغير يحتوي على المعلومات الخاصة بحجم الخط المستخدم في الكتابة .

جدول 1

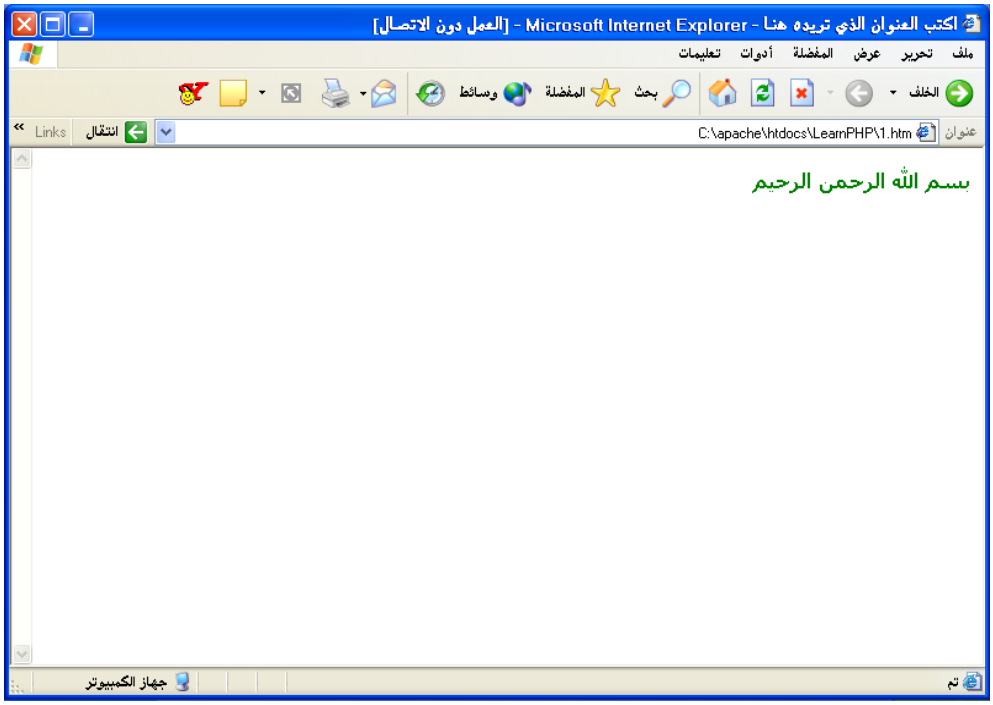
لنضيف ما تعلمناه على ملفنا السابق الآن ..

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6    <font face="Tahoma" color="green" size="4">
7      بسم الله الرحمن الرحيم
8    </font>
9  </body>
10 </html>
```

شكل 5

احفظ الملف الآن .. قم بفتح الصفحة في متصفحك أو قم بتحديث الصفحة إن كانت ما زالت مفتوحة . ستحصل على صفحة شبيهه بالصفحة التالية :



شكل 6

تلميحة ذكية

يجب أن يكون نوع الخط الذي حددته للمتصفح باستخدام الوسم السابق موجوداً على جهازك و أجهزة متصفحي موقعك . ماذا لو لم يمكن الخط موجوداً على أجهزة متصفحي موقعك ؟! يمكنك أن تحدد للمتصفح عدة خيارات يستخدمها لاسم الخط ، حتى تتجنب هذه المشكلة .

يمكنك القيام بذلك عن طريق كتابة أكثر من خط تفصلهم بفاصلة مثل :

```
<Font face="Tahoma, Verdana, Arial">
```

بعد أن تعلمنا كيفية تنسيق الجمل فإننا سنتعلم كيفية تنسيق الفقرات ..

<p>

يقوم هذا الوسم بمهمة تنسيق الفقرات على حسب المتغيرات التي ترسلها معه .
لنأخذ مثالاً على ذلك للتوضيح :

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8      بسم الله الرحمن الرحيم
9    </font>
10  </p>
11 </body>
12 </html>
```

شكل 7

سيكون الناتج كما يلي :

شكل 8

و تجد شرح هذه المتغيرات في الجدول التالي :

المتغير	الوصف
align	هذا المتغير يحتوي على المعلومات الخاصة بمحاذاة النص (يمين،وسط،يسار).

dir	هذا المتغير يحتوي على المعلومات الخاصة باتجاه الكتابة (من اليمين إلى اليسار أو العكس).

جدول 2

<H1>

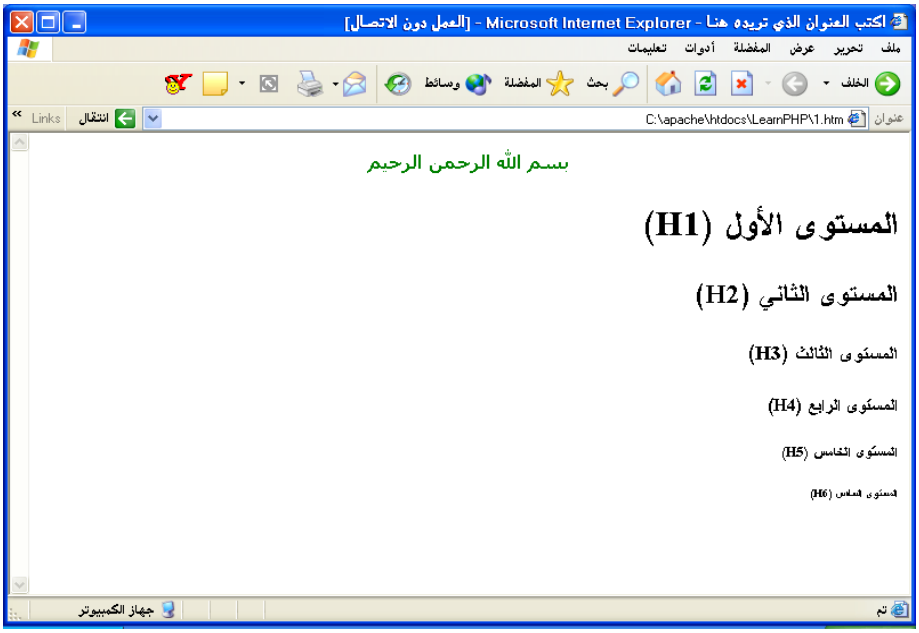
هذا الوسم يقوم بتنسيق الجملة على شكل رأس فقرة (Heading) .
في الواقع يوجد ستة مستويات لرؤوس الفقرة ..
<H1>
<H2>
<H3>
<H4>
<H5>
<H6>
الوسم الأول يشير إلى أعلى مستوى من الرؤوس ... كما يشير الوسم الأخير إلى أقل مستوى من الرؤوس .
لنأخذ مثالاً على ذلك :

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8      بسم الله الرحمن الرحيم
9    </font>
10  </p>
11  <H1>المستوى الأول (H1) </H1>
12  <H2>المستوى الثاني (H2) </H2>
13  <H3>المستوى الثالث (H3) </H3>
14  <H4>المستوى الرابع (H4) </H4>
15  <H5>المستوى الخامس (H5) </H5>
16  <H6>المستوى السادس (H6) </H6>
17 </body>
18 </html>
```

شكل 9

و هذا هو الناتج الذي ستحصل عليه – جرب بنفسك من فضلك :



شكل 10

تحذير !

يقوم المتصفح باستخدام نوع و حجم الخط الافتراضي لديه عند استخدامك لوسم رأس الفقرة .
يمكنك أن تجبر المتصفح على استخدام تنسيق معين مع هذا الوسم من خلال لغة أو تقنية السي اس اس (CSS) التي سنتعرف عليها في الفصل التالي .

لا يوجد الكثير لنقوله هنا ! يقوم هذا الوسم بطباعة سطر فارغ على الشاشة . جرب استخدامه بنفسك و راقب النتيجة . لاحظ أنه يمكنك استخدام هذا الوسم أكثر من مرة بشكل متتالي لطباعة أكثر من سطر فارغ . و لاحظ أيضاً أن استخدام هذا الوسم ضروري في أحوال كثيرة لأن المتصفح سيعرض الأسطر المختلفة لمستندك في سطر واحد ما لم تفصل بينهما باستخدام هذا الوسم أو وسم آخر يقوم بمهمة شبيهه (وسم الفقرات على سبيل المثال : <p>) . هذا يعني أن المتصفح يتجاهل الأسطر الفارغة و المسافات الفارغة تماماً .
هذا الوسم هو أحد الوسوم التي لا تحتاج لوسم إغلاق . استخدمه كما هو و انتهى !

<HR>

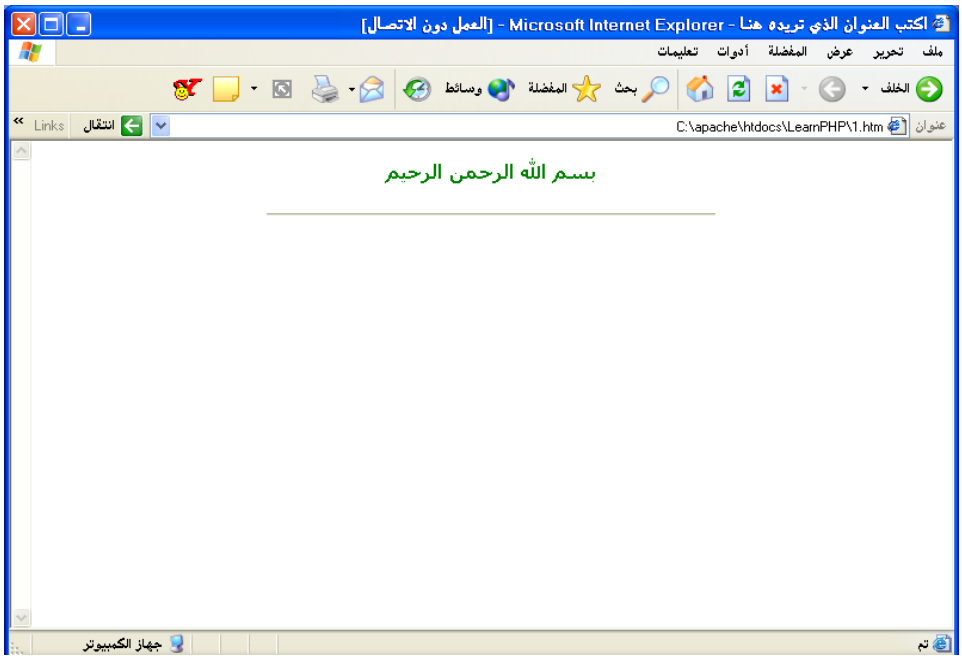
هل لاحظت وجود خط فاصل بين بعض الفقرات على الصفحات التي سبق لك زيارتها ؟ هذا الخط الفاصل نسميه في عالم الوسوم بالمسطرة الأفقية (Horizontal Line) .
أيضاً هذا الوسم لا يتطلب وسماً لإغلاقه .
مثال :

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8      بسم الله الرحمن الرحيم
9    </font>
10  </p>
11  <hr width="50%">
12 </body>
13 </html>
```

شكل 11

و هذه هي النتيجة المتوقعة :



شكل 12

<B>

يقوم هذا الوسم بتنسيق النص على هيئة نص عريض (Bold) .

<I>

يقوم هذا الوسم بتنسيق النص على هيئة نص مائل (Italic) .

<S>

يقوم هذا الوسم بتنسيق النص على هيئة نص مشطوب (Strike through) .

<U>

يقوم هذا الوسم بتنسيق النص على هيئة نص مسطر (Underline) .

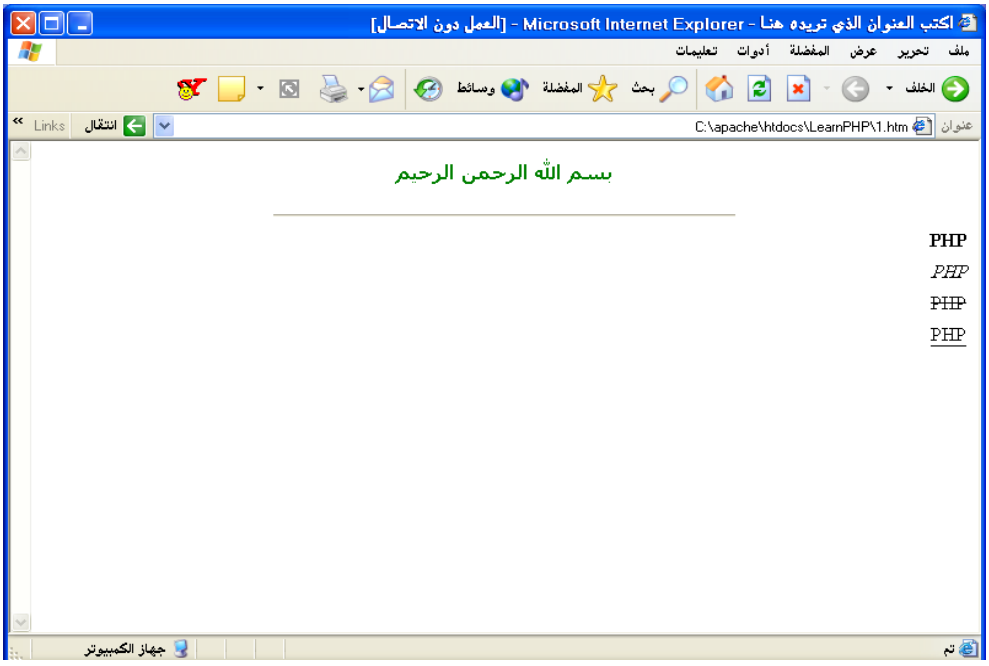
و هذا مثال توضيحي :

1.htm

```
1  <html dir="rtl">
2  <head>
3          <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8       بسم الله الرحمن الرحيم
9    </font>
10  </p>
11  <hr width="50%">
12  <B>PHP</B>
13  <br>
14  <I>PHP</I>
15  <br>
16  <S>PHP</S>
17  <br>
18  <U>PHP</U>
19 </body>
20 </html>
```

شكل 13

و هذه هي النتيجة :



شكل 14

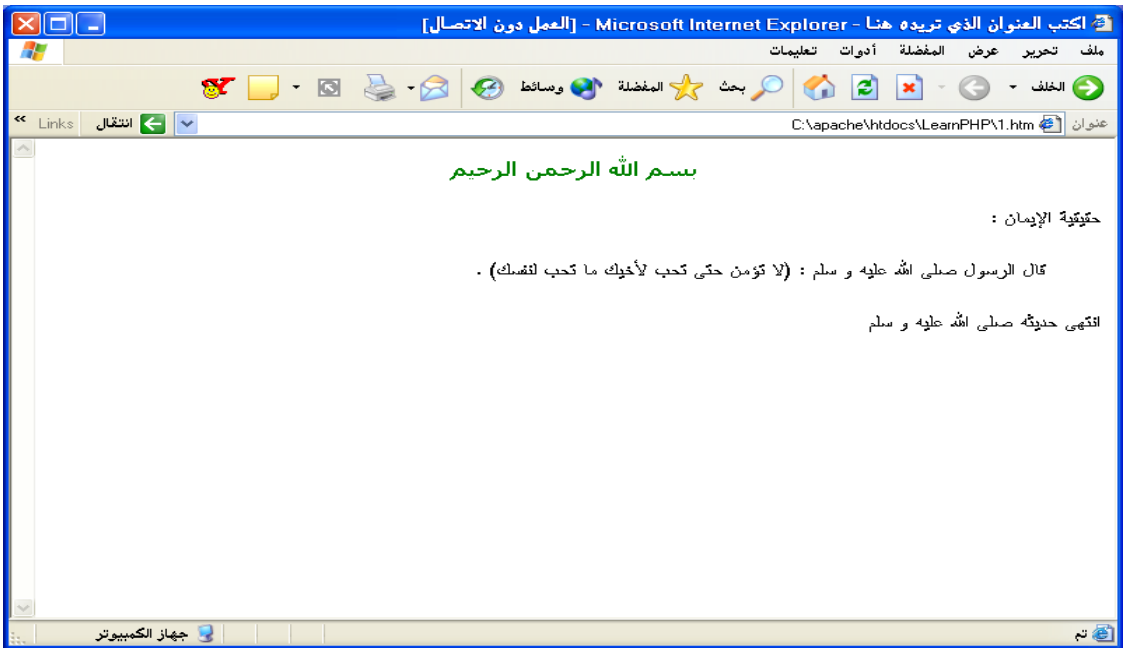
<BlockQuote>

يقدم هذا الوسم تنسيقاً خاصاً بالعبارات المقتبسة من نصوص أخرى (Quote) . فيقوم بتوسيط النص و زيادة الهوامش الجانبية له .
هذا مثال جاهز بعد استخدامه :

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8      بسم الله الرحمن الرحيم
9    </font>
10  </p>
11   حقيقية الإيمان :
12  <BlockQuote>
13   (قال الرسول صلى الله عليه و سلم : (لا تؤمن حتى تحب لأخيك ما تحب لنفسك .
14  </BlockQuote>
15  انتهى حديثه صلى الله عليه و سلم
16 </body>
17 </html>
```

شكل 15



شكل 16

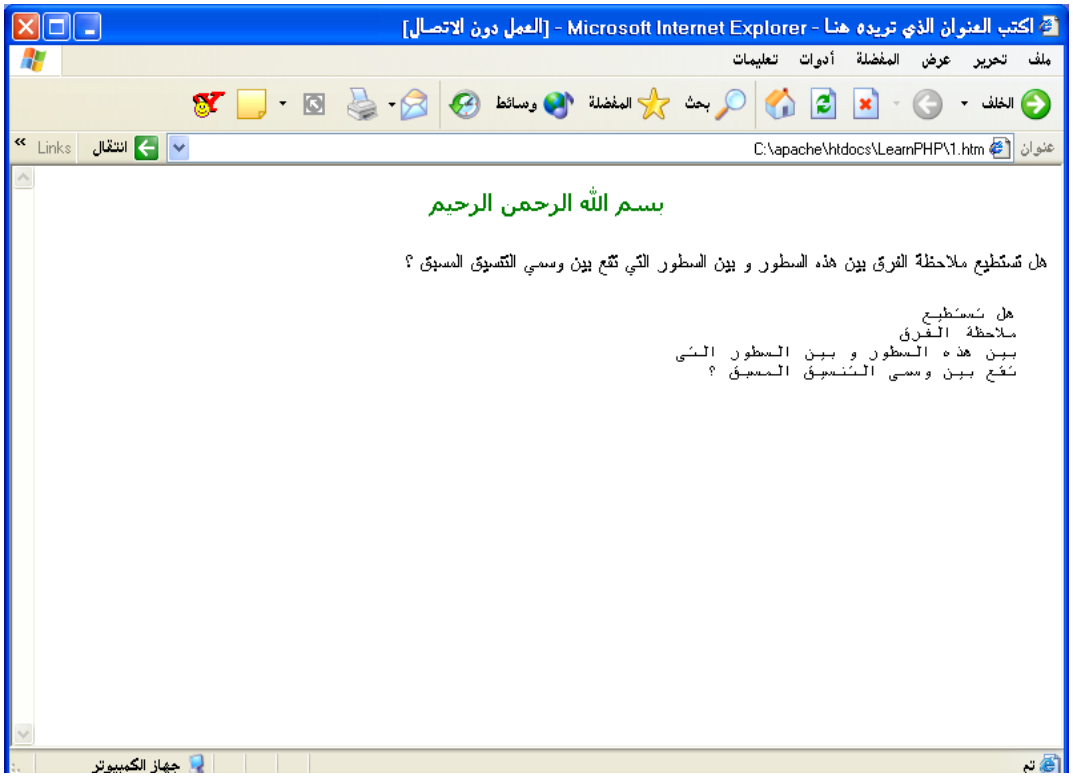
<pre>

الاختصار pre يشير إلى كلمة preformatted أي النص المنسق مسبقاً . يقوم هذا الوسم بتنسيق النص بحسب الطريقة التي كتبت بها نصك تماماً و باستخدام خط " monospace" . إذا كنت قد تركت بعض الأسطر الفارغة بين أسطر نصك فإنها ستظهر هذه المرة كما كتبتها و لن يتجاهلها المتصفح .
راقب المثال التالي بتمعن :

1.htm

```
1  <html dir="rtl">
2  <head>
3          <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8      بسم الله الرحمن الرحيم
9    </font>
10  </p>
11   هل تستطيع
12    ملاحظة الفرق
13    بين هذه السطور و بين السطور التي
14    تقع بين وسمي التنسيق المسبق ؟
15
16    <pre>
17    هل تستطيع
18    ملاحظة الفرق
19    بين هذه السطور و بين السطور التي
20    تقع بين وسمي التنسيق المسبق ؟
21    </pre>
22
23 </body>
24 </html>
```

شكل 17



شكل 18

<code>

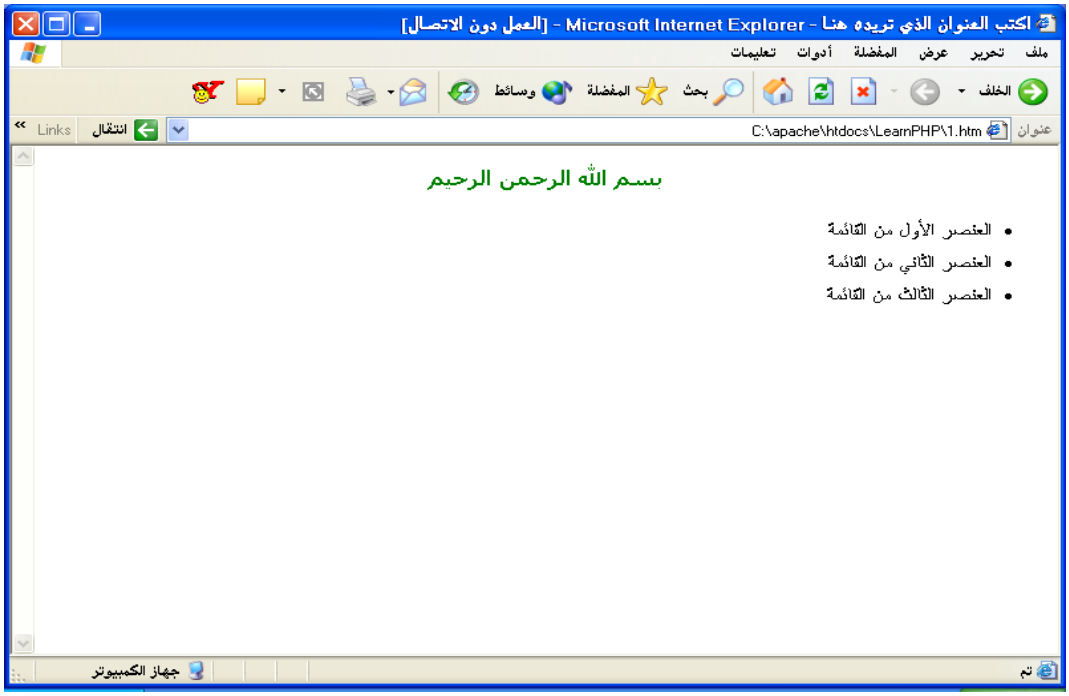
هذا الوسم شبيه جداً بالوسم السابق ،من ناحية انه ينسق النص باستخدام الخط monospace ، لكنه لا يضيف أسطر فارغة في النص المنسق . جربه بنفسك .

يستخدم هذا الوسم لكتابة قائمة منسقة . المثال التالي يوضح ذلك :

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8      بسم الله الرحمن الرحيم
9    </font>
10  </p>
11  <ul>
12  <li>العنصر الأول من القائمة
13  <li>العنصر الثاني من القائمة
14  <li>العنصر الثالث من القائمة
15  <ul>
16 </body>
17 </html>
```

شكل 19



شكل 20

يمكنك إضافة مستوى فرعي للقائمة بطريقة سهلة كما يوضح الكود التالي :

1.htm

```
1  <html dir="rtl">
2  <head>
3          <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8       بسم الله الرحمن الرحيم
9    </font>
10  </p>
11  <ul>
12  <li>العنصر الأول من القائمة
13  <li>العنصر الثاني من القائمة
14      <ul>
15      <li>العنصر الفرعي الأول
16      <li>العنصر الفرعي الثاني
17      </ul>
18  <li>العنصر الثالث من القائمة
19  <ul>
20  </body>
21  </html>
```

شكل 21

عليك أن تجرب النتيجة بنفسك ..

هذا الوسم شبيه بالوسم السابق .. الفارق الوحيد هو أن هذه القائمة هي قائمة مرتبة رقمياً في مقابل القائمة غير المرتبة رقمياً السابقة .

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8       بسم الله الرحمن الرحيم
9    </font>
10  </p>
11  <ol>
12  <li>العنصر الأول من القائمة
13  <li>العنصر الثاني من القائمة
14     <ol>
15     <li>العنصر الفرعي الأول
16     <li>العنصر الفرعي الثاني
17     </ol>
18  <li>العنصر الثالث من القائمة
19  <ol>
20  </body>
21  </html>
```

شكل 22

<a>

و ماذا نفعل إن أردنا إنشاء وصلة تشعبية تنقلنا لصفحة أخرى عند الضغط عليها ؟ علينا أن نستخدم الوسم <a> للقيام بذلك . هذا الوسم له وسم بداية ووسم نهاية و يحتاج لمتغيرين يمرران من خلاله في الغالب . لنطلع على هذا الجدول لمزيد من التوضيح :

المتغير	الوصف
href	هذا المتغير يحتوي على عنوان الصفحة المراد الانتقال إليها .
target	هذا المتغير يسمح لنا بفتح الصفحة الجديدة في نفس الصفحة أو في إطار جديد بحسب القيمة المعطاة له .

جدول 3

لنأخذ مثالاً بسيطاً يوضح ذلك :

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8      بسم الله الرحمن الرحيم
9    </font>
10  </p>
11  <a href="http://www.c4arab.com">اضغط هنا لزيارة الموسوعة في نفس الاطار</a>
12  <br>
13  <a href="http://www.c4arab.com" target="_blank">اضغط هنا لزيارة الموسوعة في إطار جديد</a>
14 </body>
15 </html>
```

شكل 23

مهما كانت صفحتك بسيطة ، فإنك تحتاج عادة لإضافة بعض الحيوية لها من خلال إدراج بعض الصور . يستخدم الوسم <img> لهذا الغرض . يحتاج الوسم إلى وسم البداية فقط و يأخذ بعض المتغيرات التي يوضحها الجدول الآتي :

المتغير	الوصف
Src	هذا المتغير يحتوي على عنوان الصورة

	المراد إدراجها .
Width	هذا المتغير يسمح لنا بتحديد حجم الصورة من حيث العرض .
Height	هذا المتغير يسمح لنا بتحديد حجم الصورة من حيث الطول .
Border	يمكنك أن تضيف إطار خارجي للصورة من خلال هذا المتغير .

جدول 4

المثال التالي يوضح أكثر من حالة :

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">
8      بسم الله الرحمن الرحيم
9    </font>
10  </p>
11  <img src="images/logo.gif" width="10" height="10" border="0">
12  <br>
13  <img src="images/logo.gif" width="10%" height="10%" border="1">
14  <br>
15  <img src="images/logo.gif">
16 </body>
17 </html>
```

شكل 24

الجداول :

دائماً ما أردد و يردد غيري ، استخدام الجداول في تصميم الصفحات هو سر جمالها .
فلنتعرف على كيفية إدراج جدول ..
سنحتاج للتعرف على أكثر من وسم يستخدم لهذا الغرض ، و حتى نفهم هذه الوسوم يجب أن نعرف الهيكل العام لأي جدول :

<وسم بداية الجدول>
<وسم بداية صف>
<وسم بداية خلية>
<وسم نهاية خلية/>
<وسم بداية خلية>
<وسم نهاية خلية/>
<وسم نهاية صف/>
<وسم نهاية الجدول/>

هذا الجدول يتكون من صف واحد يحتوي بدوره على خليتين . أو بمعنى آخر هو جدول مكون من صف واحد و عمودين .

<table>

هذا الوسم هو وسم بداية الجدول ... يحتاج لوسم نهاية أيضاً </table> . يأخذ المتغيرات الموضحة في الجدول التالي :

المتغير	الوصف
bgcolor	يحدد هذا المتغير لون خلفية الجدول .
Width	هذا المتغير يسمح لنا بتحديد حجم الجدول من حيث العرض .
Height	هذا المتغير يسمح لنا بتحديد حجم الجدول من حيث الطول .
Border	يمكنك أن تضيف إطار خارجي للجدول من خلال هذا المتغير .

جدول 5

<tr>

هذا الوسم يشير إلى بداية صف في الجدول . و يحتاج لوسم نهاية أيضاً </tr> .

<td>

هذا الوسم يشير إلى بداية خلية في الجدول . و يحتاج لوسم نهاية أيضاً </td> .

المثال التالي سيوضح هذه الوسوم إن شاء الله :

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">بسم الله الرحمن الرحيم</font>
8   </p>
9   <font face="Tahoma">
10   <table border="0" width="100%" bordercolor="#FFFF00">
11     <tr>
12       <td width="50%" bgcolor="#009900">الخليـــة الأولى</td>
13       <td width="50%" bgcolor="#33CC33">الخلية الثانيـة</td>
14     </tr>
15     <tr>
16       <td width="50%" bgcolor="#00CC99">الخلية الثالثـة</td>
17       <td width="50%" bgcolor="#339933">الخلية الرابعـة</td>
18     </tr>
19   </table>
20 </font>
21
22 </body>
23 </html>
```

شكل 25

النماذج :

ها قد وصلنا لآخر ما سنناقشه في لغة الهتمل . الآن ، أطلب منك أن تركز جيداً لأن استخدام النماذج سيستمر معنا إلى نهاية الكتاب . تأكد من أنك تفهم ما يذكر هنا جيداً ، حاول القيام بعدة تدريبات عملية ، و لا تتردد بطلب المساندة من شخص تثق به في هذا المجال . سنتعلم كيف ندرج عناصر النماذج المختلفة .. بالطبع ستكون هذه النماذج مجرد نماذج شكلية لا تفعل الكثير ؛ سنضيف لها المزيد من القوة في الباب الرابع من هذا الكتاب إن شاء الله .

<form>

يبدأ كل نموذج بهذا الوسم . و ينتهي كذلك بوسم النهاية </form> . هذا الوسم يأخذ بعض المتغيرات الهامة التي يمكن توضيحها من خلال الجدول الآتي :

المتغير	الوصف
method	يحدد هذا المتغير نوعية الطلب المرسل إلى المضيف (post , get , head) .
action	يحدد هذا المتغير اسم الملف الذي سيعالج محتويات النموذج بعد النقر على زر الإرسال .

جدول 6

<input>

الغرض الأساسي من النماذج هو التفاعل مع المستخدم عن طريق قبول بعض المدخلات منه . هذه المدخلات – كما تعرف من خلال خبرتك مع النماذج – لها أشكال متعددة ، فقد تكون عبارة عن مربع نص أو زر خيار أو مربع العلامة أو زر أوامر ... الخ .

و بطبيعة الحال ، تحتاج هذه الوسوم إلى بعض المتغيرات كي تظهر بالشكل المطلوب ، هذا ما يوضحه الجدول التالي :

المتغير	الوصف
type	يحدد هذا المتغير نوع العنصر (مربع نص ، زر خيار ، زر أوامر .. الخ) .
name	يجب أن تحدد اسماً لكل عنصر من عناصر النموذج ، يمكنك القيام بذلك عن طريق هذا المتغير .
value	قد ترغب في إعطاء قيمة افتراضية للعنصر من خلال هذا المتغير .

جدول 7

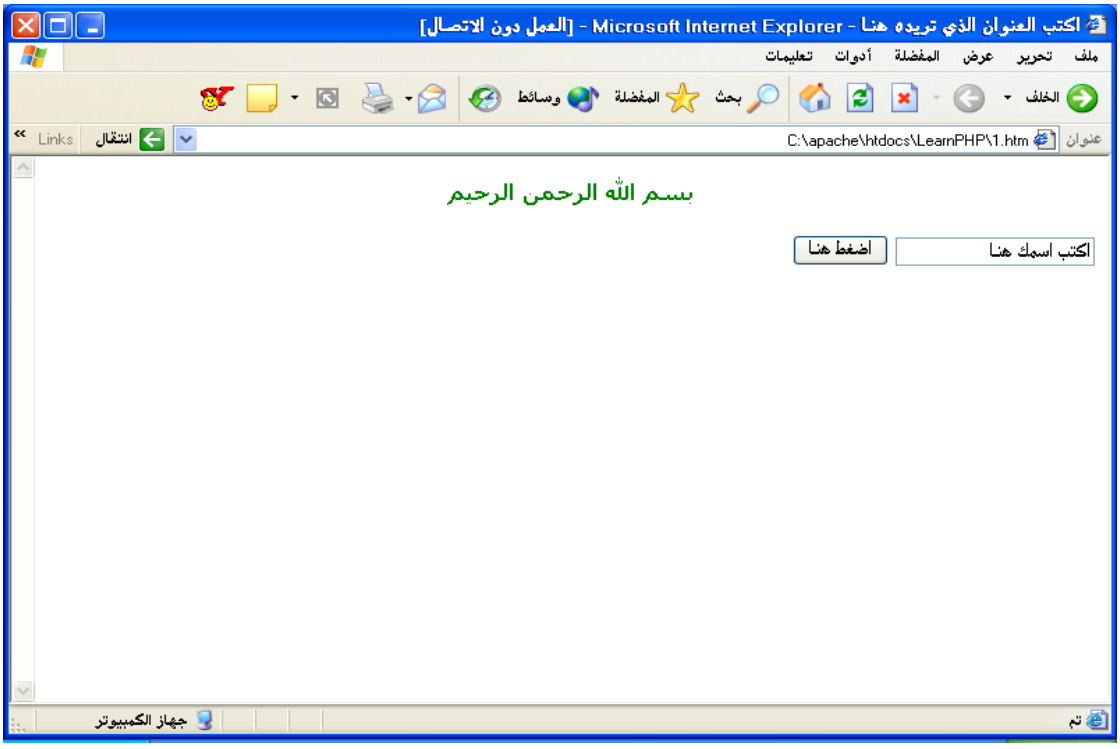
كيف ندرج أحد هذه العناصر ؟ دعنا نتعرف على ذلك سوية عن طريق الأمثلة التالية :

1.htm

```
1  <html dir="rtl">
2  <head>
3        <title>اكتب العنوان الذي تريده هنا</title>
4  </head>
5  <body>
6   <p align="center" dir="rtl">
7    <font face="Tahoma" color="green" size="4">بسم الله الرحمن الرحيم</font>
8   </p>
9  <form method="post" action="form.php">
10  <input type="text" name="text1" value="اكتب اسمك هنا">
11  <input type="submit" value="اضغط هنا" name="submit">
12  </form>
13
14 </body>
15 </html>
```

شكل 26

ستكون النتيجة هكذا :



شكل 27

إن النموذج البسيط أعلاه يحتوي على مربع نص و زر إرسال فقط .. و لكل منهما قيمة افتراضية ظاهرة على الشاشة الأخيرة . إذا جربت الضغط على زر الإرسال فإن المتصفح سيحاول فتح الملف "form.php" و ذلك لأنه الملف المحدد في خاصية action في . قد تتساءل حالياً عن الفرق بين النوعيات المختلفة التي يمكن أن نرسل بها

المعلومات إلى المضيف (راجع جدول 6) لكني سأترك هذا الجزء للباب الرابع من الكتاب بإذن الله .

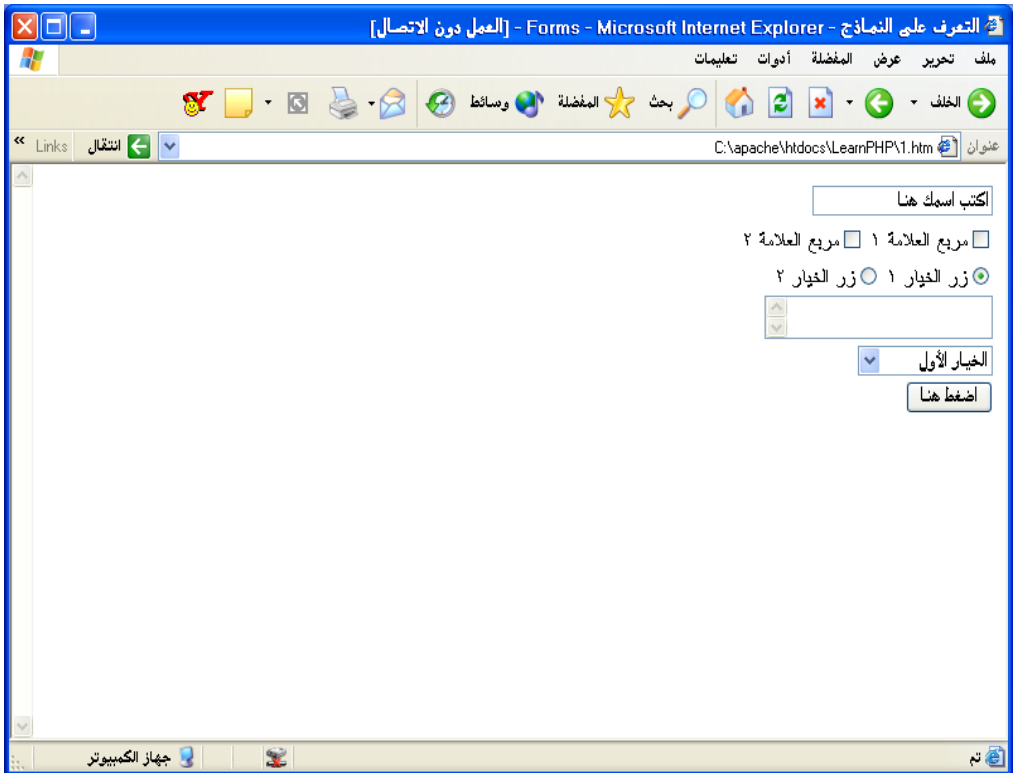
مثال آخر يحتوي على المزيد من العناصر يلي هذا السطر :

1.htm

```
<html dir="rtl">
<head>
      <title>التعرف على النماذج - Forms</title>
</head>
<body>
<form method="post" action="form.php">
<input type="text" name="text1" value="اكتب اسمك هنـــا">
<br>
<input type="checkbox" name="C1" id="a"><label for="a">١ مربع العلامـة</label>
<input type="checkbox" name="C2" id="b"><label for="b">٢ مربع العلامـة</label>
<br>
<input type="radio" value="V1" name="R1" checked id="c"><label for="c">١ زر الخيـــار</label>
<input type="radio" value="V2" name="R1" id="d"><label for="d">٢ زر الخيـــار</label>
<br>
<textarea rows="2" name="S1" cols="20"></textarea>
<br>
<select size="1" name="D1">
  <option value=1>الخيـــار الأول</option>
  <option value=2>الخيار الثـــاني</option>
</select>
<br>
<input type="submit" value="اضغط هنا" name="submit">
</form>
</body>
</html>
```

شكل 28

و هذه هي النتيجة :



شكل 29

سأشرح هذه العناصر بشيء من التفصيل :

```
<input type="checkbox" name="C1" id="a"><label for="a">
مربع العلامة 1</label>
```

يعطينا هذا الوسم مربع العلامة الذي يسمح للمستخدم بوضع علامة "✓" بداخله . هذا العنصر له الاسم "C1" و له المعرّف "a" . بعد ذلك قمنا بكتابة تعريف نصي يظهر أمام المستخدم باستخدام الوسم <label> . لاحظ أن وسم <label> يحتاج لوسم نهاية بينما الوسم <input> لا يحتاج لوسم نهاية .
ميزة مربعات العلامة أنها تسمح للمستخدم باختيار أكثر من خيار في النموذج و ذلك بوضع علامة "✓" في داخل كل خيار يناسب المستخدم ، هذا بالمقارنة مع زر الخيار الذي يسمح للمستخدم باختيار زر واحد فقط كما سنرى بعد قليل .

- = - = - = - = - = - = - = - = -

```
<input type="checkbox" name="C2" id="b"><label for="b">
مربع العلامة 2</label>
```

هذا هو نفس الوسم السابق مع تغيير الاسم و المعرّف فقط ليعطينا مربع علامة جديد يظهر للمستخدم .

- = - = - = - = - = - = - = - = -

```
<input type="radio" value="V1" name="R1" checked
id="c"><label for="c">زر الخيار 1</label>
```

هذا هو الوسم الخاص بزر الخيار . أزار الخيار (radio buttons) هي تلك الدوائر المفرغة التي شاهدناها في الشكل الأخير (شكل 29) و التي تسمح للمستخدم باختيار أحد هذه الأزرار فقط (جرب بنفسك). و كما شاهدنا مع الوسم السابق ، نستخدم الوسم <label> لكتابة تعريف نصي بجوار العنصر .

- = - = - = - = - = - = - = - = -

```
<input type="radio" value="V2" name="R1" id="d"><label
for="d">زر الخيار 2</label>
```

مرة أخرى ، هذا هو وسم زر الخيار . لقد قمت بإعادة كتابة الوسم لتوضيح نقطة هامة يجب أن تتنبه لها هنا . لاحظ اسم عنصري زر الخيار في هذا الوسم و الوسم السابق ، ماذا تلاحظ ؟ إن الوسمين يحملان نفس الاسم

```
name="R1"
```

و هذا شرط أساسي لاستخدام أزرار الخيار حتى تؤدي غرضها بالشكل الصحيح (اختيار خيار واحد فقط) .
جرب تغيير الاسم في الوسم الثاني مثلاً .. و ستكتشف أن المستخدم سيصبح قادراً على اختيار أكثر من خيار في هذه المرة .
قد تحتاج في بعض النماذج إلى استخدام أكثر من مجموعة من أزرار الخيار (يختار المستخدم خياراً واحداً من كل مجموعة) حينها فقط يمكنك تحديد اسم مختلف لكل مجموعة من الأزرار .

-=-=-=-=-=-=-=-

```
<textarea rows="2" name="S1" cols="20">النص هنا
</textarea>
```

تحتاج أحياناً لاستخدام مربع نص مكوّن من عدة أسطر لتتيح مساحة أكبر من التعبير لزوار موقعك ، و هذا ما يقدمه لك هذا الوسم . يمكنك التحكم في عرض و طول مربع النص من خلال المتغيرين (rows) و يشير إلى عدد الصفوف ، (cols) و يشير إلى عدد الأعمدة .

-=-=-=-=-=-=-=-

```
<select size="1" name="D1">
  <option value=1>الخيار الأول</option>
  <option value=2>الخيار الثاني</option>
</select>
```

تعتبر القوائم المنسدلة أحد أفضل الخيارات إذا كنت تفكر في الحصول على استجابة محددة من المستخدم . يمكنك إضافة المزيد من الخيارات عن طريق إضافة سطر يحتوي على الوسم <option> .

كان هذا هو الوسم الأخير في هذا الفصل !
تستطيع الآن أن تأخذ قسطاً من الراحة بعد أن تردد "أنهيت الفصل الأول بنجاح و لله الحمد !"

الخلاصة :

تعرفنا في هذا الفصل على لغة الهتمل . أصبحنا نعرف الهيكل العام لأي صفحة هتمل كما أن لدينا خزينة رائعة من وسوم هذه اللغة . نستطيع الآن أن ننسق نصوصنا بطريقة جيدة كما أننا نستطيع إدراج الصور و الجداول و النماذج لإضافة بعض الحيوية على صفحاتنا .

مراجع إضافية :

- **HTML4 Bible by Bryan Pfaffenberger and Alexis D. Gutzman (IDG Books)**
- **DHTML: The Definitive Reference by Danny Goodman (O'Reilly)**

-

الفصل الثاني : مدخل للـ CSS

أنت تقرأ هذا الكتاب لأنك تخطط لتنفيذ مشاريع بي اتش بي يزيد عدد صفحاتها عن العشرة صفحات ! أليس كذلك ؟

حسناً ... لنتخيل معاً أنك قبلت تنفيذ مشروع متوسط الحجم لأحد العملاء و أنك بطبيعة الحال ستحتاج لبرمجة ما يزيد عن العشر صفحات . لنتخيل أيضاً أنك انتهيت من المشروع بعد عمل متواصل لمدة عشرة أيام و قد عرضت المشروع على العميل لأنك تعتقد أنه أصبح جاهز للاستخدام الآن . ماذا لو فاجأك العميل بطلب تغيير حجم الخط من القيمة "2" إلى القيمة "3" ! هل ستضطر لفتح كل صفحة و التعديل في كل فقرة من فقراتها لتغير هذه القيمة ! ماذا لو كان المشروع مكوناً من 50 صفحة ؟!
و ماذا لو كان يعمل على المشروع أكثر من شخص بشكل منفصل ؟
في الواقع فإن هذه القصة القصيرة التي افتتنا بهذا الفصل و التي ستواجهك كثيراً عند التعامل مع عملاءك تنقلنا للتفكير من استخدام لغة "هتمل" وحدها – و التي تعلمناها في الفصل السابق إلى لغة تعطينا مرونة أكبر .
الفكرة باختصار تكمن في فصل التنسيق عن المحتوى و من ثم وضع التنسيق في ملف منفصل يمكنه تغييره لتتغير كامل صفحات موقعك !
إذاً ، أجد نفسي ملزمة بشرح هذه اللغة قبل البدء بشرح لغة بي تش بي حتى نؤكد على المبدأ الذي نسير عليه من بداية الكتاب : استخدام التقنية المناسبة في المكان المناسب .

تشير الأحرف في اسم اللغة إلى العبارة الإنجليزية التالية :
Cascading **S**tyle **S**heets
و تعني (صفحات الأنماط الانسيابية) . و اختصارها هو : CSS .
ينبغي أن لا يأخذ هذا الفصل من وقتك الكثير ! نصف ساعة من الزمان تعتبر مدة جيدة .

في الواقع فإننا نستخدم نوعاً من أنواع صفحات الأنماط الانسيابية في صفحاتنا دون أن ندري . و لنأخذ هذا المثال البسيط :

2.htm

```
1 <html dir="rtl">
2 <head>
3       <title>مقدمة - الفصل الثاني</title>
4 </head>
5 <body>
6   سيظهر هذا النص باستخدام الأنماط الافتراضية المعدة في متصفح زائر الصفحة
7 </body>
8 </html>
```

شكل 30

كما لاحظت ، لم أقم بتحديد أي تنسيق للنص و مع ذلك فإنه عند فتح الصفحة باستخدام المتصفح ستجد تنسيقاً معيناً لنوع الخط و حجمه و لونه . هذا التنسيق قد يختلف من جهاز لآخر بحسب الإعدادات الافتراضية للعرض . إذاً ، كأننا نقول بأن المتصفح يحتفظ بملف يحوي أنماط معينة من التنسيق يستخدمها إذا لم تحدد التنسيق بشكل واضح في صفحتك . في الواقع ، إن تقنية السي اس اس تعتمد نفس المبدأ . لن تكتب أي تنسيق في صفحة الهتمل الأساسية لكنك ستحدد بالتفصيل التنسيق الذي تريده في ملف منفصل .

لنلقي نظرة على الطريقة بالتفصيل ..

معلومة إضافية

ظهرت الحاجة لاستخدام هذه التقنية – CSS – بعد اهتمام المتصفحات الشهيرة (أمثال : الانترنت اكسبلورر و النت سكيب) بإضافة وسوم هتمل إضافية هي الوسوم الخاصة بتنسيق النص و التي تعرفنا عليها في الفصل السابق . قامت جمعية W3C باعتماد هذه التقنية لتساعد على نقل المواقع قديمة الطراز – تلك التي لا تحتوي على وسوم التنسيق – إلى الطراز الجديد من صفحات إنترنت .

معلومة إضافية

W3C هو اختصار لـ World Wide Web Consortium و هي الجمعية المسؤولة عن إصدار نسخ قياسية من لغة HTML .

الصيغة العامة للنمط :

كما قلنا ، فإن الأنماط تحدد التنسيق لأوسمة الهتمل المختلفة التي تعرفنا عليها في الفصل السابق . تستطيع تحديد تنسيق معين لوسم الفقرة أو تنسيق معين للجداول و هكذا . إذاً لابد أن نتوقع بأن الصيغة العامة لأوسمة السي اس اس ستتضمن وسم هتمل الذي سيطبق عليه النمط و كذلك التنسيق . كما توضح هذه الصيغة تماماً :

وسم هتمل : { المتغير : "القيمة"}

كما نلاحظ فإن الصيغة تقتضي كتابة وسم هتمل الذي سيطبق عليه النمط ثم نقطتان رأسيتان ثم قوسين معقوفين نكتب بداخلها المتغير و القيمة مفصولين بنقطتين رأسيتين أخريتين .

لنأخذ بعض الأمثلة الحقيقية لتتضح الصورة :

```
a           { color: #008000; font-size: 1em }
body        { border-style: solid }
h2          { color: #800000 }
```

```
p               { border-style: solid; border-color: #FF00FF
}
table           { border-right-style: solid }
```

هل تريد تجربة هذه الوسوم بنفسك ؟ انتظر قليلاً حتى ننتهي من شرح الصيغة العامة للأنماط و سنتعرف على طريقة إدراج الأنماط في الجزء التالي إن شاء الله .

دعنا نركز قليلاً على الأمثلة السابقة . هل لاحظت الوسم الأول (a) ؟ إنه يحتوي على متغيرين بدلاً من متغير واحد . قمنا بفصل المتغيرات المختلفة باستخدام الفاصلة المنقوطة (;). يمكنك أن تجعل أنماطك أكثر وضوحاً بكتابة كل متغير في سطر منفصل! هكذا :

```
a {
   color: #008000;
   font-size: 1em
  }
```

أفضل كثيراً أن تستخدم عادات جيدة أثناء كتابة الأنماط أو غيرها من الشفرات لأن ذلك هو طريقك لكتابة برامج يمكن أن نطلق عليها مصطلح "برامج نظيفة" ! أو كما نقول باللهجة المحلية "تفتح النفس" !

تعرفنا في المثال السابق على كيفية تخصيص أكثر من متغير لوسم هتمل واحد . ماذا لو أردنا تخصيص متغير واحد بقيمة واحدة مكررة مع أكثر من وسم هتمل ؟
أكثر الأمثلة التي توضح ذلك هي التي تحدد تنسيق وسوم رأس الفقرة (h1-h6) أو وسوم الرابط التشعبي (a) . لنأخذ مثالاً يوضح ذلك :

```
h1,h2,h3,h4,h5,h6 { color: #008000;  }
```

هل راقبت ذلك ؟ الأمر في غاية السهولة . كل ما عليك هو أن تضيف الفاصلة (,) بين الأوسمة المختلفة ثم تستخدم الصيغة العامة التي أصبحت تعرفها الآن .

استخدام الفئات (Classes) :

لنذهب أبعد من ذلك قليلاً : ماذا لو كنت تريد استخدام أكثر من تنسيق للصفحة الواحدة ، بمعنى أن تنسق الفقرة الأولى في مستندك باستخدام اللون الأحمر و الفقرة الثانية باللون الأخضر و الثالثة باللون الأصفر ! هل تعطيك تقنية السي اس اس هذه المرونة ؟
بالطبع نعم !
كل ما علينا هو أن نتعرف على خاصية الفئات (Classes) ، من خلال المثال التالي :

style.css

```
1 p.red {color : "red"}
2 p.green {color : "green"}
3 p.blue {color : "blue"}
```

شكل 31

تستطيع شرح الطريقة بنفسك ! قمنا بكتابة اسم الوسم المطلوب ، ثم نقطة ، ثم الاسم الذي نختاره للفئة الجديدة و نكمل كما تعودنا .

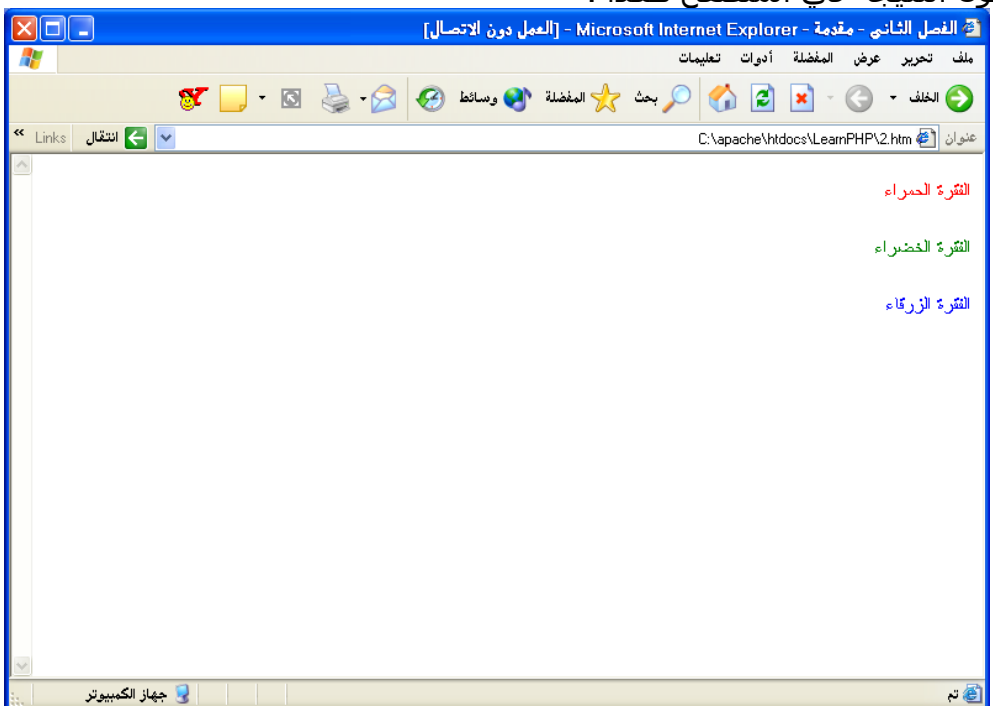
و المثال التالي يعرفنا بالطريقة التي نستخدمها مع أكواد هتمل لتحديد فئة معينة دون أخرى :

2.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مقدمة - الفصل الثاني</title>
4  <style>
5  p.red {color : "red"}
6  p.green {color : "green"}
7  p.blue {color : "blue"}
8  </style>
9  </head>
10 <body>
11 <p class="red">الفقرة الحمراء</p>
12 <p class="green">الفقرة الخضراء</p>
13 <p class="blue">الفقرة الزرقاء</p>
14
15 </body>
16 </html>
```

شكل 32

ستكون النتيجة في المتصفح هكذا :



شكل 33

ملاحظة للتذكير : إذا أردت أن تتعلم بي اتش بي حقاً ، فلا تتكاسل في تنفيذ مثل هذه الأمثلة البسيطة بنفسك ! صدقني ستكتشف أشياء جديدة كلما جربت .

تلميحة ذكية

لتضيف فئة يمكن استخدامها مع جميع الوسوم في صفحتك استخدم الطريقة التالية :

```
.red{ color: “red” }
```

احذف اسم الوسم و اكتفي بوضع نقطة مع اسم الفئة !

طريقة إدراج الأنماط في الصفحة :

تعرفنا على الهيكل العام لصفحات الأنماط الانسيابية ... بقي علينا أن نتعرف على كيفية إدراج هذه الأنماط في صفحات هتمل أو كيفية ربطها بها ؟

هناك ثلاث طرق لإدراج الأنماط في صفحتك :
أ- باستخدام ملف خارجي (external file).
ب- عن طريق رأس الصفحة (Head) .
جـ - داخل الوسم (Inline).

سنقوم بشرح كل طريقة بالأمثلة :

أ- باستخدام ملف خارجي (external file):

هذه الطريقة هي الخيار الذكي لمن يفكر في برمجة مشاريع من عدة صفحات . ضع كل التنسيقات التي تريدها في هذا الملف الخارجي ثم أدرج رابط لهذا الملف في كل صفحة تريد استخدام الأنماط فيها (جميع صفحات مشروعك عادة) .
إذا طلب منك عميلك تغيير لون الخط المستخدم من اللون الأزرق النيلي إلى اللون الأزرق الغامق – و اللذان قد لا يختلفان كثيراً لكن عليك أن ترضي ذوق العميل ! فقط قم بتغيير سطر واحد في هذا الملف و انتهينا .
هذا الملف يجب أن يحفظ بالامتداد css. ، يمكنك كتابته عن طريق برنامج المفكرة، يمكنك الاطلاع على نموذج من هذا الملف في الشكل 31 من هذا الفصل .

استخدم الوسم التالي في رأس كل صفحة من صفحات الهتمل :

2.htm

```
1 <html dir="rtl">
2 <head>
3       <title>الفصل الثاني - مقدمة</title>
4 <link rel="stylesheet" type="text/css" href="style.css" />
5 </head>
```

شكل 34

ب- عن طريق رأس الصفحة (Head) :

تستخدم هذه الطريقة فقط إذا كنت تريد تطبيق تنسيق معين على صفحة واحدة فقط ، بحيث يلزمك تغيير صفحة واحدة فقط إذا أردت أن تعدل على التنسيق . سندرج الأنماط عن طريق الوسم (<style>) في رأس الصفحة (<head>)
لنأخذ مثالاً لصفحة من صفحات هتمل لنتعرف على الطريقة بشكل أوضح :

2.htm

```
1 <html dir="rtl">
2 <head>
3         <title>الفصل الثاني - مقدمة</title>
4 <style>
5 p.red {color : "red"}
6 p.green {color : "green"}
7 p.blue {color : "blue"}
8 </style>
9 </head>
```

شكل 35

جـ - داخل الوسم (Inline) :

هذه الطريقة لا تختلف كثيراً عن استخدام وسوم هتمل للتنسيق ! لأنك تضع التنسيق مع وسم هتمل الذي تريد تنسيقه . ستحتاج لهذه الطريقة لتنسيق وسم خاص لا يتكرر كثيراً مثلاً ! أو لتغيير التنسيق المدرج من صفحة أنماط خارجية و هكذا . ما تحتاجه هو ما يحدد اختيارك !
سنستخدم كلمة style كمتغير يرسل مع الوسم المراد تنسيقه ، المثال التالي يوضح ذلك :

2.htm

```
1 <html dir="rtl">
2 <head>
3         <title>الفصل الثاني - مقدمة</title>
4 </head>
5 <body>
6 <p style="color : red">الفقرة الحمراء</p>
7 </body>
8 </html>
```

شكل 36

معلومة إضافية

هل تتساءل عن سبب تسمية صفحات الأنماط الانسيابية بهذا الاسم ؟!
أستطيع أن أخبرك عن السبب بعد شرحنا للطرق الثلاث المختلفة لإدراج الأنماط . فى

الواقع فإن السبب يعود لإمكانية استخدام أكثر من طريقة لإدراج الأنماط في ملف واحد . بمعنى أن نستخدم ملفاً خارجياً و تستخدم الأنماط في رأس الصفحة و كذلك مع الوسم . لو افترضنا بأنك حددت حجم خط كل فقرة بالحجم "3" في ملف خارجي ثم حددت الحجم "2" لخط الفقرة في رأس الصفحة ثم حددت الحجم "1" مع وسم الفقرات . نقول هنا أن الأنماط ستنساب على بعضها و تكون النتيجة ظهور الفقرة باستخدام الحجم "1" !
الأفضلية تكون باستخدام هذا الانسياب (الأول هو الأعلى أفضلية) :
1-التنسيق مع الوسم .
2-التنسيق في رأس الصفحة .
3-التنسيق في الملف الخارجي .
4-التنسيق الافتراضي في المتصفح .

أشباه الفئات (Pseudo-class):

تعرفنا على الفئات ، فما هي أشباه الفئات يا ترى ؟ إن أقرب مثال يوضح الفكرة هو مثال تنسيق الروابط التشعبية .
تأخذ أشباه الفئات الصيغة العامة التالية :
{"القيمة" : المتغير } شبه الفئة: وسم هتمل
أو (في حالة وجود فئة أيضاً):
{"القيمة" : المتغير }شبه الفئة:فئة. وسم هتمل
قد تبدو مزعجة للوهلة الأولى ! لكنك ستكتشف مدى سهولته بعد المثال التالي :

style.css

```
1 a:link {COLOR: #000088; TEXT-DECORATION: none}
2 a:visited {COLOR: red ; TEXT-DECORATION: none}
3 a:hover {COLOR: pink ; TEXT-DECORATION: underline; font-size: 10pt}
4 a:active {COLOR: lightblue; TEXT-DECORATION: none}
```

شكل 37

في المثال السابق : حددنا اللون (#000088) للرابط العادي الذي لم يزار بعد . كما حددنا تنسيق النص بلا شيء حتى لا يظهر خط سفلي تحت الرابط . حددنا اللون الأحمر للرابط الذي سبقت زيارته ! كذلك حددنا اللون الزهري مع خط بأسفل النص و حجم (10 نقاط) للنص . أما الرابط النشط (الحالي) فقد حددنا له اللون الأزرق الفاتح .
الطريقة سهلة ، جربها بنفسك الآن .

تحذير !

أنصحك بكتابة أنماط الروابط التشعبية بالترتيب الوارد في المثال السابق (link ثم

visited ثم hover ثم active) و إلا فإن الأخريين لن يعملا بالشكل الصحيح !

سنأخذ مثالاً آخر يوضح كيفية استخدام الفئات مع أشباه الفئات في مثال واحد :

style.css

```
1 a.main :link {COLOR: green}
2 a.sub  :link {COLOR: red}
```

شكل 38

المثال يشرح نفسه ، حددنا الفئة (main) للرابط الذي لم يزار باللون الأخضر و في المقابل حددنا الفئة (sub) للرابط الذي لم يزار أيضاً باللون الأحمر . يجب استخدام الطريقة المشروحة في جزء الفئات عند استخدامك إحدى الفئتين بالطبع .

المعرّف (ID) :

تستخدم فكرة المعرّف لحفظ تنسيقات معينة باسم هذا المعرّف على أن يتم إدراجها مع كل وسم يستخدم هذا المعرّف كمتغير ضمن متغيراته الأخرى . لنفرض أننا بحاجة لاستخدام المحاذاة إلى الوسط بشكل متكرر فإننا سنقوم بتعريف معرّف يؤدي هذا الغرض كما يلي :

style.css

```
1 #center {
2     text-align : center
3 }
4
```

شكل 39

و عند الوسم المراد تنسيقه نستخدم الطريقة التالية :

2.htm

```
1 <html dir="rtl">
2 <head>
3       <title>الفصل الثاني - مقدمة</title>
4 <link rel="stylesheet" type="text/css" href="style.css" />
5 </head>
6 <body>
7 <p id="center">نص متوسط</p>
8 </body>
9 </html>
```

شكل 40

إذن ، كما لاحظت بنفسك ، خاصية المعرّف تضيف لك مصمم قدرات إضافية يمكنك استخدامها متى ما احتجت إليها .

التوارث:

ذكرنا سابقاً أن الصفحة الواحدة قد تحتوي على الطرق الثلاث المختلفة لإدراج الأنماط . و أن المتصفح سيستخدم التنسيق الأقرب إلى الوسم (مع الوسم ، ثم في رأس الصفحة ، ثم في ملف خارجي) . فإذا كان الملف الخارجي يكتب الفقرة باللون الأحمر و النمط في رأس الصفحة يكتبها باللون الأصفر و النمط المرفق مع الوسم يكتبها باللون الخضر فإن الغلبة ستكون للون الأخضر . حسناً ، نريد أن ننتقل لأبعد من ذلك : لنفرض أن الملف الخارجي يحدد اللون الأحمر و الحجم "3" و أن النمط في رأس الصفحة يحدد اللون الأصفر و المحاذاة إلى الوسط و أن النمط المرفق مع الوسم يحدد اللون الأخضر . ماذا سيحدث الآن ؟ نعرف مسبقاً أن اللون سيكون أخضراً ... لكن ماذا عن المحاذاة و الحجم اللذان لم يتدخل النمط المرفق مع الوسم في تحديدهما ؟ الجواب هو التوارث . التوراث يعني أن النمط النهائي سيكون لون أخضر ، محاذاة إلى اليمين ، و حجم "3" بمعنى أن التنسيقات هنا متوارثة من الأنماط البعيدة .
لنأخذ مثالاً لتتضح الصورة أكثر :
الملف الخارجي :

style.css

```
1  p {
2      color   : red;
3      font-size   : 20pt
4  }
5
```

شكل 41

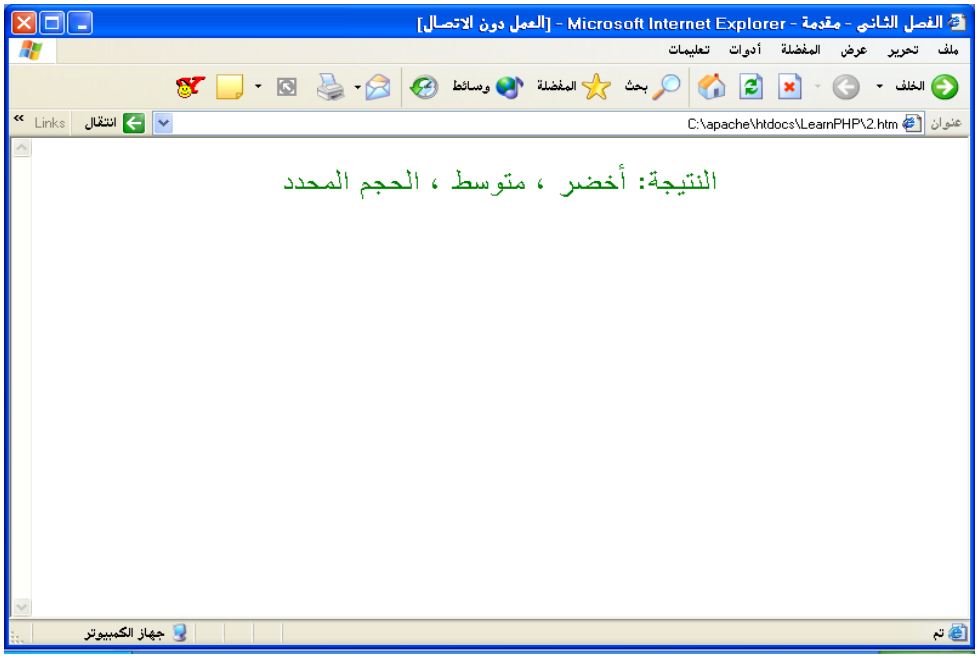
صفحة الهتمل :

2.htm

```
1  <html dir="rtl">
2  <head>
3        <title>مقدمة - الفصل الثاني</title>
4
5  <style>
6  p {
7      text-align : center;
8      color  : yellow
9  }
10 </style>
11 <link rel="stylesheet" type="text/css" href="style.css" />
12 </head>
13 <body>
14 <p style="color:green">النتيجة: أخضر ، متوسط ، الحجم الحدد</p>
15 </body>
16 </html>
```

شكل 42

النتيجة ستكون هكذا :



شكل 43

الملاحظات :

إذا كانت لديك بعض المعلومات عن لغات البرمجة – تطبيقات سطح المكتب أو تطبيقات الويب – فإنك حتماً معتاد على فكرة الملاحظات . الملاحظات هي نوع من أنواع التوثيق التي تتيح لك كتابة تلميحة صغيرة في أول كل صفحة أو بجانب الأسطر الغامضة بعض الشيء و هكذا بحيث يسهل عليك فهم ما كتبته عند الرجوع إليه بعد مدة . و بالطبع فإنه ينصح دائماً بتوثيق برامجك ، سوى كنت ستستخدمها لأغراض التوزيع العام أو لأغراض الاستخدام الشخصي لأن ذلك يحفظ الكثير من وقتك و وقت من يقرأ برامجك .
لإدراج ملاحظة في ملف السي اس اس ، قم باتباع الطريقة الموضحة في المثال التالي :

style.css

```
1 /* يمكنك وضع الملاحظات في أول الصفحة*/
2 /* يمكن أن تتجاوز
3   الملاحظة
4   حدود السطر الواحد
5 */
6 #center {
7     text-align : center
8     /* كما يمكنك وضع الملاحظات داخل الأوسة */
9 }
```

شكل 44

المزيد من الأمثلة :

حتى الآن ، تناولنا الصيغة العامة لاستخدام أنماط السي اس اس ، تناولنا كيفية إدراج هذه الأنماط في صفحاتنا ، تكلمنا عن أمور أخرى مثل المعرفات و التوارث و الملاحظات

، لكن ما زلنا بحاجة لمزيد من الأمثلة التي تجعلنا معتادين على استخدام الأنماط مع أي وسم من وسوم هتمل . السبب في التركيز على المزيد من الأمثلة أنه و كما جاء معنا في بعض الأمثلة السابقة ، نلاحظ اختلاف متغيرات السي اس اس عن بعض متغيرات هتمل ! فإذا كان حجم الخط في وسوم هتمل يرمز له بـ (size) فإننا نرمز له بـ (font-size) في ملفات السي اس اس . إذن ، الأفضل أن نأخذ كل وسم من وسوم هتمل – التي تعرفنا عليها في الفصل السابق – و نتعرف على المتغيرات المناسبة لها . سنأخذها على شكل جداول للتسهيل :
(هذه الجداول ليست للحفظ ! احتفظ بها كمرجع فقط بعد أن تطلع عليها و تجرب بعضها بنفسك)

<body>

المتغير	الوصف
font-family	يحدد هذا المتغير نوع الخط المستخدم .
font-size	يحدد هذا المتغير حجم الخط المستخدم .
color	يحدد هذا المتغير لون الخط المستخدم .
Background-color	يحدد هذا المتغير لون الخلفية .
Background-image	يحدد هذا المتغير إحدى الصور كخلفية للصفحة .
Background-repeat	يحدد هذا المتغير كيفية تكرار صورة الخلفية و يأخذ إحدى القيم التالية : (repeat, repeat-x, repeat-y, no-repeat)
SCROLLBAR-BASE-COLOR	يحدد هذا المتغير لون القاعدة لأشرطة التمرير الطولية و العرضية .
SCROLLBAR-ARROW-COLOR	يحدد هذا المتغير لون السهم لأشرطة التمرير الطولية و العرضية .
Scrollbar-track-color	يحدد هذا المتغير لون الخلفية لأشرطة التمرير الطولية و العرضية .
Scrollbar-face-color	يحدد هذا المتغير لون الواجهة لأشرطة التمرير الطولية و العرضية .
Scrollbar-highlight-color	يعطينا هذا المتغير بعض الإضاءة على أشرطة التمرير (بحسب اللون المستخدم)
Scrollbar-3dlight-color	يعطي هذا المتغير التأثير الثلاثي على أشرطة التمرير .
Scrollbar-darkshadow-color	يحدد هذا المتغير لون الظل لأشرطة التمرير الطولية و العرضية .
scrollbar-shadow-color	يحدد هذا المتغير لون الظل لأشرطة التمرير الطولية و العرضية (يختلف قليلاً عن المتغير السابق).

جدول 8

تحذير !

إذا كنت قد حددت تنسيقاً معين للنص باستخدام الوسم (body) ثم وضعت النصوص

داخل جداول ، فإن المتصفح لن يأخذ تنسيق الوسم (**body**) بعين الاعتبار . يجب أن تحدد تنسيق الخط مع الوسم (**table**) كما سنرى بعد قليل .

<table>

المتغير	الوصف
font-family	يحدد هذا المتغير نوع الخط المستخدم داخل الجدول .
font-size	يحدد هذا المتغير حجم الخط المستخدم داخل الجدول .
color	يحدد هذا المتغير لون الخط المستخدم داخل الجدول .
Background-color	يحدد هذا المتغير لون خلفية الجدول.
Background-image	يحدد هذا المتغير إحدى الصور كخلفية للجدول .
Border-color	يحدد هذا المتغير لون الإطار (الحدود الأربعة)
Border-width	يحدد هذا المتغير سمك الإطار (الحدود الأربعة) ، قد يأخذ أحد القيم التالية : (thin, medium, thick, رقم)
Border-style	يحدد هذا المتغير شكل الإطار (الحدود الأربعة) ، قد يأخذ أحد القيم التالية : (none, hidden, dotted, dashed, solid, double, inset, outset)
Border-left-color	يحدد هذا المتغير لون الحد الأيسر للإطار .
Border-left-width	يحدد هذا المتغير سمك الحد الأيسر للإطار .
Border-right-color	يحدد هذا المتغير لون الحد الأيمن للإطار .
Border- right-width	يحدد هذا المتغير سمك الحد الأيمن للإطار .
Border-top-color	يحدد هذا المتغير لون الحد العلوي للإطار .
Border-top-width	يحدد هذا المتغير سمك العلوي الأيسر للإطار .
Border-bottom-color	يحدد هذا المتغير لون الحد الأسفل للإطار .
Border-bottom-width	يحدد هذا المتغير سمك الحد الأسفل للإطار .

جدول 9

<ol>,<ul>

المتغير	الوصف
font-family	يحدد هذا المتغير نوع الخط المستخدم في القوائم .
font-size	يحدد هذا المتغير حجم الخط المستخدم

	في القوائم.
color	يحدد هذا المتغير لون الخط المستخدم في القوائم.
list-style-image	يحدد هذا المتغير صورة معينة كمحدد للقائمة (يأخذ مسار الصورة كقيمة له) .
list-style-position	يحدد هذا المتغير مكان محدد القوائم ، يأخذ القيمة: (Inside, outside)
list-style-type	يحدد هذا المتغير شكل محدد القوائم ، يمكن أن يأخذ أحد القيم التالية : (None, disc, circle, square, decimal, lower-alpha, upper-alpha, lower-latin, upper-latin)

جدول 10

تحذير !

طالما أن الشبكة موجودة ، فإن المطورين لن يتوقفوا عن إضافة و حذف بعض المتغيرات من وقت لآخر !
كما أن المتصفحات المختلفة تختلف في دعمها للمتغيرات المختلفة و قيمها ، فإذا جربت أحد القيم المذكورة في هذه الجداول و لم تظهر لك أي نتيجة ، فقد يكون السبب أن متصفحك لا يدعم ذلك ، إما تقصيراً من الشركة المنتجة للمتصفح ! أو لأن جمعية w3c قد أسقطت المتغير من القائمة القياسية للوسوم .
لذلك ! دائماً جرب بنفسك كل الأمثلة في هذا الكتاب . ستستفيد الكثير و لن تخسر شيئاً !

<input>

المتغير	الوصف
font-family	يحدد هذا المتغير نوع الخط المستخدم في عناصر النماذج (مربع نص، زر الخيار) .
font-size	يحدد هذا المتغير حجم الخط المستخدم في عناصر النماذج (مربع نص، زر الخيار).
color	يحدد هذا المتغير لون الخط المستخدم في عناصر النماذج (مربع نص، زر الخيار).
border	يحدد هذا المتغير شكل الإطار ، إنها صيغة مختصرة يمكن استخدامها مع عناصر النماذج و غيرها ، حدد جميع خصائص الإطار بالشكل التالي كمثال : (border: 1px solid #000000;)
background-color	يحدد هذا المتغير لون خلفية عنصر النموذج .

الخلاصة :

تعرفنا في هذا الفصل على تقنية الـ CSS بشكل سريع ! أصبحت تعرف كيفية الاستفادة من هذه التقنية في وقت قصير لتحفظ أوقات طويلة من التعديل ! تعرفنا على الصيغة العامة لمتغيرات السي اس اس ، كما تعرفنا على الطرق الثلاثة لإدراج الأنماط في صفحاتنا . تعرفنا على الفئات (عد سريعاً إلى العنوان الفرعي : الفئات إن كنت قد نسيت المقصود بالفئات !) . كما تعرفنا على أشباه الفئات و المعرفات و طريقة إدراج الملاحظات في الأنماط .
لقد كان الفصل ممتعاً ! أضاف لمعلوماتنا الكثير خلال وقت قصير . أليس كذلك ؟

المراجع :

إذا كنت مهتماً بتعلم المزيد عن صفحات الأنماط الانسيابية ، يمكنك الرجوع لبعض هذه الكتب :

- **Cascading Style Sheets – The Defintive Guide by Eric A. Meyer (O'Reilly & Associates)**
- **Core CSS: Cascading Style Sheets by Keith Schengili-Roberts (Prentice Hall PTR)**
- **Cascading Style Sheets: Designing for the web by Hakon Wium Lie, Bert Bos, Robert Cailliau (Addison-Wesley Pub Co)**

الفصل الثالث : مدخل للجافا سكريبت Java Script

ها قد انتهينا من تعلم لغتين ! و وصلنا بحمد الله للغة الثالثة – المحطة الأخيرة قبل البدء بتعلم البي اتش بي . لماذا برأيك وضعت هذا الفصل كمدخل في كتاب لتعليم لغة بي اتش بي ؟ بالتأكيد ليس لزيادة عدد صفحات الكتاب !

في الواقع سنجني فوائد جمة من خلال تصفح هذا الفصل ، إليك بعض الأهداف التي نطمح إلى تحقيقها من وراء هذا الفصل :

- سنحقق الانتقال المنطقي من مستوى تصميم المواقع بواسطة لغة الهتمل و تقنية السي اس اس من جهة إلى مستوى تطوير المواقع بواسطة لغة بي اتش بي ، كيف ذلك ؟ تعتبر لغة الجافا سكريبت من المستوى المتوسط بين المستويين السابقين – سواءً من ناحية السهولة و الصعوبة ، أو من ناحية القدرات و الإمكانيات التي توفرها . كما أنها جاءت بين المستويين السابقين من الناحية التاريخية فقد بدأ مطورو مواقع ويب باستخدام الجافا سكريبت قبل أن تظهر بي اتش بي على الساحة . لذلك ، أريد حقاً أن تتبع الطريق الذي استخدمته شخصياً للوصول لـ بي اتش بي و هو ذات الطريق الذي اتبعه أشهر مطورو المواقع كذلك .
- تعتبر الجافا سكريبت لغة برمجة من الوزن الخفيف ، بمعنى أن لديها قدرات لغات البرمجة الأخرى لكن باستبعاد القدرات العالية التي تقدمها لغات البرمجة الحقيقية . إذن ، ألا تعتقد أن الجافا سكريبت ستكون مدخلاً جيداً بالنسبة لكل من يقرأ هذا الكتاب دون أن يكون لديه خلفية برمجية سابقة ؟ بالتأكيد الإجابة نعم ! و الأجمل أن الجافا سكريبت و بي اتش بي تتفقان في كثير من الصيغ العامة التي سنتعرف عليها بعد قليل .
- هل تذكر قاعدة التقنية المناسبة في المكان المناسب ؟ لقد ناقشناها في الفصول السابقة من هذا الكتاب ، مرة أخرى : نحن نتعلم الجافا سكريبت في هذا الكتاب لأننا لن نستخدم بي اتش بي في كل صغيرة و كبيرة في مشاريعنا ! فإذا كانت بي اتش بي تقدم ميزة تستغرق 20 جزء من الثانية حتى يكتمل ، بينما تقدم الجافا سكريبت نفس الميزة في 10 أجزاء من الثانية ، وقتها يجب أن تكون ذكياً بما فيه الكفاية و تختار الجافا سكريبت ، حتى لو كنت تعشق بي اتش بي .. أليس كذلك ؟ غالباً سنستفيد من الجافا سكريبت في مشاريعنا في الجزء الذي يتطلب التأكد من صحة مدخلات المستخدم في النماذج (Form Validation) .

إذن ، هل أنت مستعد لتعلم الجافا سكريبت الآن ؟ سيأخذ منك هذا الفصل وقتاً أكثر من الفصلين السابقين غالباً ، لكن لا تحاول أن تتخطى الفصل للفصول التالية قبل أن تكمل ! هذه هي نصيحتي الشخصية لك ، و الخيار لك بالنهاية .

سأحاول قدر الإمكان أن أرتب هذا الفصل بنفس الترتيب العام لفصول الكتاب التي تتناول أساسيات بي اتش بي ، حتى لا يختلف عليك التسلسل فقط .
قبل أن نبدأ بالتعرف على الصيغ العامة لكتابة أكواد الجافا سكريبت ، سنتناول بعض المعلومات العامة عن الجافا سكريبت على هيئة نقاط :

- تم تطوير الجافا سكريبت بواسطة شركة نت سكيب (Netscape) . كانت تدعى **Live Script** في البداية ، و مع انتشار لغة الجافا التي طورتها شركة

Sun انضم فريق تطوير لغة الـ **Live Script** مع فريق عمل شركة Sun ، خرجوا لنا بلغة جديدة ! و أطلقوا عليها مسمى **Java Script** ليستفيدوا من الانتشار الواسع للغة الجافا . و هذا يفسر تشابه الجافا و الجافا سكريبت في كثير من الأمور .

- تختلف الجافا سكريبت **Java Script** عن كل من لغة الجافا **Java** و لغة الجاي سكريبت **JScript** . فالأولى من إنتاج شركة Netscape و الثانية من إنتاج شركة Sun و الثالثة من إنتاج شركة مايكروسوفت .
- الجافا سكريبت هي لغة برمجة لتطبيقات الويب فقط ! سنضيف سطور الجافا سكريبت في نفس الصفحات التي تحتوي على وسوم الهتمل .
- تدعم أغلب المتصفحات الحديثة أكواد الجافا سكريبت ، مما يجعلها تتغلب على الفيجوال بيسك سكريبت المدعومة من قبل متصفح الإنترنت اكسبلورر فقط .
- نحن نحتاج لاستخدام بعض أكواد الجافا سكريبت – بالإضافة إلى لغة هتمل – لأن لغة الهتمل وحدها تعطينا صفحات ثابتة غير ديناميكية أما الجافا سكريبت فإن لها القدرة على إضافة محتويات تتغير تلقائياً بتغير الوقت ، نوع المتصفح ، قيمة مدخلات المستخدم و هكذا .
- تتمتع الجافا سكريبت بالقدرة على التحكم بالنتيجة بناء على الحدث (event driven) . يمكنك تنفيذ أمر معين عند الضغط على زر معين ، عند مرور مؤشر الفأرة فوق كائن معين ، عند تحميل الصفحة ، أو عند إغلاقها – هذه بعض الأمثلة على الأحداث .
- هناك نوعان من لغات برمجة الويب :
  - جهة العميل (Client Side): و العميل في حالتنا - غالباً - هو المتصفح الذي تستخدمه لاستعراض الصفحة . هذا النوع من اللغات يعتمد على تنفيذ سطوره على جهاز المستخدم و ليس على المضيف الذي يحفظ الصفحة .
  - جهة المضيف (Server Side): و المضيف هو جهاز الشركة التي تقدم خدمة الاستضافة لموقعك في الغالب . تحتاج هذه اللغات لوقت أطول حتى تصل النتيجة للمستخدم النهائي بطبيعة الحال لذلك نبتعد عنها إذا توفر البديل .

تصنف الجافا سكريبت ضمن اللغات التي تنفذ جهة العميل . (سنفصّل الحديث عن هذين النوعين في الفصول القادمة بإذن الله).

لقد تكلمنا كثيراً ! لنبدأ بكتابة بعض الأمثلة البسيطة ...

الصيغة العامة لإدراج كود الجافا سكريبت إلى صفحتك :

ذكرنا أننا سنكتب أسطر الجافا سكريبت داخل أسطر الهتمل ، و سنضيف الآن معلومة بسيطة أخرى . إننا نستخدم أحد وسوم هتمل الخاصة الذي يتيح لنا إدراج أكواد للغات خارجية ، ألا و هو الوسم **<script>** هذا الوسم له وسم بداية و وسم نهاية ، كما أنه يأخذ متغير واحد هو اسم اللغة الخارجية التي سنستخدمها . كما أنه قد يأخذ متغيراً آخر إذا كانت أسطر الجافا سكريبت مكتوبة في ملف خارجي .

إذن هناك طريقتان لإدراج الجافا سكريبت :

1. إدراج الجافا سكريبت في نفس الصفحة :

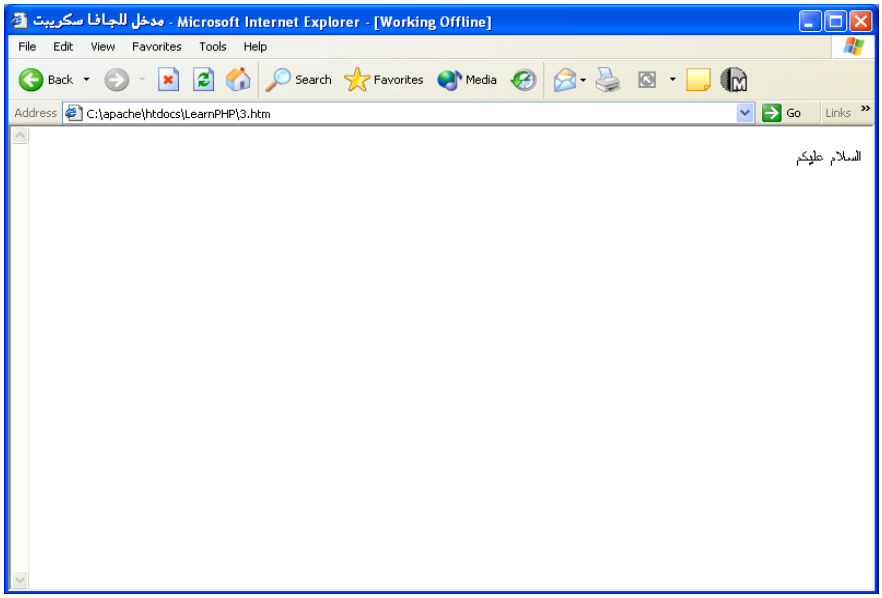
لنراقب المثال التالي :

3.htm

```
1  <html dir="rtl">
2  <head>
3        <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6     <script language="javascript">
7     document.write("السلام عليكم");
8     </script>
9  </body>
10 </html>
```

شكل 45

ستكون النتيجة كالتالي :



شكل 46

ليس هناك الكثير لنشرحه عن المثال أعلاه ! فالسطر رقم 6 يحتوي على الأمر <script> و قد أخذ المتغير language القيمة JavaScript للدلالة على أن اللغة المستخدمة هي لغة الجافا سكريبت .
السطر رقم 7 يحتوي على سطر كتب بلغة الجافا ، إنه يقوم بكتابة عبارة "السلام عليكم" على شاشة المتصفح . لاحظ أن السطر خُتِم بالفاصلة المنقوطة (;) . أريد أن أقول هنا أن المبرمجين في أغلب لغات البرمجة يضعون الفاصلة المنقوطة في نهاية كل سطر برمجي للدلالة على نهايته ، هذه الفاصلة المنقوطة شرط أساسي في اللغات الأخرى و لا يعتبر الكود صحيحاً بدونها ، لكنها على العكس من ذلك اختيارية فقط في الجافا سكريبت و لا تكون أساسية إلا إذا أردت كتابة أكثر من أمر في سطر واحد فعليك حينئذ أن تفصلهم بفاصلة منقوطة .
السطر رقم 8 يحتوي على وسم النهاية </script> .
ألم أقل لك بأن الأمر غاية في السهولة !

معلومة إضافية

يمكنك إدراج الجافا سكريبت بنفس الأمر السابق مع تغيير بسيط ، هكذا :

```
<Script type="text/JavaScript">
الأمر هنا
</script>
```

النتيجة واحدة في النهاية ، اختر ما يعجبك !!

2. إدراج الجافا سكريبت من ملف خارجي :

تحدثنا سابقاً عن تقنية السي اس اس – الفصل الثاني – و تحدثنا عن ميزة كتابة الكود في ملف خارجي و إدراج اسمه فقط في الملفات التي تستخدمه . لقد كانت الميزة الأساسية هي عدم الحاجة لتكرار نفس الكود في أكثر من موضع و بالتالي الحاجة للتعديل في أكثر من ملف عند الحاجة لتعديل شيء بسيط في الكود .
و الآن لدينا ذات الميزة ، و ذات السبب . و الطريقة غاية في السهولة ، سنشرحها من خلال المثال الآتي :
أولاً ، كيفية إدراج الملف الخارجي في صفحة الهتمل :

3.htm

```
1 <html dir="rtl">
2 <head>
3       <title>مدخل للجافا سكريبت</title>
4 </head>
5 <body>
6   <script language="javascript" src="script1.js"></script>
7 </body>
8 </html>
```

شكل 47

كما نلاحظ ، لقد استخدمنا المتغير src لتمرير اسم الملف الخارجي . ثم أغلقنا الوسم دون كتابة أي شيء داخله .

ثانياً ، الملف الخارجي :

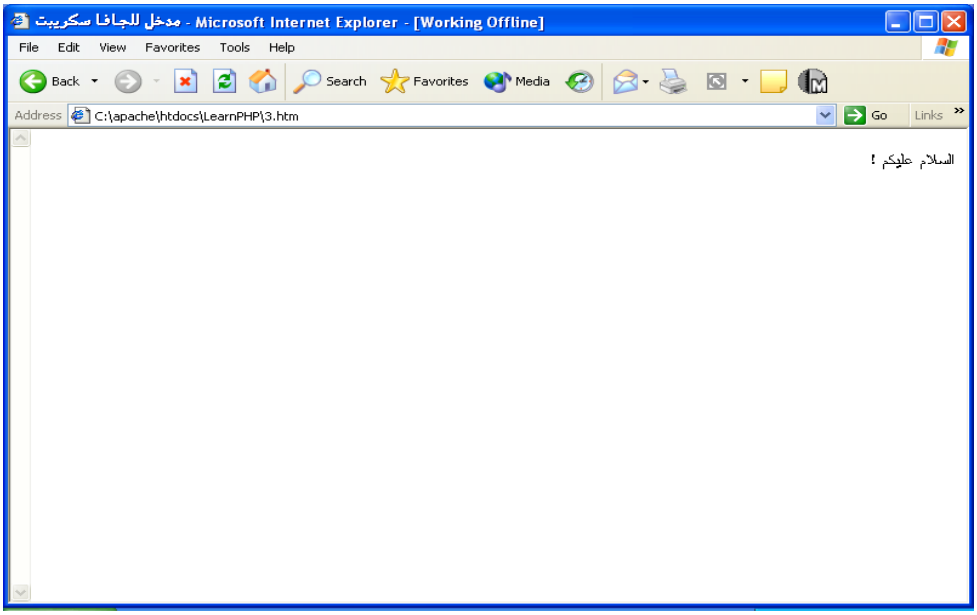
script1.js

```
1 document.write ("السلام عليكم !");
```

شكل 48

هناك عدة أشياء يجب أن نلاحظها في المثال السابق ، أولاً بالنسبة لاسم الملف ، لابد أنك لاحظت الامتداد js في نهاية الملف ، هذا هو امتداد ملفات الجافا سكريبت الخارجية . ثانياً نلاحظ أننا قمنا بكتابة الأسطر البرمجية وحدها بدون وسم <srcipt> ، إن إضافة هذا الوسم داخل الملف الخارجي يعتبر خطأ برمجي ! و أخيراً بالنسبة للفاصلة المنقوطة ، هي اختيارية كما قلنا سابقاً .

ثالثاً ، النتيجة ستكون كالتالي :



تحذير !

لتتجنب حدوث مشاكل مع المتصفحات التي لا تدعم الجافا سكريبت ، كل ما عليك هو أن تضيف الكود داخل وسم الملاحظة من الهتمل ، هكذا :

```
<Script type="text/JavaScript">
<!--
الأمر هنا
//-->
</script>
```

إدراج كود الجافا سكريبت في رأس الصفحة :

شاهدنا في الأمثلة السابقة طريقة إدراج الجافا سكريبت في جسم الصفحة مباشرة (body) حيث ظهرت نتيجة الكود على شاشة المتصفح مباشرة أيضاً . و الواقع أنه ليس المكان الوحيد الذي يمكننا إدراج كود الجافا سكريبت فيه ، حيث يمكننا إدراجه في رأس الصفحة حيث لا ينفذ مباشرة و لكن ينفذ عن طريق استدعاءه لاحقاً . غالباً ما نستخدم هذه الطريقة عند تعريف دالة (function) معينة . و بما أن رأس الصفحة هو

الجزء الذي يتم تحميله أولاً ، فإننا نضمن تحميل الدالة عند المستخدم قبل استدعائها .

لاحظ أنه يمكننا إدراج الجافا سكريبت في رأس الصفحة ، في جسم الصفحة ، أو في كلاهما .. أي أنه يمكنك إدراج أكثر من كود جافا سكريبت واحد في صفحة واحدة .
لنأخذ مثالاً يوضح كيفية إدراج الجافا سكريبت في رأس الصفحة :

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  <script language="javascript">
5  الأوامر هنا
6  </script>
7  </head>
8  <body>
9  سنقوم بإستدعاء الكود هنا فيما بعد
10 </body>
11 </html>
```

شكل 49

لم يختلف الأمر كثيراً كما نلاحظ ، كل ما تغير هو مكان إدراج الوسم <script> فقط .

سنبدأ الآن بشرح الأشياء التي يمكنك إدراجها في أسطر الكود نفسه ، سنتعلم البرمجة ذاتها الآن !

المتغيرات :

المتغير هو مخزن مؤقت لخزن المعلومات (حروف ، أرقام ..الخ) . يصبح هذا المخزن متوفراً لك عندما تقوم بتعريفه في صفحتك . لماذا نحتاج لهذه المخازن في برامجنا ؟ لنفرض أننا نريد من المستخدم أن يدخل اسمه ، ثم نقوم نحن بطباعة هذا الاسم . نحن إذن لا نعرف اسم المستخدم مسبقاً ! لكننا يجب أن ننتهي من كتابة الكود قبل أن نسلم البرنامج للمستخدم . إذاً لابد أن نستخدم اسم المخزن الذي سنستقبل فيه اسم المستخدم بدلاً من الاسم ذاته – الذي لا نعرفه بعد . عندما يقوم المستخدم بتشغيل البرنامج – أو فتح الصفحة – و يقوم بإدخال اسمه فإن هذا الاسم سيتم تخزينه في مخزن مؤقت – ينتهي بإغلاق الصفحة – ثم يقوم البرنامج بعرض الاسم الذي تم تخزينه في المخزن . إذن ، نستطيع أن نعتبر المتغيرات مجرد صناديق سوداء مغلقة : قد تحتوي على قيمة ، و قد تحتوي على لا شيء أيضاً ! تبدأ حياة هذه الصناديق منذ تعريفها في الصفحة و تنتهي حياتها بإغلاق الصفحة .

لابد أنك تسأل نفسك الآن : كيف أقوم بتعريف المتغير ؟!

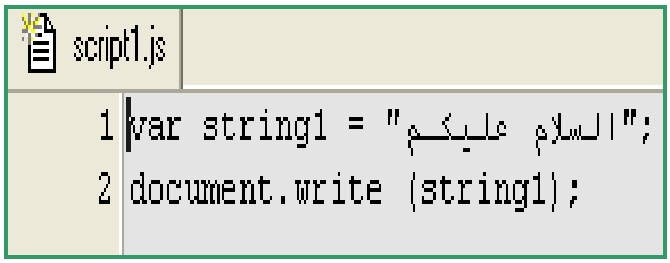
كالعادة ! الأمر غاية في السهولة أيضاً .. سنستخدم الأمر var ثم اسم المتغير ثم علامة المساواة ثم قيمة المتغير الابتدائية ، هكذا :

```
Var variable_name = value
```

كما تستطيع تعريف المتغير بدون كتابة الكلمة var :

variable_name = value

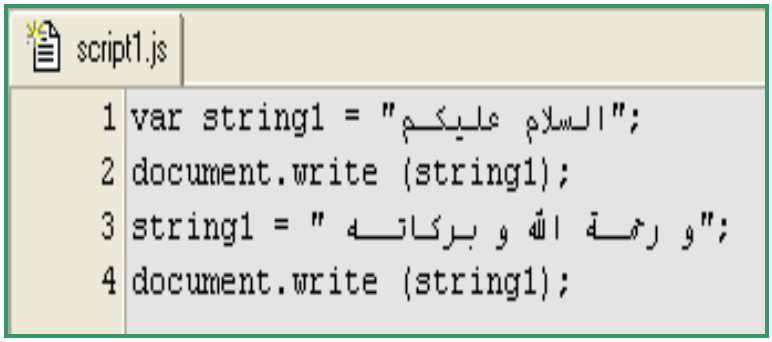
مثال :



شكل 50

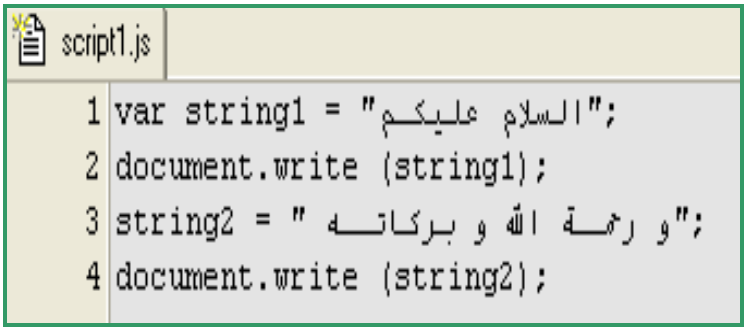
قمنا بتعريف المتغير string1 و خزنا القيمة "السلام عليكم" في هذا المتغير ثم قمنا بطباعة المتغير على الشاشة .

و الآن سنضيف بعض الأسطر إلى الكود السابق ، سنغير القيمة المخزنة ثم نعيد طباعتها :



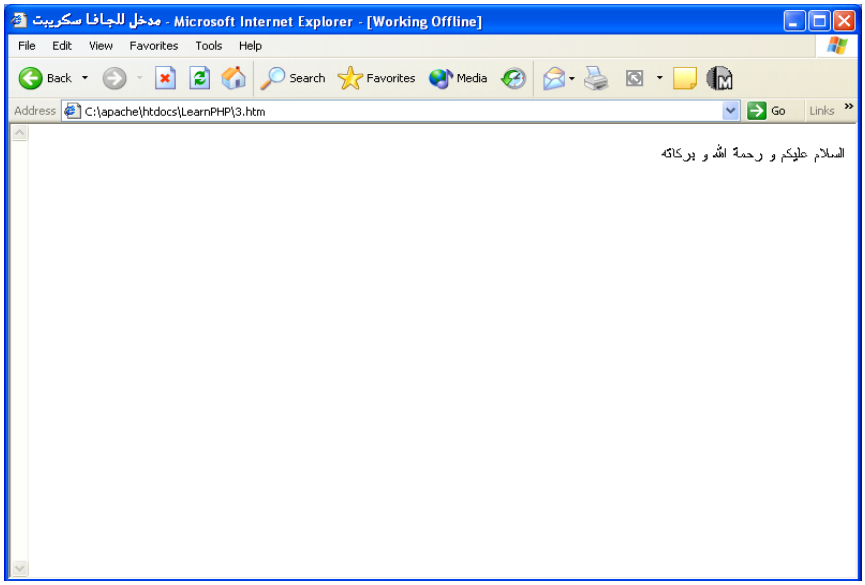
شكل 51

تستطيع بالطبع استخدام أكثر من متغير في الصفحة الواحدة (لاحظ الأرقام في أسماء المتغيرات):



شكل 52

ستكون النتيجة في الحالتين السابقتين هكذا :



شروط تسمية المتغيرات :

تفرض علينا لغات البرمجة المختلفة شروطاً معينة على الأسماء التي نختارها لمتغيراتنا .. و كذلك الحال بالنسبة للجافا سكريبت ، فيجب أن نراعي الأمور التالية :

- يتكون اسم المتغير من حروف أو أرقام أو كلاهما ، كما يمكن أن يحتوي على رمز الشرطة السفلية (_) .
- يجب أن يبدأ بحرف أو برمز الشرطة السفلية .(مثال : 1var يعتبر اسم خاطئ)
- يختلف اسم المتغير باختلاف حالة الأحرف (case sensitive). (مثال : var1 يختلف عن VaR1)
- يجب أن لا يكون كلمة محجوزة (reserved word) في الجافا سكريبت (مثال : var يعتبر اسم خاطئ لأنه عبارة عن أمر معروف في الجافا سكريبت)

المعاملات :

هي عبارة عن الإشارات المستخدمة لتنفيذ العمليات الرياضية أو الحسابية و كذلك العمليات المنطقية . (بعض الأمثلة على المعاملات تشمل : + ، - ، * ، &&)

سنتحدث في هذا الجزء عن أنواع المعاملات ، طريقة استخدامها ، أفضلية الترتيب ، كل ذلك من خلال أمثلة واضحة كما سنرى :

أنواع المعاملات :

أ- معاملات حسابية :

هي تلك المعاملات التي تنفذ عمليات حسابية رياضية كالجمع و الطرح و الضرب و القسمة و غيرها ..
الجدول التالي يقدم قائمة بهذه المعاملات :

المعامل	العملية
()	يمكنك استخدام الأقواس للتغلب على أسبقية تنفيذ عملية على أخرى .

*	يقوم هذا المعامل بعملية ضرب عددين، مثال : X=1;y=2;z=x*y; النتيجة : المتغير z يحتوي على القيمة 2
/	يقوم هذا المعامل بعملية قسمة عددين، مثال : X=4;y=2;z=x/y; النتيجة : المتغير z يحتوي على القيمة 2
%	يقوم هذا المعامل بعملية قسمة عددين و الاحتفاظ بباقي القسمة، مثال : X=3;y=2;z=x%y; النتيجة : المتغير z يحتوي على القيمة 1
+	يقوم هذا المعامل بعملية جمع عددين، مثال : X=1;y=2;z=x+y; النتيجة : المتغير z يحتوي على القيمة 3
-	يقوم هذا المعامل بعملية طرح عددين، مثال : X=1;y=2;z=x-y; النتيجة : المتغير z يحتوي على القيمة -1
++	يقوم هذا المعامل بإضافة واحد إلى قيمة المتغير الذي يسبقه ، مثال : X=1;x++; النتيجة : المتغير x يحتوي على القيمة 2
--	يقوم هذا المعامل بطرح واحد من قيمة المتغير الذي يسبقه ، مثال : X=1;x--; النتيجة : المتغير x يحتوي على القيمة صفر

ب – معاملات منطقية :

المعامل	العملية
&&	يقوم هذا المعامل بعملية "و" المنطقية (AND) ، مثال : X=1;y=2;z=(x<y && y>1); النتيجة : z يحتوي على القيمة True
||	يقوم هذا المعامل بعملية "أو" المنطقية (OR) ، مثال : X=1;y=2;z=(x< y || y>1); النتيجة : z يحتوي على القيمة True
!	يقوم هذا المعامل بعملية "النفي" المنطقية (NOT) ، مثال : X=1;y=2;z=!(x<y); النتيجة : z يحتوي على القيمة True

من المستحسن أن نأخذ مثالاً حقيقياً على هذا النوع من المعاملات :

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  x=1;y=2;
8  z=(x < y && y > 1);
9  document.write (z);
10 z=(x < y || y > 1);
11 document.write ("<br>");
12 document.write (z);
13 z=!(x < y);
14 document.write ("<br>");
15 document.write (z);
16 </script>
17 </body>
18 </html>
```

شكل 53

جـ - معاملات المساواة :

المعامل	العملية
=	هذه هي إشارة المساواة التي استخدمناها أكثر من مرة لإعطاء قيمة للمتغير ، مثال : X=1; النتيجة : المتغير x يحتوي على القيمة 1
+=	يقوم هذا المعامل بإضافة القيمة الأصلية للمتغير على القيمة التي تقع يمين علامة المساواة ،مثال : X=1;x += 2; النتيجة : المتغير x يحتوي على القيمة 3 (2+1)
-=	يقوم هذا المعامل بطرح القيمة التي تقع يمين علامة المساواة من القيمة الأصلية للمتغير ،مثال : X=2;x -= 1; النتيجة : المتغير x يحتوي على القيمة 1 (2-1)
*=	يقوم هذا المعامل بضرب القيمة الأصلية للمتغير في القيمة التي تقع يمين علامة

	المساواة ،مثال : X=2;x *= 2; النتيجة : المتغير x يحتوي على القيمة 4 (2*2)
/=	يقوم هذا المعامل بقسمة القيمة الأصلية للمتغير على القيمة التي تقع يمين علامة المساواة ،مثال : X=4;x /= 2; النتيجة : المتغير x يحتوي على القيمة 2 (4/2)
%=	يقوم هذا المعامل بقسمة القيمة الأصلية للمتغير على القيمة التي تقع يمين علامة المساواة و يخزّن باقي القسمة،مثال : X=3;x %= 2; النتيجة : المتغير x يحتوي على القيمة 1 (3%2)

د-معاملات المقارنة :

المعامل	العملية
==	يقوم هذا المعامل بمقارنة تساوي قيمتين و يخزّن القيمة true أو false بناء على ذلك ، مثال : X=1;y=2;z=(x==y); النتيجة : المتغير z يحتوي على القيمة false.
!=	يقوم هذا المعامل بمقارنة عدم تساوي قيمتين و يخزّن القيمة true أو false بناء على ذلك ، مثال : X=1;y=2;z=(x !=y); النتيجة : المتغير z يحتوي على القيمة true.
<	يقوم هذا المعامل بمقارنة قيمتين و يخزّن القيمة true أو false بناء على ذلك ، مثال : X=1;y=2;z=(x < y); النتيجة : المتغير z يحتوي على القيمة true.
>	يقوم هذا المعامل بمقارنة قيمتين و يخزّن القيمة true أو false بناء على ذلك ، مثال : X=1;y=2;z=(x > y); النتيجة : المتغير z يحتوي على القيمة false.
<=	يقوم هذا المعامل بمقارنة قيمتين و يخزّن القيمة true أو false بناء على ذلك ، مثال :

	X=1;y=2;z=(x <= y); النتيجة : المتغير z يحتوي على القيمة true.
>=	يقوم هذا المعامل بمقارنة قيمتين و يخزّن القيمة true أو false بناء على ذلك ، مثال : X=1;y=2;z=(x >= y); النتيجة : المتغير z يحتوي على القيمة false.

هـ - المعاملات النصية :

المعامل	العملية
+	يقوم هذا المعامل بدمج (إضافة) متغيرين نصيين ، مثال : str1=”he”;str2=”llo”;str3=str1+srt2; النتيجة : المتغير str3 يحتوي على القيمة hello .

هـ - المعامل الثلاثي :

المعامل	العملية
(condition)?value1:value2	يقوم هذا المعامل باختبار شرط ما ، فإذا كان متحققاً فإنه يخزّن القيمة التي تسبق النقطتان الرأسيتان و إن لم يكن متحققاً فإنه يخزّن القيمة التي تلي النقطتان الرأسيتان ، مثال : X=1;y=2; max=(x>y)?x:y; النتيجة : المتغير max يحتوي على القيمة 2.

أوامر التحكم في البرنامج :

يتكون البرنامج أو كود الجافا السكريبت من عدة أسطر برمجية تنفذ الواحد تلو الآخر بالترتيب الذي نكتب فيه هذه السطور . لكن تصادفنا أحياناً بعض البرامج التي نحتاج لاتخاذ بعض القرارات فيها . مثلاً : قد نحتاج لاختبار قيمة المتغير x فإذا كان أكبر من العدد 10 فإننا سنقوم بطباعته ، أما إذا كان أقل من عشرة فإننا لن نقوم بطباعته ! للقيام بمثل هذه القرارات ، فإننا نستخدم أوامر التحكم في البرنامج و هي الجمل الشرطية و جمل التكرار .. سنشرح كلاً من الطريقتين بالأمثلة :

الجمل الشرطية :

تستخدم الجمل الشرطية لتنفيذ الأمر المناسب وفق شرط معين . فنحن نقول في لغتنا العربية : إذا (تحقق الشرط المعين) فافعل أمراً معيناً ، و إذا (لم يتحقق الشرط) فافعل أمراً آخر .

و باللغة الإنجليزية نقول :

```
If (condition)
{
Some statements
}
```

هناك ثلاثة أنواع من أوامر الشرط :

أ-الجملة if :

نستخدم هذه الجملة لتنفيذ أمر أو عدة أوامر إذا تحقق شرط معين ، لنراقب المثال التالي :

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  x=3;y=2;
8  if (x > y){
9  document.write ("المتغير الأول أكبر من المتغير الثاني");
10 }
11 </script>
12 </body>
13 </html>
```

شكل 54

ب-الجملة if .. else :

نستخدم هذه الجملة لتنفيذ أمر أو عدة أوامر إذا تحقق شرط معين ، و لتنفيذ أمر أو عدة أوامر إذا لم يتحقق الشرط .

3.htm

```
1  <html dir="rtl">
2  <head>
3         <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  x=1;y=2;
8  if (x > y){
9  document.write (" المتغير الأول أكبر من المتغير الثاني");
10 }else {
11 document.write ("المتغير الأول ليس أكبر من المتغير الثاني");
12 }
13 </script>
14 </body>
15 </html>
```

شكل 55

جـ-الجملة switch :

نستخدم هذه الجملة لتنفيذ أمر أو عدة أوامر إذا تحققت حالة معينة .

سنأخذ أولاً الصيغة العامة للجملة :

```
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
}
```

شرح المثال : استخدمنا الكلمة switch و هي كلمة محجوزة تستخدم لتنفيذ الأمر الشرطي ثم نضع فيما بين القوسين عبارة (غالباً اسم متغير) ثم نفتح قوساً معقوفاً . ثم نسرد الاحتمالات المختلفة للقيم التي قد يأخذها المتغير . كل احتمال نسبقه بالكلمة المحجوزة case و نختتمه بالأمر Break حتى يتوقف تنفيذ الأوامر عنده – في حالة تحقق الشرط . و نستخدم الكلمة المحجوزة default كآخر خيار حتى يتم تنفيذه في حالة عدم تحقق أي حالة من الحالات المكتوبة . لنأخذ مثالاً حقيقياً لتتضح الصورة بشكل أوضح :

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  x=2;
8  document.write("x = ")
9  switch (x){
10 case 1:
11   document.write(1);
12   break
13 case 2:
14   document.write(2);
15   break
16 case 3:
17   document.write(3);
18   break
19 default:
20   document.write("قيمة غير معروفة")
21 }
22 </script>
23 </body>
24 </html>
```

شكل 56

معلومة إضافية

قد يكون استخدام الأمر break مألوفاً لديك إذا كنت من مبرمجي لغة السي ، لكن قد يكون غريباً عليك إذا كنت من مبرمجي لغة الباسكال التي تقوم بهذه الوظيفة تلقائياً دون الحاجة لهذا الأمر !

لاحظ الحاجة لهذا الأمر سواء كنت من الفريق الأول أو الفريق الثاني و حتى إن لم تكن من أي الفريقين !!يمكنك اكتشاف السبب بشكل أوضح من خلال إزالة الأمر من الجملة ...

جمل التكرار :

لنتخيل أنك تريد كتابة العبارة "بسم الله الرحمن الرحيم" عشر مرات ، فهل ستضطر إلى كتابة عشر أسطر مختلفة لتنفيذ أمر هو في الحقيقية مكرر ؟

تقدم لنا لغة الجافا سكريبت خاصية أوامر التكرار ، التي تقوم بتكرار أمر أو عدة أوامر أكثر من مرة .

لدينا ثلاث أنواع من جمل التكرار أيضاً :

أ-الجملة For :

تستخدم لتنفيذ أمر أو عدة أوامر عدد معين من المرات .
الصيغة العامة للجملة :

```
for (البداية; شرط; زيادة العداد)
{
    بعض الجمل التنفيذية
}
```

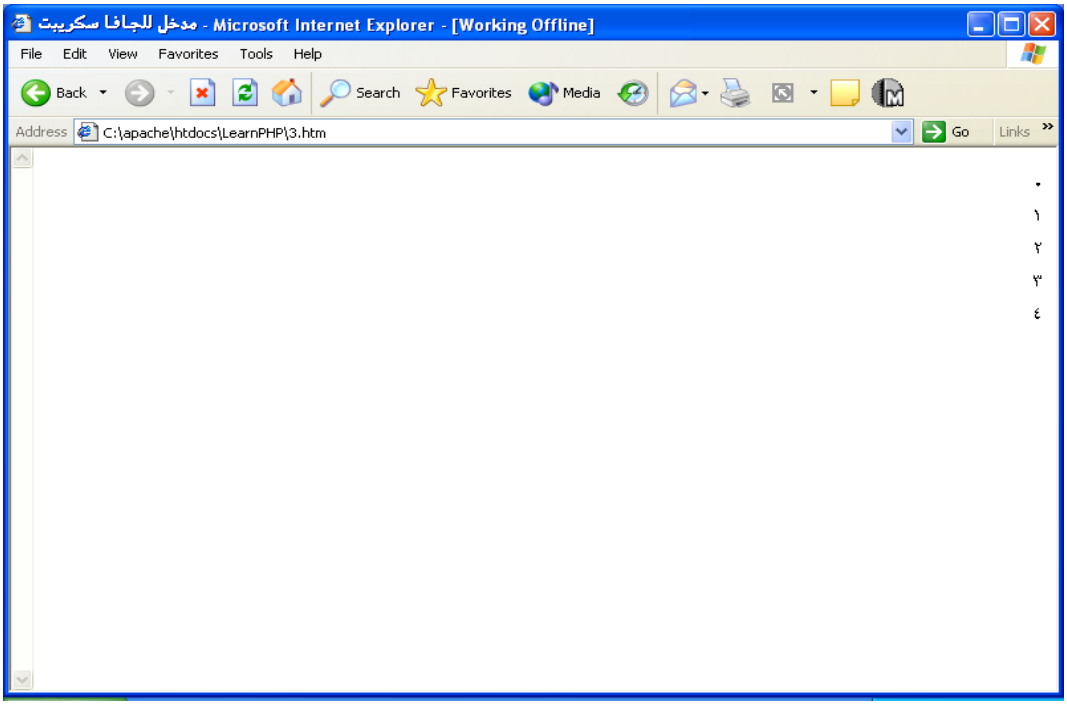
مثال حقيقي :

3.htm

```
1  <html dir="rtl">
2  <head>
3        <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  for (i=0 ; i < 5 ; i++){
8      document.write (i+"<br>");
9  }
10 </script>
11 </body>
12 </html>
```

شكل 57

ستكون النتيجة هكذا (جرب بنفسك):



شكل 58

ب-الجملة while :

تستخدم لتنفيذ أمر أو عدة أوامر طالما أن شرطاً معيناً متحققاً .
الصيغة العامة للجملة :

```
while (شرط)
{
    بعض الجمل التنفيذية
}
```

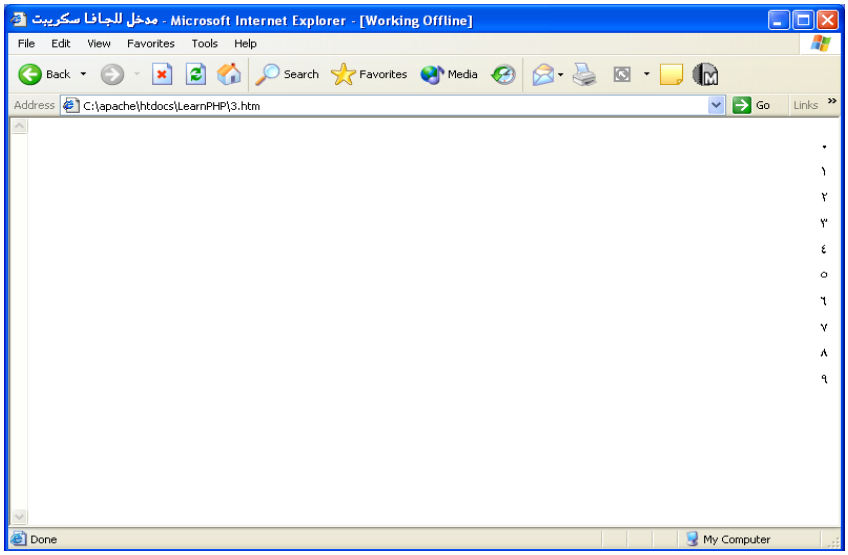
مثال حقيقي :

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  i = 0;
8  while (i < 10){
9      document.write (i+"<br>");
10     i++;
11 }
12 </script>
13 </body>
14 </html>
```

شكل 59

لابد أنك توقعت مسبقاً بالنتيجة ؟جرب بنفسك الآن .. و ستحصل على النتيجة التالية :



شكل 60

جـ-الجملة do .. while :

تستخدم لتنفيذ أمر أو عدة أوامر مرة واحدة ، ثم تختبر تحقق شرطاً معيناً و تكرر تنفيذ الأوامر طالما أن الأمر متحققاً .
الصيغة العامة للجملة :

```
do
{
    بعض الجمل التنفيذية
}
while (شرط)
```

مثال حقيقي :

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  i = 0;
8  do{
9      document.write (i+"<br>");
10     i++;
11 }while (i < 10)
12 </script>
13 </body>
14 </html>
```

شكل 61

و النتيجة موجودة في الشكل رقم 60. تحذير! هناك فرق في طريقة عمل while و .. do while إذا كان هذا الفرق غير واضح لك حتى الآن فأنصحك بالقيام بالمزيد من الأمثلة .

أوامر الإخراج :

اطلعنا على الدالة ()write في أغلب أمثلتنا السابقة .. أعتقد أنها أصبحت واضحة لك بعد تكرارها أكثر من مرة . سنتناول في هذا الجزء تفاصيل إضافية حول كيفية استخدام هذه الدالة لكتابة بعض الأحرف الخاصة .

لابد و أنك لاحظت أننا ندرج النص المراد كتابته بين علامتي تنصيص "النص هنا" .

إذاً رمز التنصيص يشكل بداية و نهاية النص المراد كتابته . ماذا لو أردنا استخدم رمز التنصيص داخل النص المراد لكتابته – كي يظهر على الشاشة ؟ إذا استخدمنا رمز التنصيص مباشرة ، سيعتقد المتصفح أن الجملة المراد كتابتها انتهت بسبب علامة التنصيص ! لذا فإننا نستخدم طريقة خاصة للتعامل مع رمز التنصيص و غيره من الرموز الخاصة . كل ما علينا القيام به هو أن نسبق الرمز الخاص بعلامة الشرطة المائلة (\) . راقب الجدول التالي :

الرمز	ماذا تستخدم
"	\"
&	\&
\	\\

قد نحتاج أحياناً لكتابة بعض النصوص الطويلة التي تأخذ مساحة كبيرة من طول الصفحة مما يمنعنا من قراءة الجملة بسهولة ... أليس كذلك ؟ الحل يكمن في استخدام خاصية مفيدة تقدمها لنا الجافا سكريبت ، حيث يمكننا كتابة النص على سطرين منفصلين ، مع وضع رمز الشرطة المائلة (\) في آخر السطر الأول .

هكذا :

```
document.write("السلام \
عليكم!")
```

لاحظ أنه لا يمكنك كتابة الأمر السابق بالطريقة التالية :

```
document.write \
("السلام عليكم!")
```

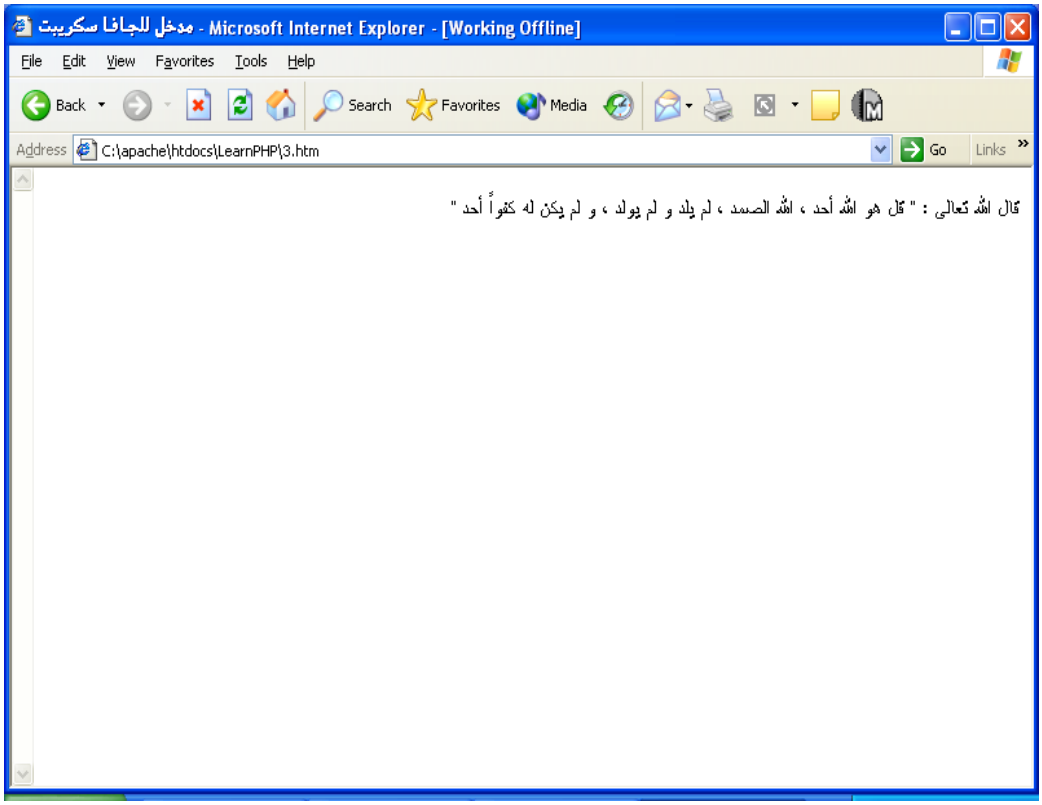
لنأخذ مثالاً يوضح ما تم شرحه حتى الآن :

3.htm

```
1  <html dir="rtl">
2  <head>
3        <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7    document.write ("\" قال الله تعالى : \" قل هو الله
8    \"أحد ، الله الصمد ، لم يلد
9     و لم يولد ، و لم يكن له كفواً أحد\"");
10 </script>
11 </body>
12 </html>
```

شكل 62

يجب أن تفرق بين الشرطة المائلة التي وردت في آخر السطر و الشرطة المائلة التي وردت قبل علامة التنصيص في منتصف الجملة النصية . نتيجة المثال ستكون هكذا :



شكل 63

الدوال :

أبسط تعريف للدالة قد يكون بأنها مجموعة من الأسطر البرمجية التي نعطيها اسماً معيناً يمكننا من استخدامها أكثر من مرة بمجرد كتابة سطر واحد يحتوي على اسمها .

هناك نوعين من الدوال :

أ-دوال جاهزة (built-in): و المقصود بها تلك الدوال التي تقدمها لنا الجافا سكريبت . مثال : الدالة (write()) التي تعرفنا عليها خلال هذا الفصل . و كذلك الدالة (alert()) التي تقوم بإظهار مربع تحذير .

ب-دوال خاصة بالمستخدم :حيث يقوم المستخدم بتعريف دالة خاصة به ؛ يمكنه استخدامها أكثر من مرة بعد ذلك .

سنتناول هنا كيفية إنشاء دوال بسيطة خاصة بك أنت !

هناك صيغة عامة للدوال .. قبل أن نأخذ هذه الصيغة ، أحب أن أذكرك بأن عليك تعريف الدالة قبل استخدامها ! لذا يجب أن تضع التعريف – حسب الصيغة العامة التي سنتناولها بعد قليل – في رأس الصفحة ، حتى تضمن تحميلها قبل مناداتها من خلال صفحتك . الآن ، لقد كنا نكتب نصاً بداخل قوسي الدالة (write() أليس كذلك ؟ هذا يعني أننا كنا نرسل هذا النص للدالة لكي تقوم بكتابته على الشاشة ، هذا النص يمكن اعتباره متغيراً نقوم بإرساله للدالة . إذن أنت تعرف الآن أن الدوال قد تأخذ متغيراً ، و قد لا تحتاج لمتغير أيضاً ! لذلك سنأخذ أولاً الصيغة العامة للدوال التي تحتاج لمتغيرات :

```
function اسم الدالة(المتغير الأول,المتغير الثاني,..)
{
بعض الجمل التنفيذية
}
```

و هذه هي الصيغة العامة للدوال التي لا تحتاج لمتغيرات (لاحظنا أننا مازلنا بحاجة للأقواس):

```
function اسم الدالة()
{
بعض الجمل التنفيذية
}
```

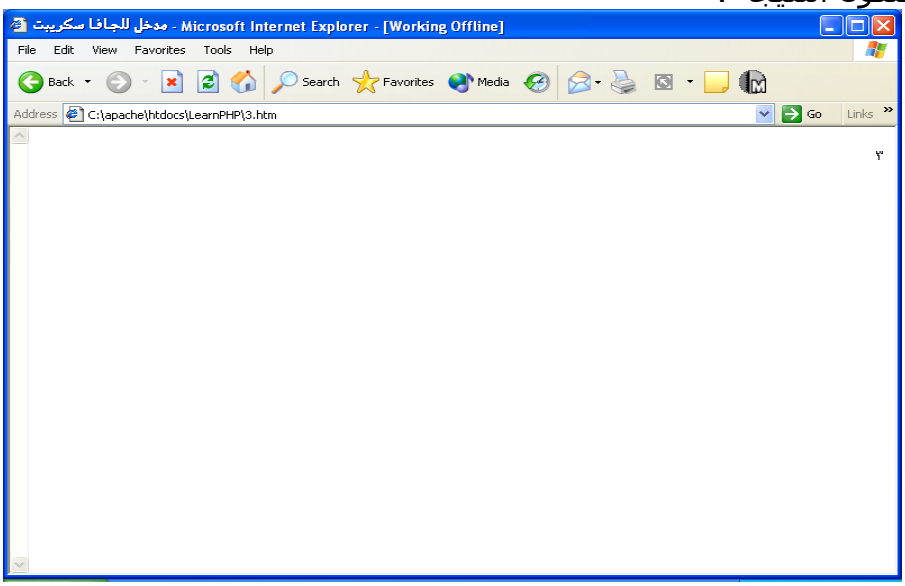
و هذا مثال حقيقي (دالة تقوم باستقبال متغيرين ، تطبع حاصل جمعهما):

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  function write_sum(x,y){
8      document.write (x+y);
9  }
10 x=1;y=2;
11 write_sum(x,y);
12 </script>
13 </body>
14 </html>
```

شكل 64

هكذا ستكون النتيجة :



شكل 65

إذن ، هلا لاحظت كيف ننشأ الدالة ؟ ثم كيف يمكننا أن نستدعيها فيما بعد ؟

يمكنك إضافة المزيد إلى دوالك ! نعم .. تستطيع أن تجعل الدالة تعيد قيمة معينة عند استدعائها .. قد تجعلها تعطينا ناتج الجمع مثلاً .. بدلاً من طباعته !

لنأخذ هذا المثال الذي يوضح ذلك :

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  function return_sum(x,y){
8      z = x+y;
9      return z;
10 }
11 x=1;y=2;
12 document.write(return_sum(x,y));
13 </script>
14 </body>
15 </html>
```

شكل 66

كما تستطيع كتابة المثال السابق بالطريقة التالية :

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script language="javascript">
7  function return_sum(x,y){
8      z = x+y;
9      return z;
10 }
11 x=1;y=2;z=return_sum(x,y);
12 document.write(z);
13 </script>
14 </body>
15 </html>
```

شكل 67

في كلا المثالين : استخدمنا الأمر return ثم اسم المتغير الذي يحمل الناتج المطلوب .

التأكد من بيانات النماذج :

أصدقك القول إذا قلت بأني شرحت الفصل كاملاً من أجل هذا الجزء تقريباً ! سنحتاج للكثير من النماذج في مشاريع البي اتش بي .. يمكننا أن نستخدم البي اتش بي نفسها للتأكد من صحة البيانات المستخدمة في النموذج ، لكن هذا يعني بأن على المستخدم أن يرسل بياناته إلى الجهاز الخادم أولاً ، ثم ينتظر معالجة البيانات هناك لمعرفة ما إذا كانت صالحة أم لا ثم يأتيه الرد ! بينما ، باستخدام الجافا سكريبت – التي تعتبر من اللغات التي يتم تنفيذها على جهاز العميل مباشرة – لن نخسر كل هذا الوقت ! لذلك فالجافا سكريبت هي الخيار المناسب للتأكد من بيانات النماذج . أنت متقنع بذلك الآن ، و تريد أن تعرف الطريقة .. سنعرف ذلك الآن :

العملية تتكون من خطوتين :

1-تعريف الدالة التي تقوم بالتأكد من صحة البيانات المدخلة في رأس الصفحة (head).

2-استدعاء الدالة في وسم النموذج (form) من خلال المتغير التالي : onsubmit="return validate()" حيث أن validate هي اسم الدالة التي قمنا بتعريفها في الخطوة الأولى .

إن أفضل طريقة لتعلم ذلك هي عن طريق بعض الأمثلة الحقيقية ..

مثال : التأكد من ادخال بريد إلكتروني صحيح

3.htm

```
<html dir="rtl">
<head>
        <title>مدخل للجافا سكريبت</title>
<script type="text/javascript">
function validate(){
x=document.Form1
index=x.email.value.indexOf("@")
if (index == -1){
    alert("ادخل بريداً صحيحاً من فضلك !")
    return false
    }
}
</script>
</head>
<body>
<form name="Form1" action="script1.php"
onsubmit="return validate()">
البريد الإلكتروني :
<input type="text" name="email">
<input type="submit" value="تأكد من البريد">
</form>
</body>
</html>
```

شكل 68

شرح المثال : يقوم هذا المثال بالبحث عن رمز "@" في البريد الإلكتروني المدخل في مربع النص . فإذا ما وجد هذا الرمز فإنه يعتبر البريد صحيحاً و يسمح بإرسال البيانات إلى الصفحة (script1.php)، و إذا لم يجده فإنه يعتبر البريد المدخل غير صحيحاً و يظهر رسالة تحذير بالإضافة إلى أنه سيعيد القيمة false كنتيجة للدالة .

مثال : إدخال قيمة عددية لا تزيد عن رقم معين

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  <script type="text/javascript">
5  function validate(){
6  x=document.Form1
7  val=x.num.value
8  if (val>100) {
9   alert("ادخل رقماً لا يزيد عن المئة !")
10  return false
11  }
12 }
13
14 </script>
15 </head>
16 <body>
17 <form name="Form1" action="script1.php"
18 onsubmit="return validate()">
19 عمرك :
20 <input type="text" name="num">
21 <input type="submit" value="ارسل العمر">
22 </form>
23 </body>
24 </html>
```

شكل 69

شرح المثال : يقوم المثال بالتأكد من قيمة الرقم المدخل في مربع النص . إذا كانت القيمة أكبر من المائة فإنه يقوم بإظهار رسالة تحذير و يعيد القيمة false كنتيجة للدالة ، إذا كانت القيمة أقل من أو تساوي المائة ، فإنه يقبل القيمة و يرسلها للملف script1.php .

التاريخ و الوقت :

يمكنك أيضاً الاعتماد على الجافا سكريبت في إظهار تاريخ اليوم على الصفحة الأولى لموقعك ! لذا سنتطرق بشكل سريع لكيفية الاستفادة من الجافا سكريبت في هذه المسألة ..

تحذير !

أنت تعلم بأن الجافا سكريبت هي إحدى لغات جهات العميل (client-side) و لهذا السبب نفضل استخدامها في بعض الأمور البسيطة حتى لا نهدر وقتاً يضر بجودة مشاريعنا !
لكن يجب أن ننتبه لمسألة هامة تتعلق بالوقت و التاريخ .. هذه المسألة تعود لاحتمال إعداد الوقت بشكل خاطئ على جهاز العميل و بالتالي فإن الجافا سكريبت يستخدم

التاريخ الخاطئ المعد على الجهاز ! و بالتالي قد تجد أن صفحتك تقول بأن اليوم هو الأول من السنة العاشرة للهجرة إذا كان جهاز العميل يقول ذلك !

يمكننا الحصول على تاريخ اليوم و التوقيت الحالي عن طريق الكائن Date . و للقيام بذلك يجب أن ننشأ نسخة من هذا الكائن عن طريق استخدام الأمر new ثم نستخدم بعض الدوال التي يحتويها هذا الكائن لكي نحصل على تاريخ اليوم (1-31) بشكل منفصل ، رقم الشهر بشكل منفصل ، السنة بشكل منفصل أيضاً ، و كذلك الوقت بالساعات و الدقائق و الثواني . سيبدو الأمر سهلاً لمن اعتاد على لغات البرمجة الشيئية من أمثال السي++ (OOP) و قد يبدو مربكاً قليلاً لمن لم يعتد عليها ، لكن بالتأكيد سيبدو لك الأمر سهلاً بعد قراءة المثال الآتي بتمعن :

3.htm

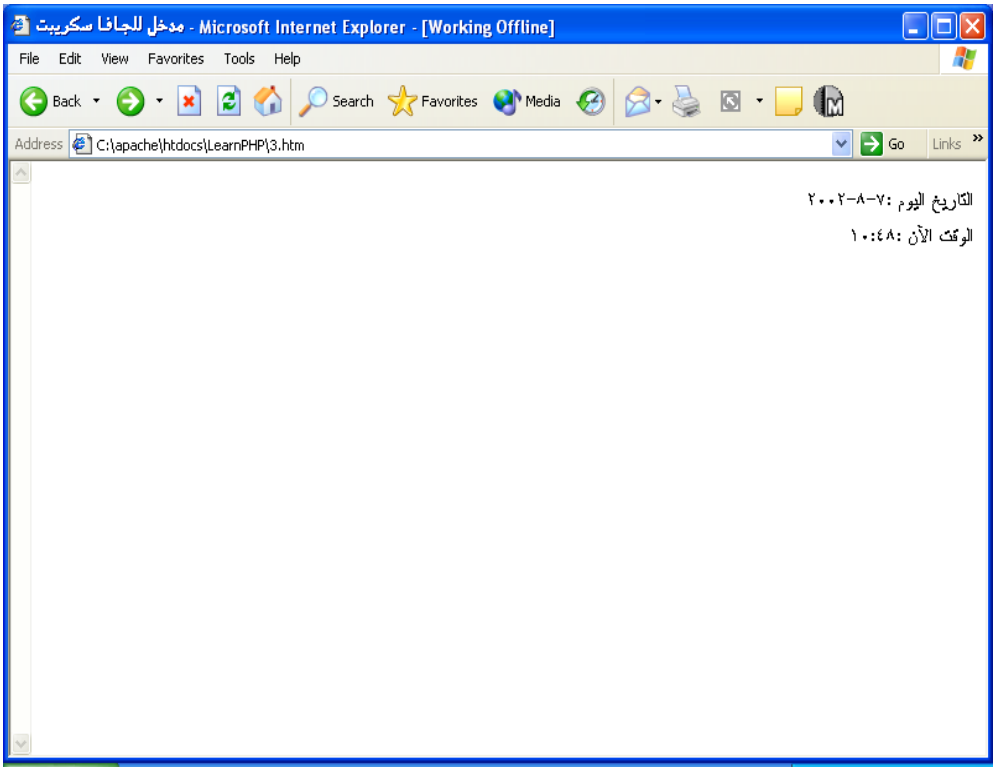
```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script type="text/javascript">
7  var date1 = new Date();
8  d = date1.getDate();
9  m = date1.getMonth() + 1;
10 y = date1.getFullYear();
11 document.write ("التاريخ اليوم :");
12 document.write (d+"-"+m+"-"+y);
13 h = date1.getHours();
14 mi = date1.getMinutes();
15 document.write ("<br>الوقت الآن :");
16 document.write (h+":"+mi);
17 </script>
18 </body>
19 </html>
```

شكل 70

شرح المثال :

في البداية قمنا بإنشاء نسخة من الكائن Date بواسطة الأمر new و أعطينا هذه النسخة الاسم date1 . الآن هذا المتغير ليس متغير عادي ! إنه عبارة عن نسخة من كائن ، هذا الكائن يحتوي على متغيرات و يحتوي على دوال .
سنستفيد من الدوال التي يحتويها هذا الكائن : للحصول على تاريخ اليوم (1-31) نستخدم الدالة GetDate() ، للحصول على رقم الشهر الحالي (0-11) نستخدم الدالة GetMonth() ، للحصول على السنة الحالية نستخدم الدالة GetFullYear() ، للحصول على الساعة الحالية نستخدم الدالة GetHours() ، للحصول على الدقائق الحالية نستخدم الدالة GetMinutes() . مع ملاحظة أننا قمنا بإضافة واحد صحيح إلى رقم الشهر حتى يعطينا الرقم الصحيح للشهر(0=يناير ، 1=فبراير) .

و هذه هي نتيجة المثال :



شكل 71

اقرأ الجدول التالي لمزيد من الدوال المفيدة الخاصة بالتاريخ و الوقت :

الدالة	الوظيفة
getDate()	للحصول على التاريخ (1-31)
getDay()	للحصول على اليوم (0-6 ، 0=الأحد ، 1 =الأثنين ... الخ)
getMonth()	للحصول على الشهر (0- 11 ، 0=يناير ، 1 = فبراير)
getYear()	للحصول على السنة (0-99)
getFullYear()	للحصول على السنة الكاملة (0-9999)
getHours()	للحصول على الساعات (0-23)
getMinutes()	للحصول على الدقائق (0-59)
getSeconds()	للحصول على الثواني (0-59)
getMilliseconds()	للحصول على أجزاء الثانية (0-999)
getTime()	للحصول على الوقت بأجزاء الثواني (منذ تاريخ 1-1-1970)
getTimezoneOffset()	للحصول على الفارق الزمني بين توقيت الجهاز و توقيت جرينتش ()
toString()	لتحويل التاريخ إلى سلسلة نصية .

الملاحظات :

لا يمكننا أن نختتم هذا الفصل قبل أن نشرح كيفية إدراج ملاحظة (comment) داخل أكواد الجافا سكريبت . الطريقة مشابهه لطريقة الملاحظات التي تعلمناها في الفصل السابق – سي اس اس .

المثال التالي يوضح طريقتين مختلفتين لإدراج الملاحظات :

3.htm

```
1  <html dir="rtl">
2  <head>
3          <title>مدخل للجافا سكريبت</title>
4  </head>
5  <body>
6  <script type="text/javascript">
7  //ملاحظة من سطر واحد
8
9  /*ملاحظة مكونة
10 من
11 عدة أسطر
12 */
13 </script>
14 </body>
15 </html>
```

الخلاصة :

تعرفنا في هذا الفصل على لغة الجافا سكريبت . نحن نعرف الآن أن الجافا سكريبت هي لغة برمجة من الوزن الخفيف ، نعرف كيفية تعريف متغيرات جديدة ، كيفية الاستفادة من المعاملات المختلفة ، كيفية استخدام أوامر الشرط و أوامر التكرار . و الأهم من هذا كله : أصبحنا نعرف متى يجب علينا استخدام الجافا سكريبت بدلاً من بي اتش بي .

لم يكن الفصل خفيفاً مثل الفصل السابق ؟ ربما ، لكنك استفدت الكثير و اكتسبت قاعدة ممتازة للبدء في تعلم لغة البي اتش بي .

المراجع :

هذا الكتاب غير مخصص للغة الجافا سكريبت لذلك لم نتطرق لكل ما يخصها ، إذا كنت مهتماً بتعلم المزيد عن الجافا سكريبت ، يمكنك الرجوع لبعض هذه الكتب :

- **Professional JavaScript with DHTML, ASP, CGI, FESI, Netscape Enterprise Server, Windows Script Host, LiveConnect and Java by Sing Li, Andrea Chirelli, Stuart Updegrave, Cliff Wootton, Nigel McFarlane, Mark Wilcox, Paul Wilton, James De Carli (Wrox Press)**
- **Beginning JavaScript by Paul Wilton (Wrox Press)**
- **JavaScript Bible by Danny Goodman (Hungry Minds, Inc)**

الباب الثاني : مدخل للغة PHP

مقدمة الباب الثاني

بعد أن انتهينا من مداخل للغات أخرى ، في الفصل السابق ، سندخل الآن في لغة بي اتش بي . هذا الباب عبارة عن مدخل للغة ، سنتناول في الفصل الرابع بعض المعلومات حول كيفية بداية اللغة مع مقارنة بينها و بين اللغات المنافسة . سنتناول كذلك الفرق بين اللغات جهة المزود و اللغات جهة العميل و بعض المعلومات التي توضح كيفية تنفيذ صفحات البي اتش بي التي سنكتبها فيما بعد . الفصل الخامس يقدم طريقة واضحة لتنصيب البي اتش بي على جهازك . الفصل السادس يقدم طريقة واضحة لتنصيب الماس اس كيو ال (MySQL) .

الفصل الرابع: كيف و متى و لماذا PHP ؟

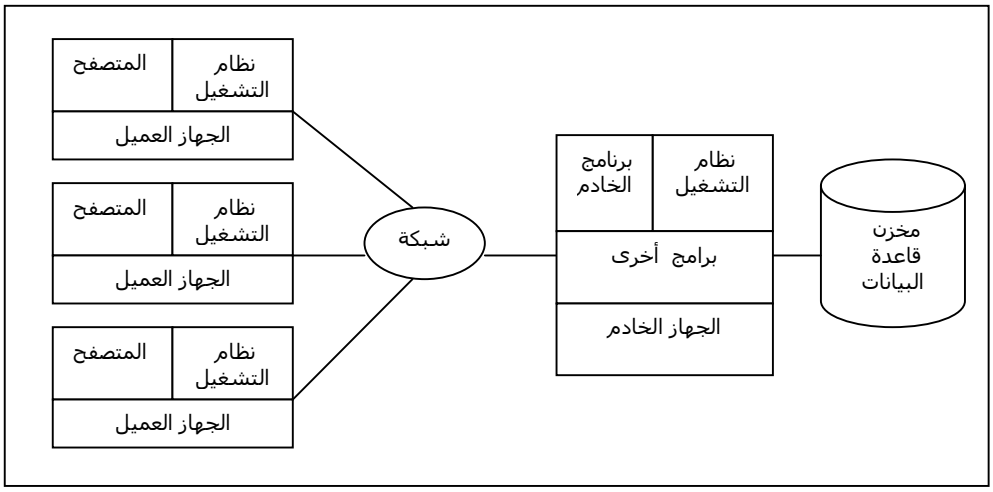
عنوان هذا الفصل يحتوي على ثلاثة أسئلة ، يفترض أننا بعد أن أنهينا الثلاثة فصول السابقة تكوّنت لدينا مثل هذه الأسئلة (كيف ؟ متى ؟ لماذا ؟) و العديد من الأسئلة الأخرى التي تبحث عن إجاباتها .إذا كانت هذه الأسئلة قد تبادرت إلى ذهنك ، فاسمح لي أن أحييك يا سيدي\سيدتي ! إذ لا ينبغي عليك أن تبدأ بالجزء البرمجي من اللغة قبل أن تعرف تاريخها ، مدى انتشارها ، تقارنها بغيرها من اللغات المنافسة- و ما أكثرها - ، يجب أن تحدد الآن إجابة للسؤال التالي : هل أنت مقتنع بهذه اللغة بناء على معلومات منطقية ؟ هل تستطيع الإجابة على سؤال لماذا تتعلم PHP ؟ سيقدم لك هذا الفصل أجوبة لكل هذه الاستفسارات بإذن الله .. سنحاول أن نستعرض واقع الشبكة العنكبوتية بشكل عام أولاً .

عندما نقول إنترنت ، فإن مصطلح "شبكة الشبكات" يتبادر إلى أذهاننا بسرعة . أيضاً : نستطيع أن نقول أن هذه الشبكة مكونة من نوعين من الأجهزة : أجهزة خادمة و أجهزة عميلة (server/client) . تتمثل مهمة الأجهزة الخادمة في حفظ المعلومات(المواقع) على أجهزتها التي تكون عادة متصلة بالشبكة على مدار الأربع و عشرين ساعة ، و بالطبع فإن إمكانيات الجهاز الخادم و جودة طريقة الاتصال بالشبكة تكون عالية حتى تتحمل الطلبات الكثيرة على المعلومات المحفوظة على هذا الجهاز . هذه الأجهزة تكون تابعة لشركات الاستضافة المختلفة عبر أنحاء العالم . و بالإضافة إلى المكونات الصلبة العالية التي تتمتع بها هذه الأجهزة ، فإنها تحتوي أيضاً على نوعية خاصة من البرامج تجعل هذا الجهاز خادم حقيقي ، يشمل ذلك : نظام تشغيل يعتمد عليه من ناحية الأداء و الأمان ، برنامج تحويل النظام إلى خادم (أشهر برنامج بهذا الخصوص هو برنامج الأباتشي Apache المجاني و المفتوح المصدر) ، برنامج لترجمة اللغات البرمجية التي يدعمها هذا الجهاز (من أمثال : PHP , Perl , ASP , ColdFusion) - برنامج لكل لغة على حده .
أما الأجهزة العميلة ، فهي ليست سوى أجهزتنا التي نتصل من خلالها بالشبكة . هذه الأجهزة لا يتطلب منها أن تكون بمواصفات عالية ، كما أنه لن تحدث كارثة لو تم قطع الاتصال بين هذا الجهاز و بين الشبكة . البرامج المطلوبة على هذا الجهاز تشمل : نظام التشغيل (أياً كان) ، متصفح الإنترنت .
نتطرق الآن لطريقة اتصال هذين النوعين مع بعضهما : يقوم الجهاز العميل (من خلال المتصفح) بطلب عنوان معين ، هذا العنوان يكون عادة عبارة عن حروف تشكل اسم موقع مثل : www.c4arab.com و الذي يشير بدوره إلى رقم تعريفي خاص (IP) و هذا الرقم يشير بدوره إلى جهاز خادم معين داخل شبكة معينة . و عند الاتصال ببرنامج محدد (MySQL مثلاً) فإن هذا الاتصال يتم عبر منفذ (Port) معين .
و في الواقع فإنه يوجد نوع ثالث من الأجهزة هو مخزن قواعد البيانات الذي يكون متصلاً بالجهاز الخادم فقط بحيث يتحقق نظام الشبكات ذات الثلاث طبقات (3-Tier Architecture) .

معلومة إضافية

سنتعلم في الفصول القادمة كيفية تنصيب برامج الجهاز الخادم على أجهزتنا العميلة لتصبح بدورها أجهزة خادمة أيضاً (بشكل شخصي لك أنت فقط و لأغراض التجربة و التعلم) .

لو نظرنا للشبكة بشكل تاريخي ، فإننا سنلاحظ أن نظام الأجهزة العميلة \ الأجهزة الخادمة لم يتغير أبداً خلال السنين الماضية ، الشيء الوحيد الذي اختلف في هذا المجال هو نوعية البرامج التي نحتاج لتنصيبها على الجهاز الخادم ، حيث أن ظهور لغة جديدة يستلزم تنصيب البرامج الداعمة لها (ينطبق ذلك في بعض الأحيان على الأجهزة العميلة أيضاً) .
الشكل التالي يوضح لنا ما ذكرناه بطريقة مصورة :



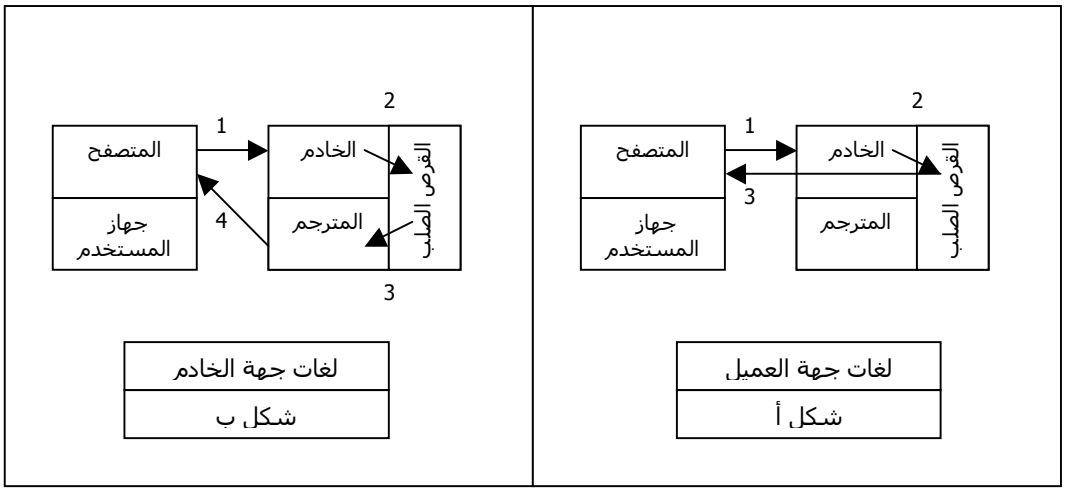
شكل 72

ماذا عن نوعية المعلومات التي يحفظها الجهاز الخادم (المواقع) ؟ هل تغيرت هي الأخرى ؟
في الواقع – و كما ذكرنا في جزء سابق من هذا الكتاب – فإن هذا المحتوى في البداية كان موجها للعلماء بالدرجة الأولى و كان عبارة عن صفحات نصية فقط ، الاهتمام موجه على المعلومات أكثر من الشكل . كانت اللغة المستخدمة حين ذاك هي لغة الهتمل (HTML) . هذه الصفحات كانت تحتوي على روابط للتنقل من معلومة لأخرى . مع مرور الوقت ، أصبحنا بحاجة إلى المزيد ! ظهرت تقنيات مختلفة لتحسين الوضع .. سنناقشها فيما يلي :

يمكن تقسيم اللغات إلى نوعين رئيسيين :
أ-اللغات جهة العميل :
يطلب المستخدم صفحة معينة ، يتم جلب هذه الصفحة (المعلومة) من القرص الصلب للجهاز الخادم ثم يقوم العميل (المتصفح) بترجمة هذه الصفحة إلى معلومات تظهر لدى جهاز المستخدم . لاحظ أننا احجتنا لإرسال طلب واحد فقط . انظر الشكل (73-أ) .
ميزة هذا النوع من اللغات هي السرعة في التنفيذ و عيبه أن المصمم ليس لديه تحكم كامل في نوعية المتصفح المستخدم لترجمة ما كتبه من أكواد !

ب-اللغات جهة المزود (الخادم) :

يطلب المستخدم صفحة معينة ، يتم جلب هذه الصفحة (المعلومة) من القرص الصلب للجهاز الخادم ثم يقوم الجهاز الخادم بترجمة المعلومات المطلوبة و تحويلها إلى لغة الهتمل ثم يرسلها للعميل . انظر الشكل (73-ب). ميزة هذا النوع من اللغات هو أن لدينا تحكم أكبر في البرنامج المستخدم للترجمة ، لكن يوجد عيوب تتمثل في زيادة الوقت المطلوب لوصول الاستجابة (قد تكون أجزاء من الثانية ، لكنها تهمنا !).



شكل 73

HTML

تشير الأحرف الأولى من اسم اللغة إلى الكلمات الإنجليزية : **H**yper **T**ext **M**arkup **L**anguage ، التي تشير إلى (لغة ترميز النص التشعبي) . هذه اللغة هي اللغة الأولى على الشبكة . تدعم جميع المتصفحات الموجودة الوسوم الأساسية لهذه اللغة مع وجود بعض الوسوم القليلة التي قد تكون مدعومة من متصفح دون آخر . يحدث هذا عادة بسبب إضافة المزيد من الوسوم بشكل سريع أو بسبب أن الوسم منتج من الشركة المصنعة للمتصفح و بالتالي تكون داعمة له في بقية منتجاتها أيضاً بينما لا تدعمه بقية الشركات المنافسة . هنا لابد أن نسأل أنفسنا : ما الذي ينقص هذه اللغة ؟ و لماذا نحتاج لحلول أخرى قد لا تكون مدعومة من جميع المتصفحات أو قد تحتاج بعض الوقت لإتقانها ؟

و الجواب الواضح هو محدودية هذه اللغة ! لغة الهتمل هي لغة تنتج محتوى ثابت فالجملة التي يكتبها مصمم الموقع ستظهر كما كتبها لدى جميع المستخدمين و في كل وقت ! لنفرض أنه كتب "الوقت الآن : 5:40" و الذي يشير إلى وقت كتابته لهذه الجملة ، ماذا سيحدث عندما يطّلع المستخدم على صفحته بعد ساعة ؟ ستظل الساعة تشير إلى الخامسة و أربعون دقيقة !بينما كان بإمكاننا أن نجعل هذا المحتوى متحرك يتغير بتغير الزمن أو بتغير المستخدم .

العيب الآخر للغة الهتمل أنها لا تفصل المحتوى عن التنسيق – على فرض عدم استخدام تقنية السي اس اس – بينما يوجد حلول أخرى تقوم بذلك .

الحل قد لا يكون بالإستغناء عن هذه اللغة تماماً لكن بدمجها مع لغات أخرى – بحسب الحاجة – مثل لغات البرمجة الخفيفة (الجافا سكريبت و شبيهاتها) أو لغات البرمجة الثقيلة (PHP,ASP...الخ) .

بناءً على المقاييس الصادرة عن جمعية w3c فإن لغة الهتمل سيتم استبدالها قريباً بالجيل الثاني من هذه اللغة (XHTML= **EX**tensible **H**yper **T**ext **M**arkup **L**anguage) . هذه اللغة لا تختلف كثيراً عن لغتنا إلا بإضافة بعض الشروط على طريقة كتابة الوسوم . هذه اللغة تعتبر لغة من اللغات التي تنفذ جهة العميل ، حيث أن المتصفح هو المسؤول عن ترجمة وسوم الهتمل (يتجاهل أي وسم لا يفهمه عادة) .

للاطلاع على مزيد من المعلومات حول لغة الهتمل يرجى مراجعة الفصل الأول من هذا الكتاب .

DHTML

تشير الأحرف الأولى من اسم اللغة إلى الكلمات الإنجليزية : **D**ynamic **H**yper **T**ext **M**arkup **L**anguage ، التي تشير إلى (لغة ترميز النص التشعبي الآلية) . ظهرت هذه اللغة كحل لثبات المحتوى في لغة الهتمل . حيث أصبح بإمكان المصمم أن يضيف بعض الحركة لصفحته . الأمثلة على مثل هذه الإضافات كثيرة جداً و محدودة بخيال المصمم فقط . يمكنك أن تجعل إحدى مكونات صفحتك تستدير باتجاه معين ، تغير لون العناصر ، تضيف تأثير الأمواج أو الخلفية الشبه شفافة ، باختصار : تستطيع أن تجعل كل عنصر من عناصر صفحتك يتفاعل مع المستخدم . تعتبر هذه اللغة علم واسع حقاً يمكن أن تكتب عنه مئات الصفحات لشرحه ! هذه اللغة تعتبر من لغات جهة العميل أيضاً (راجع الشكل 73- أ) .

ActiveX Controls

مكونات الأكتيف اكس هي عبارة عن بريمجات صغيرة يمكن صنعها بواسطة لغتي السي ++ أو الفيجوال بيسك . قامت شركة مايكروسوفت بإنتاج هذه المكونات لإضافة بعض الوظائف التي قد يحتاجها مطورو مواقع الشبكة العنكبوتية . بعض هذه الوظائف هي : الرسوم البيانية ، المؤقتات ، الاتصال بقواعد البيانات . يمكنك إضافة هذه المكونات إلى صفحات الهتمل عن طريق الوسم <object> . هذه اللغة لا يمكن اعتبارها من لغات جهة العميل أو لغات جهة الخادم ، حيث يعتمد ذلك على ما يقوم به هذا الكائن . الواقع أن هذه المكونات قد تكون مفيدة حقاً ، لكن لا ينصح باستخدامها على أي حال ! السبب هو أنها غير مدعومة سوى من متصفح المايكروسوفت اكسبلورر و بالتالي فإن صفحتك لن تعمل بالشكل المطلوب على متصفحات أخرى شهيرة مثل متصفح النت سكيب إلا بواسطة مكونات إضافية تضاف للمتصفح (Plug-ins) .

CGI

تشير الأحرف الأولى من اسم اللغة إلى الكلمات الإنجليزية : **C**ommon **G**ateway **I**nterface، التي تشير إلى (واجهة البوابة العامة) . تعتبر هذه اللغة من أقدم و أشهر اللغات المستخدمة لتطبيقات الإنترنت جهة الخادم . ستجد أن هذه اللغة مدعومة في كل شركات الاستضافة الحالية ! يمكنك كتابة برامج السي جي أي باستخدام أي لغة تقريباً ، إلا أن أشهر اللغات المستخدمة لذلك هي لغة البيرل . قد تكون تتسائل الآن عن سبب التسمية ؟ في الواقع هذا يعود لطريقة تنفيذ البرنامج . حيث أن السي جي أي يكون بمثابة الواجهة أو البوابة التي تربط الخادم بالبرنامج . يتم هذا الربط عبر خطوات : أولاً ، يتم إنشاء عملية (Process) يعمل البرنامج من خلالها . ثانياً ، يتم تحميل البرنامج المطلوب . ثالثاً ، يتم تحميل أي برامج أخرى يحتاجها البرنامج الأصلي . أخيراً ، بعد أن ينتهي البرنامج يقوم الخادم بإرسال النتيجة إلى المستخدم . مميزات هذه اللغة تتمثل في الدعم الواسع لها على جميع خوادم الشبكة العنكبوتية ، هذا يعني أنك لن تضطر لاختيار شركة استضافة معينة أو دفع المزيد من المبالغ للحصول على دعم خاص لبرامجك . عيوب هذه اللغة تتمثل في انخفاض مستوى الأداء عندما تزيد الطلبات على الخادم حيث أن ذلك يتطلب إنشاء عملية منفصلة لكل طلب . يوجد تقنية أخرى قد تحل هذه المشكلة هي FastCGI .

ASP

تشير الأحرف الأولى من اسم اللغة إلى الكلمات الإنجليزية : **Active Server Pages** ، التي تشير إلى (صفحات الخادم النشطة) . هذه اللغة تعتبر منافس قوي للغة البي اتش بي . كما أنها تتخذ نفس أسلوب البي اتش بي من حيث أن أكواد اللغة تكون مدمجة من أكواد لغة الهتمل في ملف واحد . تتم ترجمة الأكواد في الخادم ثم ترسل النتيجة على أكواد هتمل فقط . صفحات الخادم النشطة تأخذ الامتداد asp. عادة .
كما أنها تدعم لغتي برمجة من الوزن الخفيف : JScript و VBScript .

مميزات اللغة قد تحسب لمبرمجي الفيجوال بيسك و الفيجوال بيسك سكريبت (VBScript) من حيث أن تعلم هذه اللغة سيعتبر أمراً سهلاً بالنسبة لهم . عيوب هذه اللغة تتمثل في أنها لغة غير مجانية ، غير مفتوحة المصدر – مما يعني أن الأخطاء لا يتم إصلاحها في الوقت المناسب ، بالإضافة لمحدودية تطوير اللغة – و العيب الأكبر هو أنك لا تستطيع استخدامها اللغة إلا على خادم يحتوي على نظام التشغيل ويندوز مع برنامج خادم من إنتاج شركة مايكروسوفت (IIS, PWS) .
إذا كنت تخطط لتطوير مشاريع تعمل على شبكات منتجة من شركة مايكروسوفت فقط (شبكة داخلية مثلاً) فيمكنك الاعتماد على لغة صفحات الخادم النشطة أما إذا كانت أعمالك موجه للقاعدة العريضة من المستخدمين اللذين يستخدمون منتجات من شركات أخرى فإن هذا الحل لا يعتبر أفضل الحلول بالنسبة لك .

معلومة إضافية

تدعم بعض أنظمة اللينكس لغة الـ ASP بشكل غير كامل حتى مع عدم استخدام أحد برامج الخادم المنتجة من شركة مايكروسوفت . لكن هذا الدعم – كما ذكرنا – غير كامل .

JSP

تشير الأحرف الأولى من اسم اللغة إلى الكلمات الإنجليزية : **Java Server Pages**، التي تشير إلى (صفحات الجافا – جهة الخادم) . و هي إحدى لغات جهة الخادم كما يشير الاسم بالطبع . هذه اللغة هي من إنتاج فريق المطورين التابع لشركة سن (SUN) . تعمل هذه اللغة بطريقة مشابهة للغتي صفحات الخادم النشطة (ASP) و لغة البي اتش بي (PHP) من حيث أن صفحة الجافا تحتوي على أكواد الهتمل بالإضافة إلى أكواد الجافا . تعتمد هذه اللغة على فصل المحتوى عن طريقة العرض . تنتهي صفحات الجافا – جهة الخادم بالامتداد jsp. في العادة . و عند طلب إحدى صفحات الجافا من الخادم ، يقوم الخادم بترجمة الكود و يرسل النتيجة ، لكن يظل الكود بعد الترجمة موجود في الذاكرة مما يسمح بإرسال النتيجة في فترة قياسية عند تكرار الطلب . هذه الخاصية تعطي للغة قوة عالية يعتمد عليها في المشاريع الضخمة .
تستمد هذه اللغة قوتها من قوة اللغة الأساسية (الجافا) حيث يمكنك الاستفادة من الميزات الكثيرة التي تقدمها لغة الجافا في مشاريعك على الشبكة العنكبوتية .
قد تستغرب من عدم انتشار هذه اللغة على مالها من مميزات ، و السبب يكمن في أنها قد تكون صعبة بعض الشيء على من لم يعتد على البرمجة بلغة الجافا ، الشيء الذي يمكننا اعتباره ميزة إذا كان لك خبرة سابقة في لغة الجافا ! أليس كذلك ؟

ColdFusion

هذه إحدى اللغات المنافسة للغة البي اتش بي أيضاً . صممت هذه اللغة من أجل هدف واضح هو تقديم حلول عملية لتطبيقات الشبكة العنكبوتية . تم تطوير اللغة من

قبل شركة ألير (Allaire) . اللغة مدعومة من قبل أنظمة تشغيل مختلفة مثل : الويندوز ، السولاريس الرد هات ، و غيرها . هذه اللغة غير مجانية ، بل إنها تعتبر من أكثر اللغات ارتفاعاً في السعر ! تنتج الشركة ثلاثة مستويات من الحلول : (Express, Professional, Enterprise) . المستوى الأول مفيد للاستخدام الشخصي . المستوى الثاني يناسب الشركات ذات الأعمال الصغيرة . المستوى الثالث يناسب الأعمال الكبرى . هذه اللغة يمكن دمجها مع تقنيات شبكية مختلفة و متعددة تزيد قوة اللغة . كما أن هناك بيئة مرئية مقدمة مع اللغة تسمح لك بكتابة برامجك في بيئة أفضل و أسهل .
مميزات هذه اللغة تتمثل في إمكانية دمجها مع قاعدة عريضة جداً من التقنيات المتوفرة . أما عيوبها فتتمثل في السعر المرتفع للغة بالإضافة لصعوبة تعلمها إلى حد ما .

Perl

قد لا يكون الاسم غريب عليك ، إذ أن لهذه اللغة شهرة واسعة مستمدة من قدم و قوة هذه اللغة . تم تطوير هذه اللغة بالأساس من قبل لاري وول (Larry Wall) . ثم تم تطويرها من قبل قاعدة عريضة من المطورين إذ أن اللغة مفتوحة المصدر مما يعطي فرصة أكبر لتطويرها . تعتمد اللغة بالأساس على لغات من أمثال : (C, sed , awk Unix shells, و غيرها من اللغات) . الواقع أن هذه اللغة غير مخصصة لتطبيقات الشبكة فقط ! بل يمكن أن تستخدمها لتطبيقات أخرى مختلفة . يمكنك إضافة المزيد إلى هذه اللغة عبر إدراج مكتبات إضافية (Libraries) مكتوبة بلغات أخرى كالسي و الجافا . تتغلب هذه اللغة على البي اتش بي من حيث قدم اللغة و انتشارها الواسع بين المطورين . لكن البي اتش بي تتغلب على لغة البيرل من ناحية أن بي اتش بي تم تطويرها لخدمة أغراض الشبكة فقط ، مما يعني أنها الأفضل لمثل هذا الغرض .

PHP

كانت بداية لغة البي اتش بي على شكل مجموعة من الأكواد التي قام شخص يدعى راسموس ليردورف (Rasmus Lerdorf) بتطويرها لمراقبة الأشخاص اللذين يزورون ملفه الشخصي . ازداد اهتمام المطورين بهذه الأكواد التي قام راسموس بتطويرها فقرر الأخير أن ينشرها للاستخدام العام باسم أدوات المواقع الشخصية (Personal Home Page tools) . من هنا جاء الاسم (**PHP** = **P**ersonal **H**ome **P**ages) . كانت الفكرة هي أن نشر المصدر على العامة سيسمح بتطويرها بشكل أسرع من الاحتفاظ بالمصدر مغلقاً لدى راسموس فقط ! (Open-Source) . كانت هذه الأكواد في البداية مكتوبة بواسطة لغة البيرل . أطلق على هذه المجموعة فيما بعد اسم PHP/FI . FI هي اختصار للكلمتين الإنجليزيتين (Form Interpreter) . كل ذلك كان بين عامي 1994 و 1995 .

PHP/FI 2

مع زيادة الاهتمام بالمشروع الشخصي لراسموس ، قام بتطوير المشروع و إضافة المزيد من الوظائف ، لكن باستخدام لغة السي هذه المرة . هذه المرة كان للغة القدرة على الاتصال بقواعد البيانات و تطوير صفحات آلية للشبكة العنكبوتية . و بنفس الطريقة و لنفس السبب ، قرر راسموس أن يقوم بتوزيع النسخة مجاناً على الجميع لغرضي التطوير و إصلاح الأخطاء . ظهرت هذه النسخة في منتصف عام 1997 تقريباً . انتشرت هذه النسخة انتشار واسع عبر الشبكة ، حيث تشير الإحصائيات الواردة في الموقع الرسمي للغة بأن حوالي 50.000 شبكة قد أبلغت باستخدامها لها – مما يمثل 1% تقريباً من حجم الشبكة في ذلك الحين .

PHP3

قد تكون النسخ السابقة مختلفة بشكل كبير عن الصيغة العامة للغة بي اتش بي كما نعرفها اليوم ، لكن النسخة الثالثة كانت مقاربة جداً لبي اتش بي اليوم . في الواقع لقد قام شخصان مهتمان بهذه اللغة بإصدار نسخة مطورة عنها و هما :أندي و زيف (Andi Gutmans و Zeev Suraski) . لقد قاما بإصدار نسخة معاد كتابتها بالكامل في نهاية العام 1997 . كان ذلك بعد أن استخدما النسخة السابقة من البي اتش بي في تطوير مشروع تجاري خاص بهما . في ذلك الوقت ، اتفق كل من راسموس ، أندي ، و زيف على اعتماد هذه النسخة الجديدة كنسخة رسمية تلي النسخة السابقة . قدمت هذه النسخة العديد من الإضافات المفيدة حقاً للغة ، مثل دعم المزيد من قواعد البيانات ، دعم البرمجة الشيئية و غيرها . و كالعاده ، تم نشر مصدر اللغة بشكل مفتوح لمجتمع المطورين مما زاد من شعبية اللغة و زاد من سرعة تطويرها .
هذه المرة تم نشر اللغة بالاسم المختصر PHP فقط . و أعلن أن ذلك اختصار لـ (**Hypertext Preprocessor**) . تم إصدار النسخة الرسمية من PHP3 في شهر يونيو من عام 1998 حيث كان عدد المواقع المستخدمة لها في ذلك الحين يقارب مئات الآلاف من المواقع – حوالي 10% - مع حجم مواقع الشبكة . و ما يقارب هذا العدد أو يزيد من المطورين !

PHP4

استمر أندي و زيف في تطوير اللغة . هذه المرة أعادوا كتابة مصدر اللغة مرة أخرى ! ليخرجوا لنا بالنسخة الأحدث من اللغة PHP4 . بدأت هذه الخطوة بعد نشر النسخة السابقة رسمياً مباشرة – في شتاء عام 1998 – و كان الهدف الأساسي منها هو زيادة جودة المشاريع الكبيرة . تم إضافة المزيد من الخصائص للغة ، على سبيل المثال لا الحصر :

- زيادة دعم البرمجة الشيئية .
- دعم المزيد من قواعد البيانات .
- تم إضافة خاصية الجلسات (Sessions) . سنتناول هذا الموضوع بشكل مفصل في الفصول القادمة بإذن الله .
- دعم داخلي لكل من الجافا و الاكس ام ال (XML) .
- معامل مقارنة جديد (===) ليفحص تساوي القيمة و تساوي نوع المتغير في وقت واحد .
- تمت إضافة نوع جديد من البيانات هو النوع (Boolean) .
- تمت إضافة بعض المصفوفات الجديدة التي تحمل قيم لمتغيرات هامة تخص الخادم أو البيئة المستخدمة .
- الكثير ، الكثير غير ذلك .

قدم كل من أندي و زيف محرك جديد تعتمد عليه اللغة قاموا بتسميته بمحرك زيند Zend (راجع الجزء الخاص بمحرك زيند لمزيد من المعلومات). تم نشر النسخ الأولية من اللغة لأول مرة في منتصف عام 1999 . و تم نشر النسخة الرسمية في عام 2000 ميلادية .
تشير الإحصائيات إلى نسبة المواقع المستخدمة للنسخة الرابعة تشكل حوالي 20% من عدد المواقع في ذلك الحين . و العدد في ازدياد مستمر !

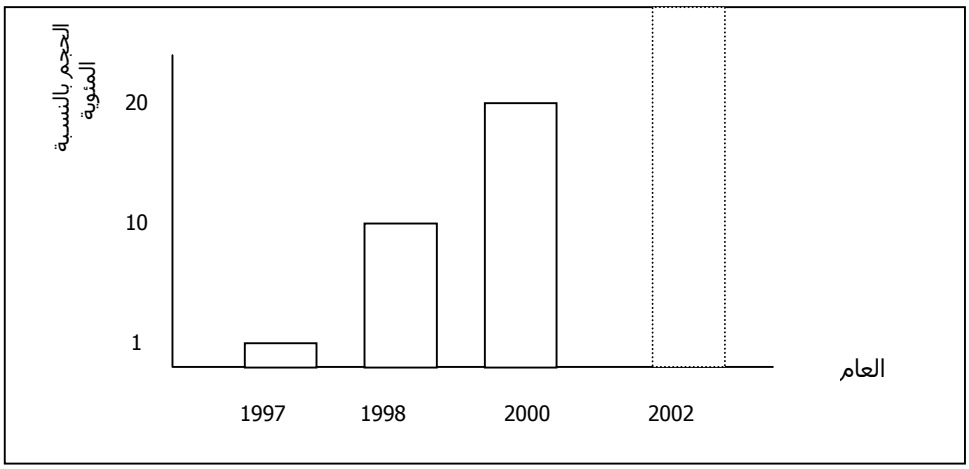
PHP5

تعتبر النسخة الرابعة هي النسخة الأخيرة من اللغة (4.2.2 تحديداً) . و مع ذلك فإننا لن ننتظر كثيراً حتى تصدر النسخة الجديدة من اللغة . كما تشير المعلومات الواردة في

الموقع الرسمي للغة ، فإن التعديلات قد بدأت بالفعل على النسخة الحالية تمهيداً لإصدار النسخة الخامسة .

معلومات إحصائية حول PHP :

ذكرنا بعض الأرقام الإحصائية التي تشير إلى حجم انتشار دعم اللغة على المواقع المختلفة على الشبكة . لنحاول تلخيص هذه المعلومات عبر هذا الرسم البياني :



محرك زند Zend :

ذكرنا بأنه في تم نشر أول نسخة من هذا المحرك في حوالي عام 1999 ميلادية على يد كل من أندي و زيف (Andi Gutmans و Zeev Suraski) . إذا كنت تتساءل عن سبب تسمية المحرك بهذا الاسم فإن الإجابة تقع في الاسمين الأولين من اسمي مطوري هذا المحرك . لابد من توضيح الفرق بين بي اتش بي و بين زند هنا ، ما هي حدود كل منهما ؟ و ما هي المجالات التي يتقاطعان فيها ؟

زند هو المحرك . بي اتش بي هي اللغة . ماذا يعني ذلك ؟

في الواقع يمكننا أن نقول بأن كلمة PHP تشمل النظام ككل كما يظهر للمستخدم الخارجي .

النظام مكون من ثلاثة أجزاء :

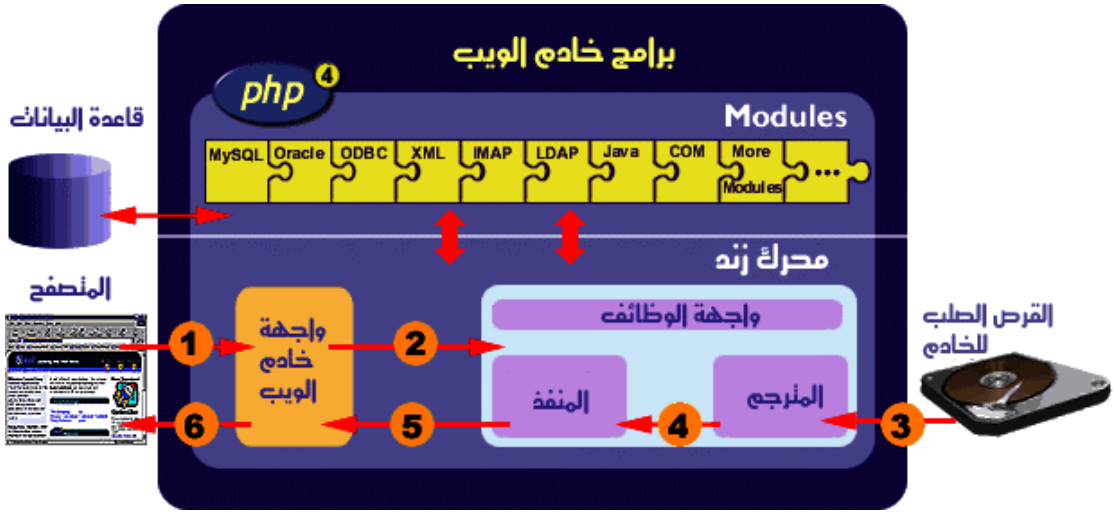
1-المترجم : و هو الجزء الذي يقوم بتحليل الكود المدخل و ترجمته و من ثم تنفيذه .

2-الجزء الوظيفي : مسؤول عن وظائف اللغة (دوالها) .

3-الواجهة : الجزء المتصل بالخادم .

بالنسبة لمحرك زند ، فإنه مسؤول عن الجزء الأول ، و القليل من الجزء الثاني فقط . أما بي اتش بي فهي مسؤولة عن الجزء الثاني و الجزء الثالث . زند و بي اتش بي معاً ، يشكلان النظام كاملاً .

قد تتضح لنا الصورة بشكل أوضح مع هذا الشكل :



شكل 74

الخلاصة :

تعرفنا في هذا الفصل على واقع الحياة على الشبكة ، حيث أصبحنا نعرف الآن أن هناك نوعين من الأجهزة : أجهزة خادمة و أجهزة عميلة . كما أن هناك نوعين من اللغات : لغات جهة العميل و لغات جهة المزود . أخذنا نبذة بسيطة عن كل لغة . أصبح لدينا القدرة على الإجابة على الأسئلة : كيف ؟ متى ؟ لماذا ؟ ماهي PHP ؟

مراجع إضافية :

يبدو أن عدد الكتب التي تتحدث عن لغة البي اتش قد زاد عن المئة كتاب حالياً ! إذ أنه يوجد ما يزيد عن 50 كتاب باللغة الألمانية (أول كتاب لهذه اللغة كان باللغة الألمانية) و ما يزيد عن الأربعين كتاب باللغة الانجليزية مع وجود نسخ مختلفة من هذه الكتب بكل اللغات !

سأذكر هنا أبرز الكتب التي أعجبتني باللغة الإنجليزية و التي قد تستفيد منها بنفسك :

- **SAMS Teach yourself PHP in 24 hours by Matt Zandstra (SAMS Publishing)**
- **Professional PHP Programming by Jesus Castagnetto, Harish Rawat, Sascha Schumann, Chris Scollo, Deepak Velialth (Wrox Press)**
- **Instant PHP4 by Micheal J. Walker, Robert M. COX, Neal Anders (Osborne)**

مواقع هامة :

أ- مواقع أجنبية :

http://www.php.net
http://www.phpbiulder.com
http://www.apache.org
http://www.mysql.com

ب-مواقع عربية :

http://www.c4arab.com
http://www.php4web.com

http://www.phpvillage.com

الفصل الخامس : تنصيب PHP

الفصل السادس : تنصيب MySQL