لغة PHP

المحتويات

مقدمة

ما هي لغة الـ PHP

إعدادات Windows XP لتحرير ملفات الـ PHP

إضافة لغة الـ PHP 4.4.2 للسرفر IIS

تركيب برنامج السيرفر الشخصي

مدخل إلى الـ PHP

المتغيرات في الـ PHP

أنواع البيانات

المعاملات

الثوابت

دوال الوقت والتاريخ في الـ PHP

العبارات الشرطية

عبارات التكرار

الصور في لغة الـ PHP

التعامل مع النماذج

المصفوفات

التحكم بالنصوص وإدراج الملفات

قواعد البيانات

التعامل مع الملفات والمجلدات

معالجة الأخطاء

أمثلة متنوعة

مقدمة

كانت اللغة المستخدمة حين ظهور الانترنيت هي لغة HTML هذه الصفحات كانت تحتوي على روابط للتنقل من معلومة لأخرى. ومع مرور الوقت، أصبحنا بحاجة إلى المزيد! فظهرت تقنيات مختلفة لتحسين الوضع..

ويمكن تقسيم هذه اللغات إلى نوعين رئيسيين:

أولاً . 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.

ميزة هذا النوع من اللغات هو السرعة في التنفيذ وعيبه أن المصمم ليس لديه تحكم كامل في نوعية المتصفح المستخدم لترجمة ما كتبه من أكواد!

ثانياً . اللغات جهة المزود (الخادم):

يطلب المستخدم صفحة معينة، يتم جلب هذه الصفحة أو المعلومة من القرص الصلب للجهاز الخادم ثم يقوم الجهاز الخادم بترجمة المعلومات المطلوبة وتحويلها إلى لغة الـ HTML ثم يرسلها للعميل. ميزة هذا النوع من اللغات هو التحكم الأكبر في البرنامج المستخدم للترجمة، لكن يوجد عيوب تتمثل في زيادة الوقت المطلوب لوصول الاستجابة.

وسنستعرض الآن هذه اللغات بشكل سريع.

HTML: اختصار لـ Hyper Text Markup Language التي تعني 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. هذا يحدث عادة بسبب إضافة المزيد من الوسوم بشكل سريع أو بسبب أن الوسم منتج من الشركة المصنعة للمتصفح وبالتالي تكون داعمة له في بقية منتجاتها أيضاً بينما لا تدعمه بقية الشركات المنافسة لها. ولكن محدودية هذه اللغة أدت إلى ظهور لغات أخرى فالجملة التي يكتبها مصمم الموقع ستظهر كما كتبها لدى جميع المستخدمين وفي كل وقت! لنفرض أنه كتب الوقت 4.30 و الذي يشير إلى وقت كتابته لهذه الجملة، ماذا سيحدث عندما يطّلع المستخدم على صفحته بعد ساعة ؟ ستظل الساعة تشير إلى الرابعة والنصف. بينما كان بإمكاننا أن نجعل هذا المحتوى متحرك يتغير بتغير الزمن أو بتغير المستخدم.

العيب الآخر للغة الـ HTML أنها لا تفصل المحتوى عن التنسيق بينما يوجد حلول أخرى تقوم بذلك. والحل قد لا يكون بالاستغناء عن هذه اللغة تماماً لكن بدمجها مع لغات أخرى مثل لغات البرمجة الخفيفة الجافا سكريبت وشبيهاتها أو لغات البرمجة الثقيلة PHP, ASP...

DHTML : اختصارها Dynamic Hyper Text Markup Language التي تشير إلى لغة ترميز النص التشعبي الكلية. ظهرت هذه اللغة كحل لثبات المحتوى في لغة الـ HTML. حيث أصبح بإمكان المصمم أن يضيف بعض الحركة لصفحته. الأمثلة على مثل هذه الإضافات كثيرة جداً ومحدودة بخيال المصمم فقط. يمكنك أن تجعل إحدى مكونات صفحتك تستدير باتجاه معين، أو تغير لون العناصر، أو تضيف تأثير الأمواج أو الخلفية الشبه شفافة، باختصار: تستطيع أن تجعل كل عنصر من عناصر صفحتك يتفاعل مع المستخدم. هذه اللغة تعتبر من لغات جهة العميل أيضاً.

ActiveX Controls: مكونات الأكتيف اكس هي عبارة عن برمجيات صغيرة يمكن صنعها بواسطة لغتي الـ C++ 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. كما يمكنك إضافة هذه المكونات إلى صفحات الـ HTML. هذه اللغة لا يمكن اعتبارها من لغات جهة العميل أو لغات جهة الخادم، حيث يعتمد ذلك على ما يقوم به هذا الكائن.

CGI: اختصار لـ **Common Gateway Interface** التي تشير إلى واجهة البوابة العامة. تعتبر هذه اللغة من أقدم وأشهر اللغات المستخدمة لتطبيقات الإنترنت جهة الخادم. ستجد أن هذه اللغة مدعومة في كل شركات الاستضافة الحالية! يمكنك كتابة برامج الـ **CGI** باستخدام أي لغة تقريباً، إلا أن أشهر اللغات المستخدمة لذلك هي لغة البيرل.

CGI تكون بمثابة الواجهة أو البوابة التي تربط الخادم بالبرنامج.

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.

ASP: اختصار لـ **Active Server Pages** التي تشير إلى صفحات الخادم النشطة. هذه اللغة تعتبر منافس قوي للغة الـ PHP. كما أنها تتخذ نفس أسلوب الـ PHP من حيث أن أكواد اللغة التي تكون مدمجة من أكواد لغة الـ HTML في ملف واحد.

عيوب هذه اللغة تتمثل في أنها لغة غير مجانية، وغير مفتوحة المصدر مما يعني أن الأخطاء لا يتم إصلاحها في الوقت المناسب، بالإضافة لمحدودية تطوير اللغة والعيب الأكبر هو أنك لا تستطيع استخدامها إلا على خادم يحتوي على نظام التشغيل ويندوز مع برنامج خادم من إنتاج شركة مايكروسوفت (IIS, PWS).

JSP: اختصار لـ **Java Server Pages** التي تشير إلى صفحات الجافا جهة الخادم. وهي إحدى لغات جهة الخادم كما يشير الاسم. تعتمد هذه اللغة على فصل المحتوى عن طريقة العرض. تنتهي صفحات الجافا جهة الخادم بالامتداد jsp. في العادة وعند طلب إحدى صفحات الجافا من الخادم، يقوم الخادم بترجمة الكود ويرسل 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، والسبب يكمن في أنها قد تكون صعبة بعض الشيء على من لم يعتد على البرمجة بلغة الجافا.

ColdFusion: هذه إحدى اللغات المنافسة للغة الـ PHP صممت هذه اللغة من أجل هدف واضح هو تقديم حلول عملية لتطبيقات الشبكة العنكبوتية. تم تطوير اللغة من قبل شركة أليير (Allaire) . اللغة مدعومة من قبل أنظمة تشغيل مختلفة. هذه اللغة غير مجانية، بل إنها تعتبر من أكثر اللغات ارتفاعاً في السعر! تنتج الشركة ثلاثة مستويات من الحلول

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 ومتعددة تزيد من قوتها.

كما أن هناك بيئة مرئية مقدمة مع اللغة تسمح لك بكتابة برامجك في بيئة أفضل وأسهل.

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: 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 وغيرها من اللغات.

تتغلب هذه اللغة على الـ php من حيث قدم اللغة و انتشارها الواسع بين المطورين . لكن الـ PHP تتغلب على لغة البيرل من ناحية أنها تم تطويرها لخدمة أغراض الشبكة فقط، مما يعني أنها الأفضل لمثل هذا الغرض.

PHP: كانت بدايتها 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 Personal Home الأخير أن ينشرها للاستخدام العام باسم أدوات المواقع الشخصية (**PHP = Personal Home Pages**) والفكرة هي أن نشر المصدر على العامة سيسمح بتطويرها بشكل أسرع من الاحتفاظ بالمصدر مغلقاً لدى راسموس فقط.

ما هي لغة الـ PHP

ظهرت هذه اللغة على يد شخص يدعى Rasmus Lerdorf وقد طورها أشخاص آخرين وخضعت لثلاث عمليات تنقيح حتى وصلت إلى الناتج الذي نراه في أيدينا هذه الأيام.

كان انتشارها ملحوظاً جداً وبسرعة لم يسبق لها مثيل وفي أول عام 2001 أصبحت يستخدمها حوالي 5 مليون في جميع أنحاء العالم وهذا الرقم في زيادة مستمرة يمكنك متابعتها ومعرفة عدد مستخدميها هذه الأيام من خلال الموقع الرسمي لها على الرابطة التالية **http://www.php.net/usage.php** وهي لغة تندرج تحت اللغات مفتوحة المصدر open source أي يمكنك الوصول للشفرة المصدرية لها واستعمالها وتعديلها وأعاده توزيعها بدون دفع أي مبلغ.

PHP هي اختصار للنص Personal Home Page ولكنها تم تعديلها لتصبح hyper text preprocessor وهي في تطوير مستمر. وهي لغة نصوص برمجية في جانب الخادم وتم تصميمها خصيصاً للويب ويمكن تنفيذها ضمن أكود الـ HTML أو تنفذ وحدها كأي لغة برمجة أخرى ويتم تفسيرها بواسطة مترجم أو Compiler لتوليد ناتج عمليات المعالجة وغالباً يتم إخراج الناتج في شكل صفحات HTML.

هذا المترجم أو الـ Compiler يعتبر جزء من البرمجيات المثبتة على خادم الويب Server المستضيف للموقع. كما يمكن تثبيتها على أنظمة تشغيل عديدة وهي تعمل كوحدة نمطية فعالية مع ملقم Apache وأيضاً يمكن تثبيتها مع خادم Microsoft IIS وهي تعمل أيضاً مع أغلب قواعد البيانات المتوفرة.

أما عن منافسيها فتوجد أيضاً العديد من اللغات تستخدم للتطوير للوب في جانب الخادم منها asp و perl و java server و Cold fusion.

تحتل الـ PHP المكانة الأولى لما يميزها عن غيرها من نقاط قوة منها:

- تعتبر لغة PHP من أسهل لغات البرمجة تعلما، فهي تريحك من جميع تعقيدات إدارة الذاكرة وتعقيدات معالجة النصوص الموجودة في C من جهة، والكثير من الضعف الموجود في بينية وتصميم لغة البرمجة Perl من جهة أخرى.
- تمتلك لغة PHP بنية وقواعد ثابتة وواضحة جداً، فمعظم قواعد اللغة مأخوذة من كل من C و Java و Perl لصنع لغة برمجة عالية السهولة والسلاسة دون فقدان أي من القوة في اللغة.
- لغة PHP من اللغات المعروفة بسرعتها العالية في تنفيذ البرامج، حيث تمت كتابة مترجم PHP من الصفر ليعطي أداءً في منتهى الروعة، وهي مصممة أصلاً كنواة لمترجم، بحيث يمكن أن تضع هذه النواة في عدة قوالب أو أغلفة لتعمل مع التقنيات المختلفة، فيمكنك تشغيل مترجم PHP كبرنامج CGI مثلاً، ولكن الأفضل هو إمكانية تركيب مترجم PHP على مزود IIS في صورة وحدة إضافية تضاف إلى المزود عن طريق دوال ISAPI، وتوجد نسخة أخرى منه تركب على مزود Apache أيضاً في صورة وحدة خارجية، وتوجد أيضاً نسخة مخصصة للدمج مع شفرة مزود Apache بحيث تصبح جزء من برنامج Apache نفسه، وهي الطريقة الأكثر استخداماً الآن في مزودات الويب التي تعمل على أنظمة UNIX وهي الطريقة التي تعطي أفضل أداء لمترجم PHP، حيث يصبح المترجم جزء من المزود، وبالتالي فإنه سيكون محمل في الذاكرة بانتظار صفحات PHP ليقوم بترجمتها وعرضها للزوار مباشرة دون التأخير الإضافي الذي تتطلبه البرامج الأخرى.
- يأتي مترجم PHP مزود بعدد هائل من الدوال الجاهزة الاستخدام في جميع المجالات، من دوال المعالجة الرياضية والحسابية إلى دوال الوصول إلى قواعد البيانات ومزودات FTP، توفر لك دوال PHP مثلاً وصولاً إلى

مزودات البيانات MySQL و PostgreSQL و MS SQL و Oracle وغيرها من مزودات قواعد البيانات، وهنالك أيضاً مجموعة من الدوال لمعالجة ملفات XML، ودوال أخرى لإرسال واستقبال الملفات عن بعد باستخدام بروتوكول FTP، وهنالك مجموعة من الدوال لمعالجة وإنتاج الصور وملفات Flash ديناميكياً، إضافة إلى جميع الدوال الخاصة بمعالجة النصوص والمصفوفات.

- كما قلنا سابقاً، فعلى الرغم من أن هنالك الكثير من نسخ PHP التي يعمل كل منها في بيئة مختلفة، إلا أنها جميعاً تشترك في النواة الأصلية التي تقوم بالمعالجة الحقيقة لملفات PHP لذا فإن جميع مترجمات PHP تتصرف بنفس الطريقة فيما يتعلق بتنفيذ السكريبتات، فإذا كان السكريبت الذي أنشأته يعمل على نظام Windows مع مزود IIS فيجب أن يعمل دون الحاجة لأية تغييرات عند نقله إلى مزود Apache، بالطبع تظل بعض الأمور البسيطة جداً التي توفرها بعض المزودات دون غيرها.

- يوفر PHP الكثير من المزايا المتقدمة، ولكنه يوفر لك الطرق المناسبة لوضع الحدود على هذه المزايا، فيمكنك التحكم بعدد الاتصالات المسموحة بقاعدة البيانات مثلاً، أو الحجم الأقصى للملفات التي يمكن إرسالها عبر المتصفح، أو السماح باستخدام بعض الميزات أو إلغاء استخدامها، كل هذا يتم عن طريق ملف إعدادات PHP والذي يتحكم به مدير الموقع.

- يمكن توسيع مترجم PHP بسهولة وإضافة الميزات التي تريدها إليه بلغة C، وحيث أن الشفرة البرمجية للمترجم مفتوحة فإنك تستطيع تغيير ما تريده مباشرة لتحصل على النسخة التي تناسبك من المترجم، ويمكنك أيضاً عمل الوحدات الإضافية التي تركب على المترجم لزيادة ميزاته والوظائف المبيته فيه، وفي قد قام فريق تطوير مترجم PHP مسبقاً بعمل هذه المهمة وتحويل كمية ضخمة من المكتبات المكتوبة بلغة C إلى مكتبات مخصصة لتضاف

إلـى المتـرجم، ومنهـا حـصلنا علـى جميـع الميـزات التـي تحـدثنا عنهـا مثـل الوصول إلى قواعد البيانات ومعالجة ملفات XML.

ABAHE

إعدادات Windows XP لتحرير ملفات الـ PHP

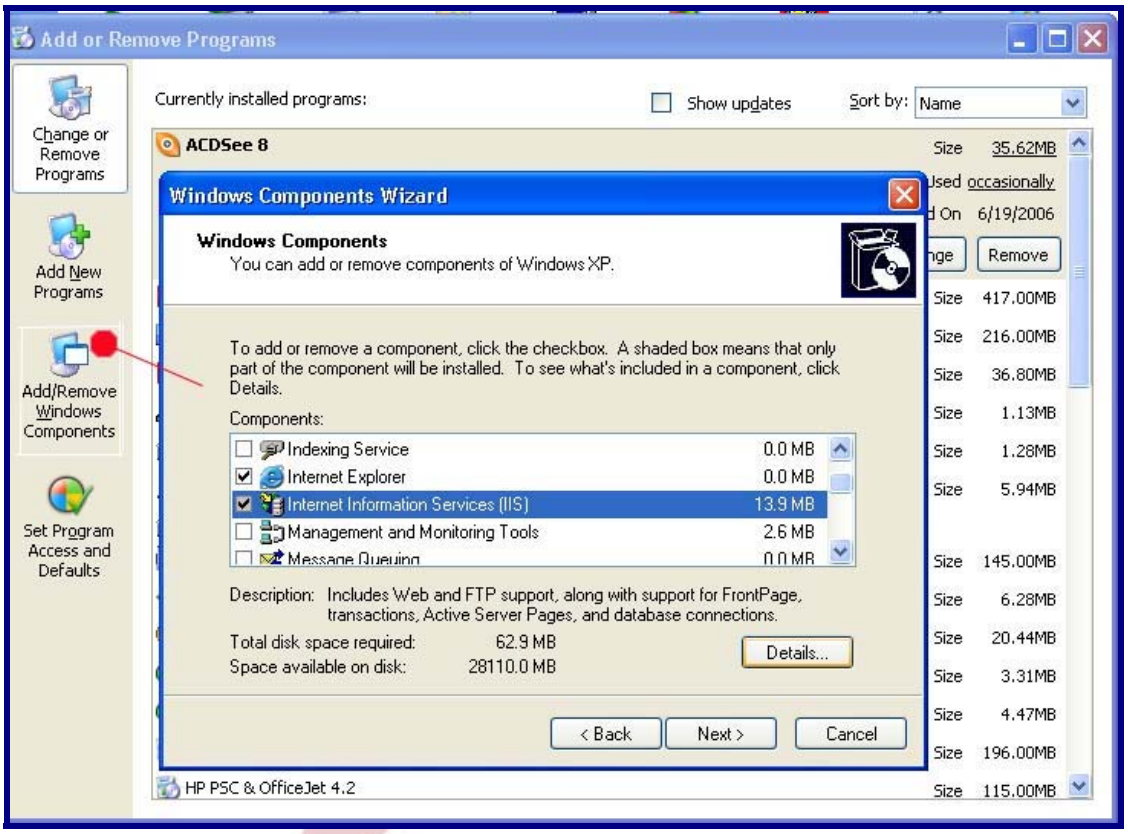
ملفات PHP هي ملفات نصية بسيطة تماماً كما هي ملفات HTML، يمكنك كتابة سكريبت PHP بأي برنامج كتابة نصوص يتيح لك كتابة الملفات النصية البسيطة Plain Text مثل Notepad على النظام ويندوز، ولكن أغلبية مبرمجي PHP يستخدمون أدوات أخرى تسهل عليهم عملية البرمجة عن طريق تلوين الشفرات البرمجية، وتسهل عملية البحث عن الملفات واستبدال المقاطع من عدة ملفات في نفس الوقت، مثل HomeSite، على الرغم من أنك لن تحتاج إلى الكثير من هذه الميزات إلا أن استخدام Notepad في عمل ملفات PHP يعتبر أمرا صعباً نوعاً ما وخاصة في الملفات الضخمة حيث أن Notepad لا تتيح فتح الملفات الكبيرة، والمشكلة الأكبر هي أنها لا توفر ترقيماً للأسطر، فإذا ظهرت لك رسالة الخطأ تشير إلى وجود خطأ في السطر 53 فلن تستطيع معرفة السطر المطلوب في Notepad إلا إذا قمت بالعد يدوياً من السطر الأول وحتى 53 .. حسناً ماذا لو كان الخطأ في السطر 652، يمكنك البدء بكتابة السكريبتات بالبرنامج المتوفر الآن إلى أن تحصل على برنامج آخر، يمكنك بالطبع فتح ملفاتك بأي محرر نصوص، فإذا كتبتها باستخدام Notepad فهذا لا يعني بأنك ملزم باستخدام Notepad في جميع ملفاتك أو حتى في هذا الملف.

لعمل ملف PHP الآن قم بفتح محرر النصوص الذي اخترته وابدأ بكتابة الصفحة التي تريدها، ولا تنسى إحاطة شفرات PHP بالوسوم الخاصة بها، ثم احفظ الملف في أي مكان في دليل مزود الويب الخاص بك وأعطه الامتداد php. حسب إعدادات المزود، ثم قم بزيارة الصفحة باستخدام المتصفح وستجد الصفحة وقد تمت ترجمتها وعرضها عليك.

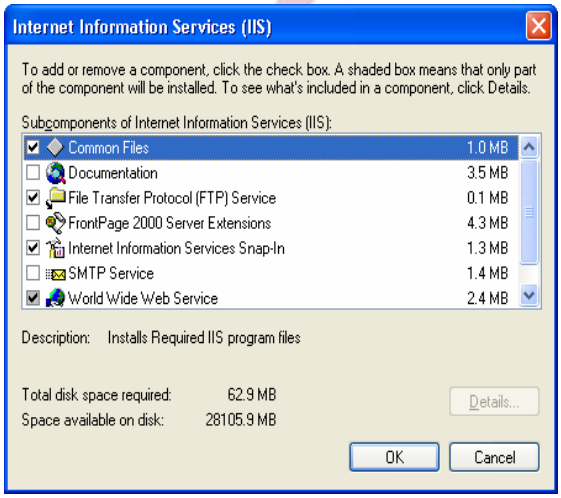
تذكر بأنك يجب أن تزور الصفحة مرور بمزود الويب، ولا يمكنك عرض الصفحة عن طريق فتحها كملف خارجي.

إلا أنني أفضل أن نقوم بتجهيز نظام التشغيل قبل البدء بالتعامل مع صفحات الـ PHP وذلك باتباع الخطوات التالية:

- نضع القرص الليزري الخاص بنظام Windows في محرك الأقراص.
- اذهب إلى الـ Control Panel وأختار Add or Remove Programs



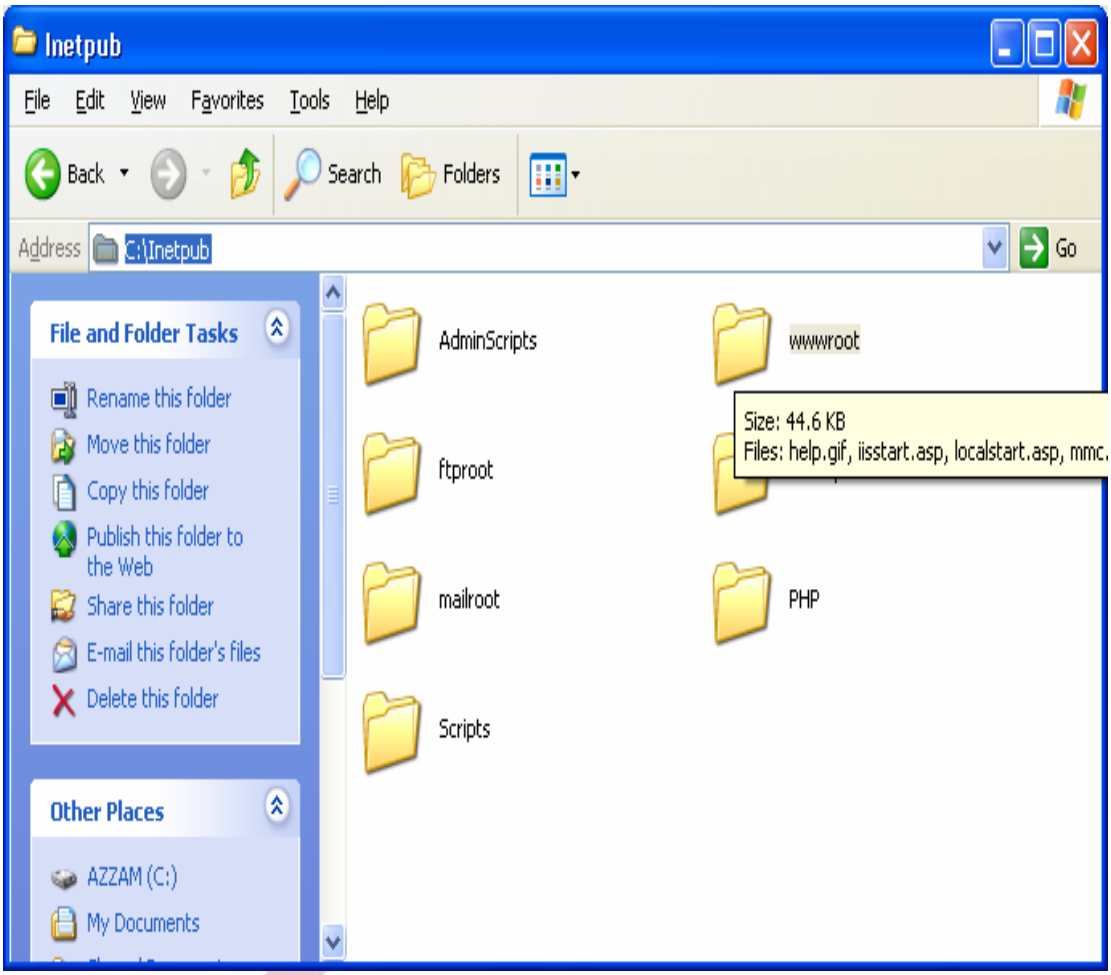
- تظهر لنا شاشة حسب الشكل أعلاه نختار منها Internet Information Services



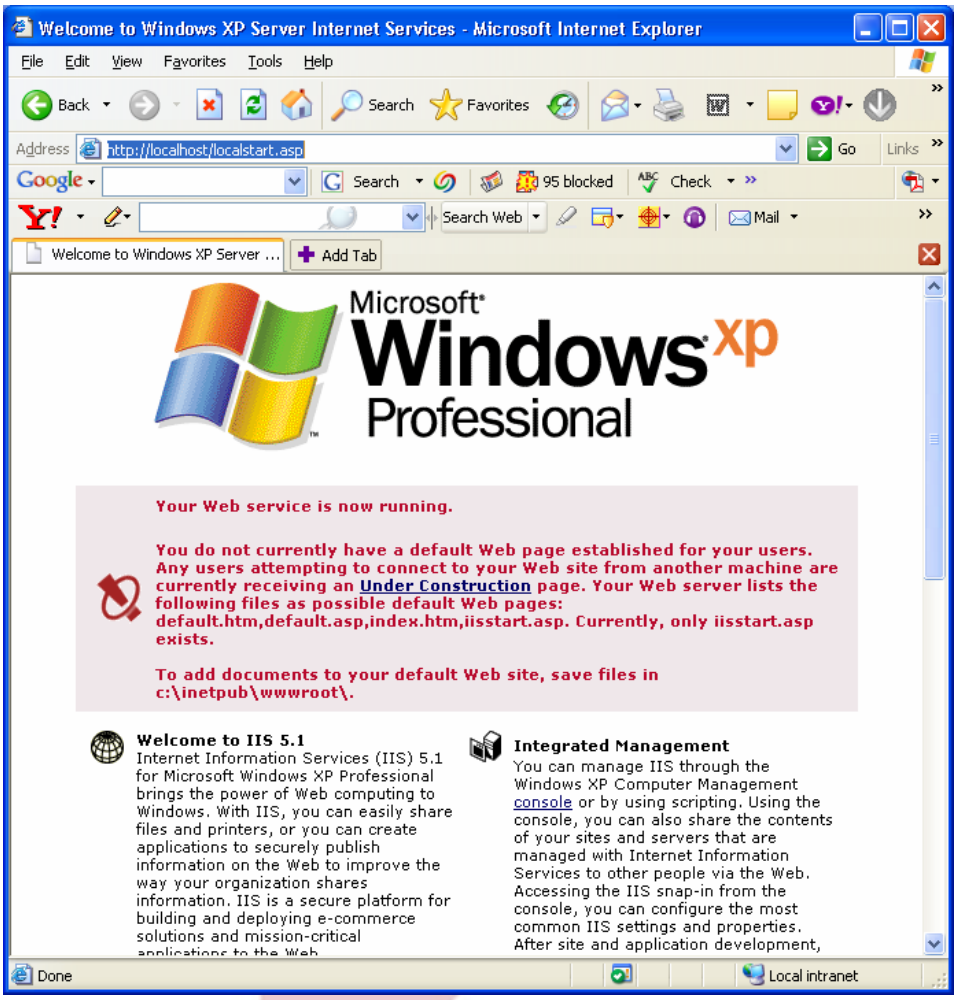
- إن نقرت على Internet Information Servicesمرتين وكأنك تفتح ملف، ستلاحظ وجود العديد من الخدمات ومن ضمنها Frontpage 2000 Server Extensions للذين يريدون استخدام الفرونت بيج.
- بعد التحميل يستحسن إعادة تشغيل

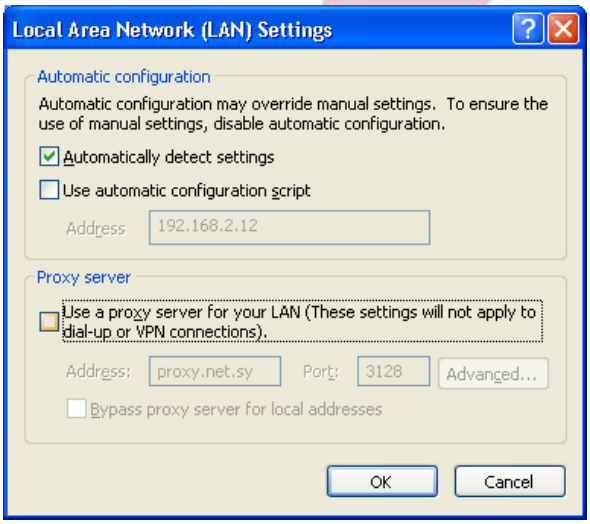
الكمبيوتر حتى تكون عملية تثبيت وتعريف السيرفر صحيحة.

- بعد إعادة التشغيل، سنلاحظ ظهور مجلد جديد في محرك الأقراص C باسم Inetpub وبداخله مجلد اسمه wwwroot .



- افتح برنامج الـ Internet Explorer وضع http://localhost/ تجد أن الملفات الموجودة في مجلد wwwroot قد ظهرت وهي تحتوي على ملفات مساعدة.





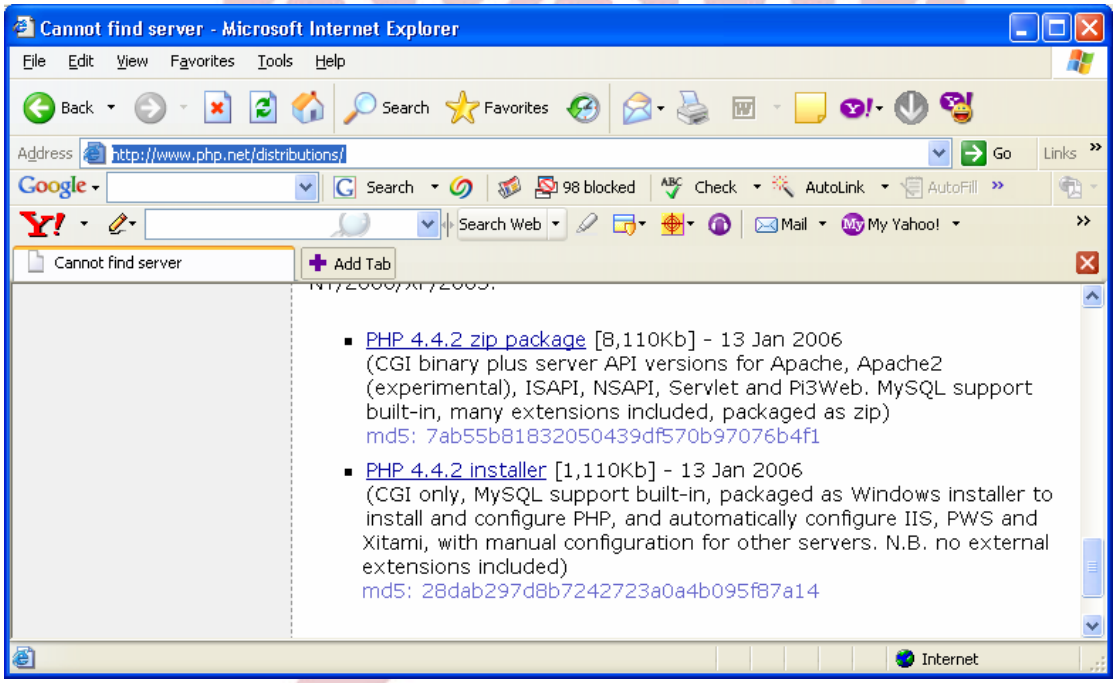
ملاحظة هامة: في حال كنت تستخدم إعدادات خاصة بمزود الانترنيت في خيار Lan Settings في لوحة إعدادات الاتصال بالانترنيت فيجب إزالة هذه الإعدادات حتى تتمكن من مشاهدة الصفحة أثناء عملك على الجهاز.

إضافة لغة الـ PHP 4.4.2 للسيرفر IIS

إذا تمت الخطوات السابقة بنجاح فالسيرفر الآن يدعم لغة الـ asp و قاعدة البيانات Access فقط ولتحميل PHP 4.4.2 علينا القيام بالخطوات التالية:

- تحميله من الموقع عن طريق هذا الرابط:

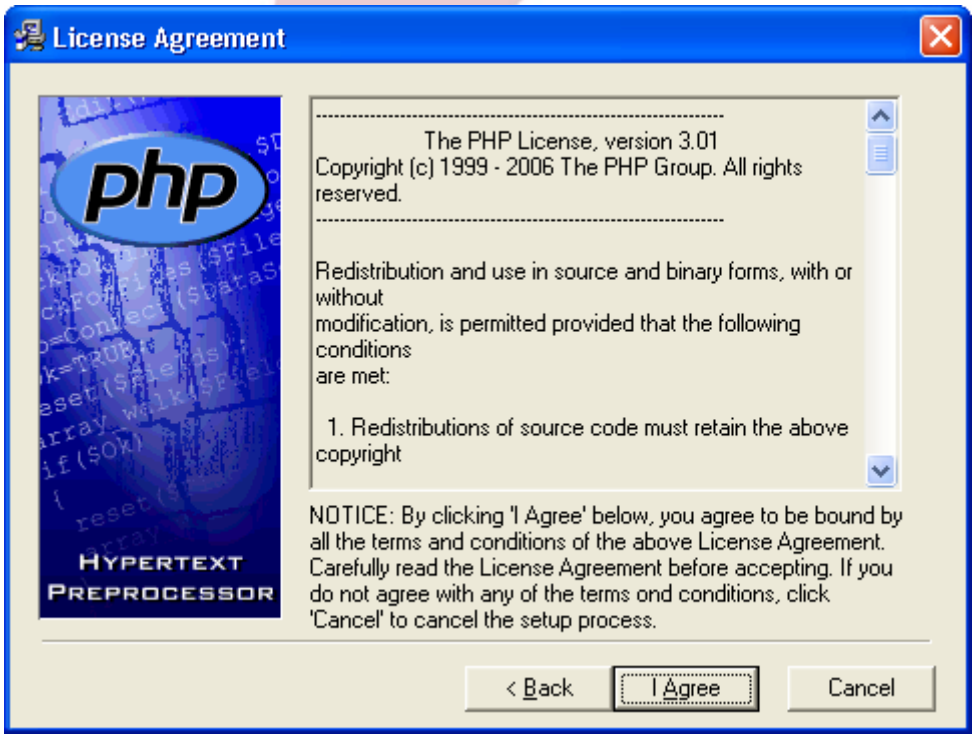
http://www.php.net/distributions/php-4.4.2-installer.exe



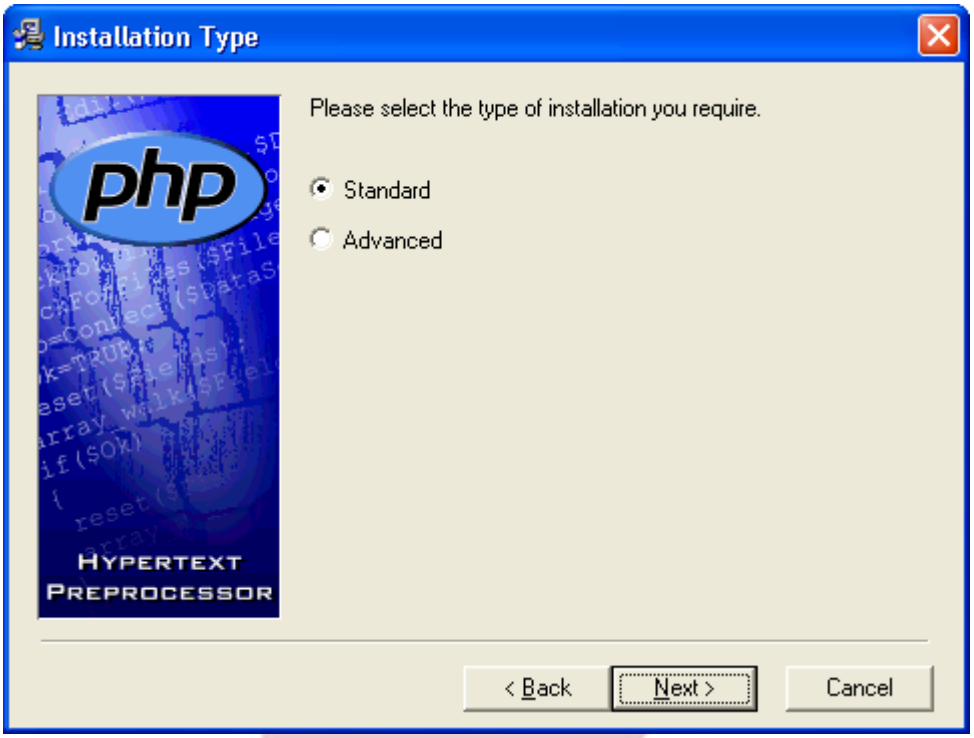
- بعد تحميل البرنامج من الموقع يظهر لنا على الشكل التالي:

php-4.4.2-installer.exe

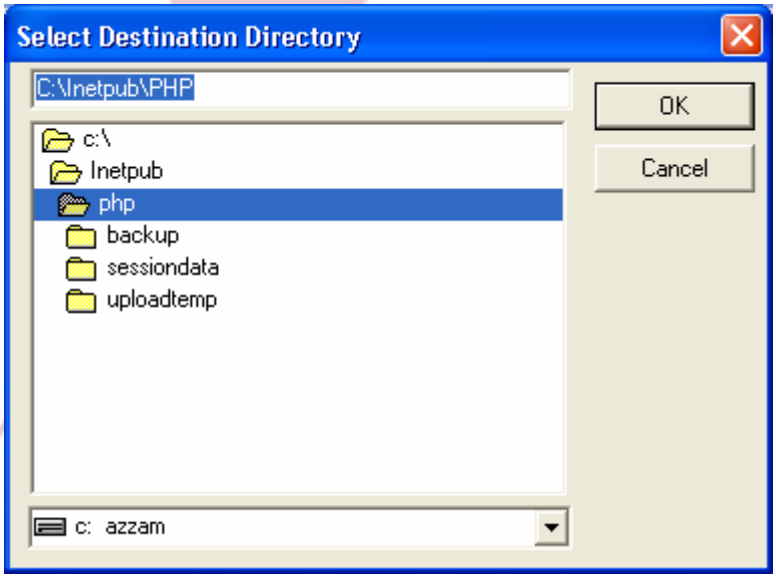
- نضغط على البرنامج ونبدأ التحميل.

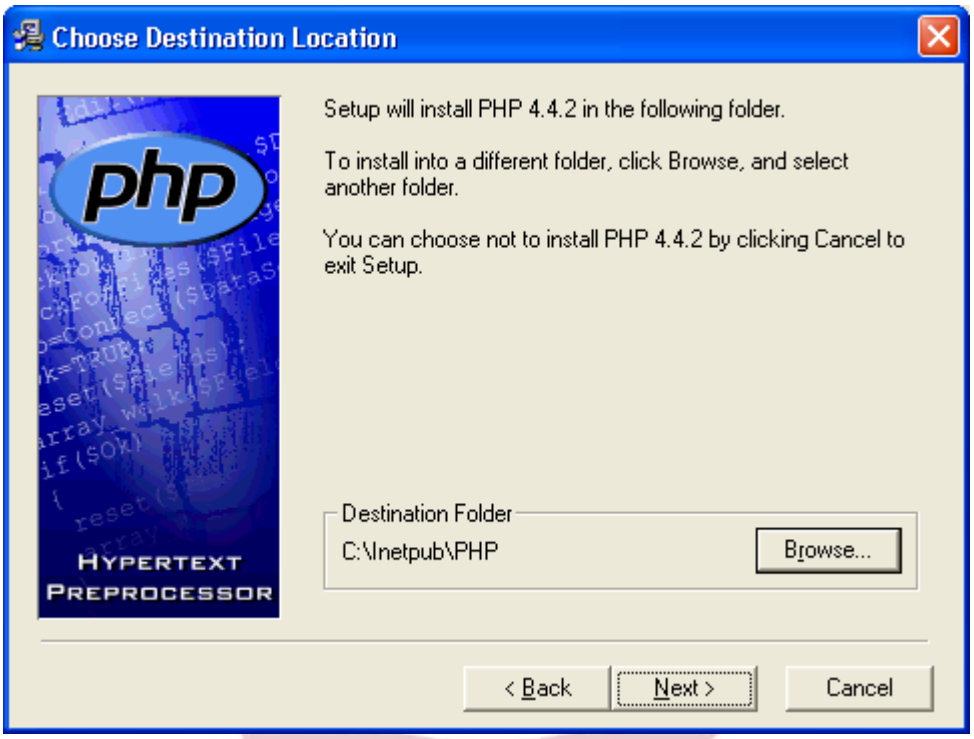


من اللوحة التالية نختار Standard ثم Next:

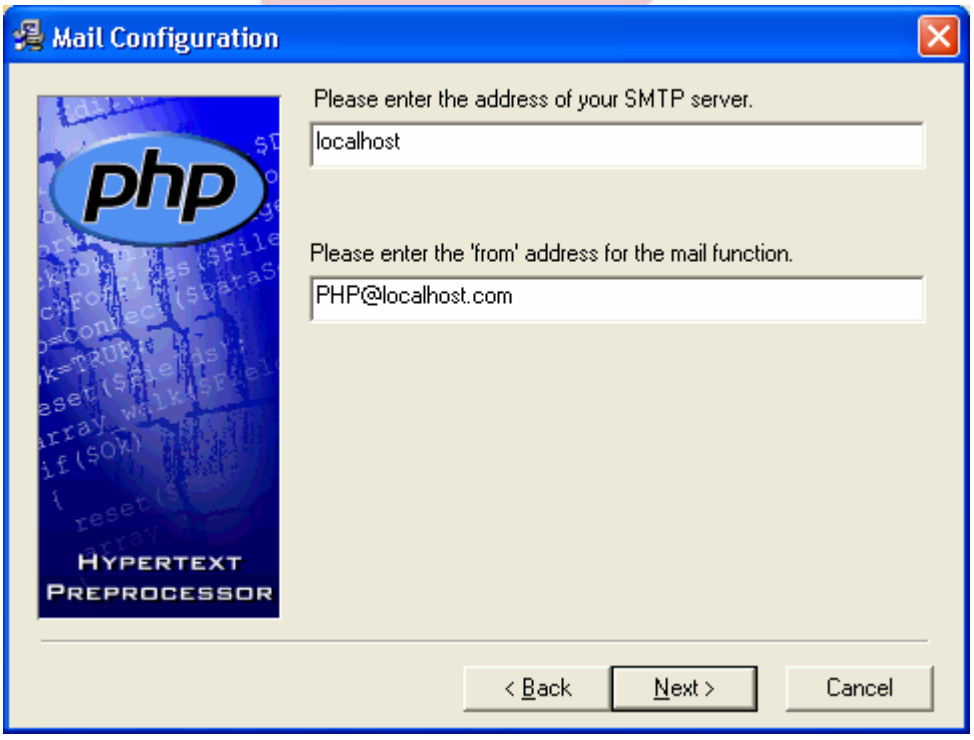


تظهر لوحة نختار منها Browse ثم نعدل المسار حسب الشكل:

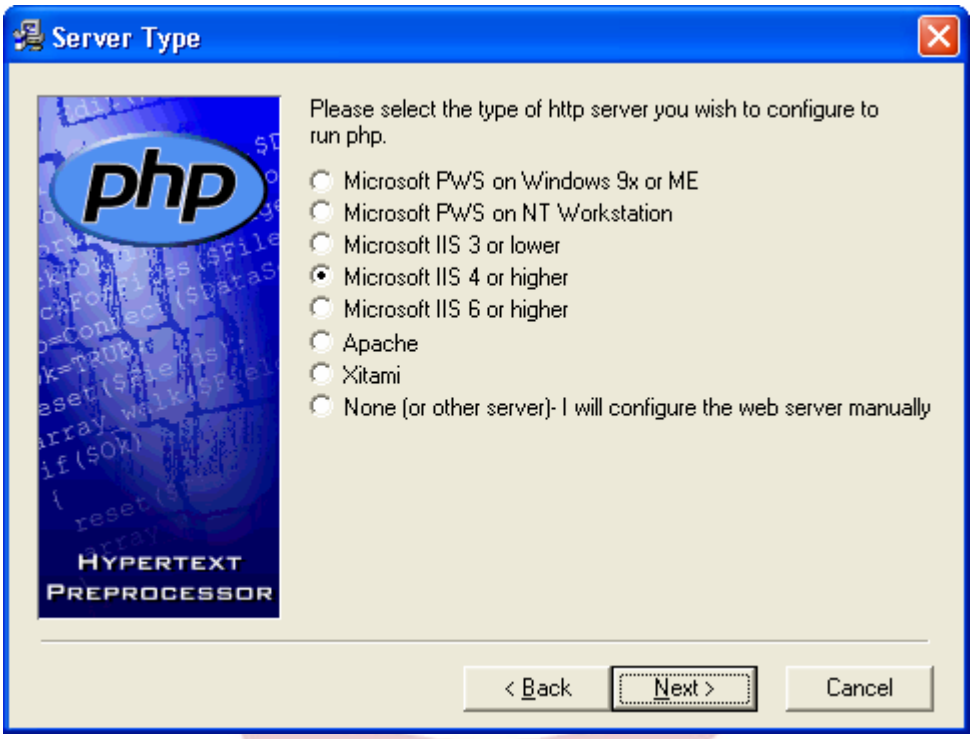




نضغط Next:

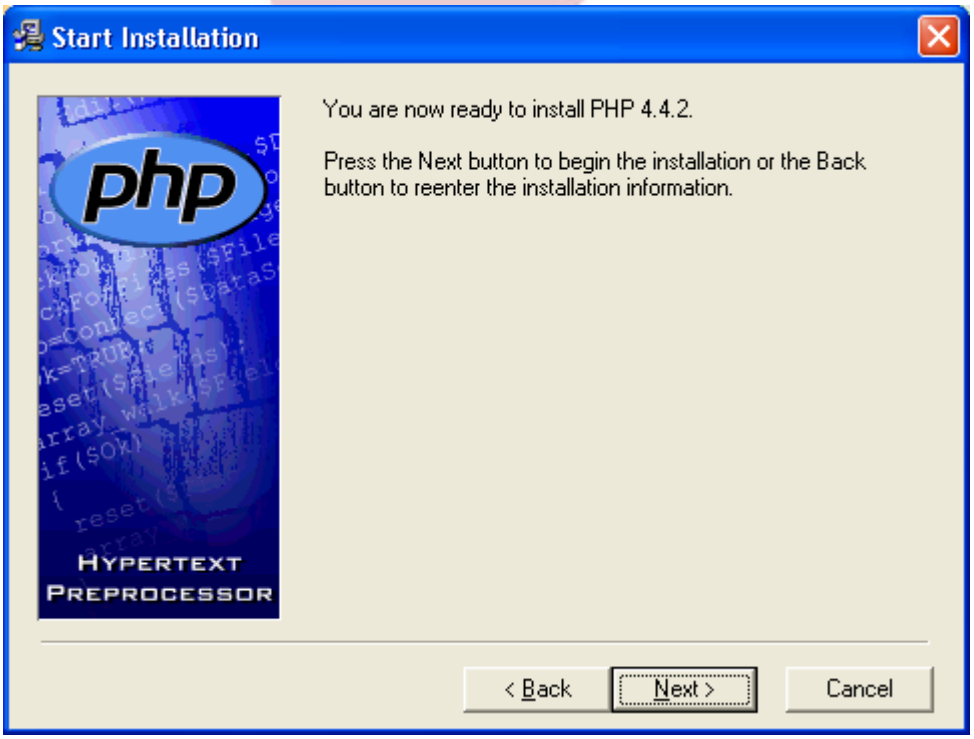


بعد التعديل حسب الشكل أعلاه نضغط Next:

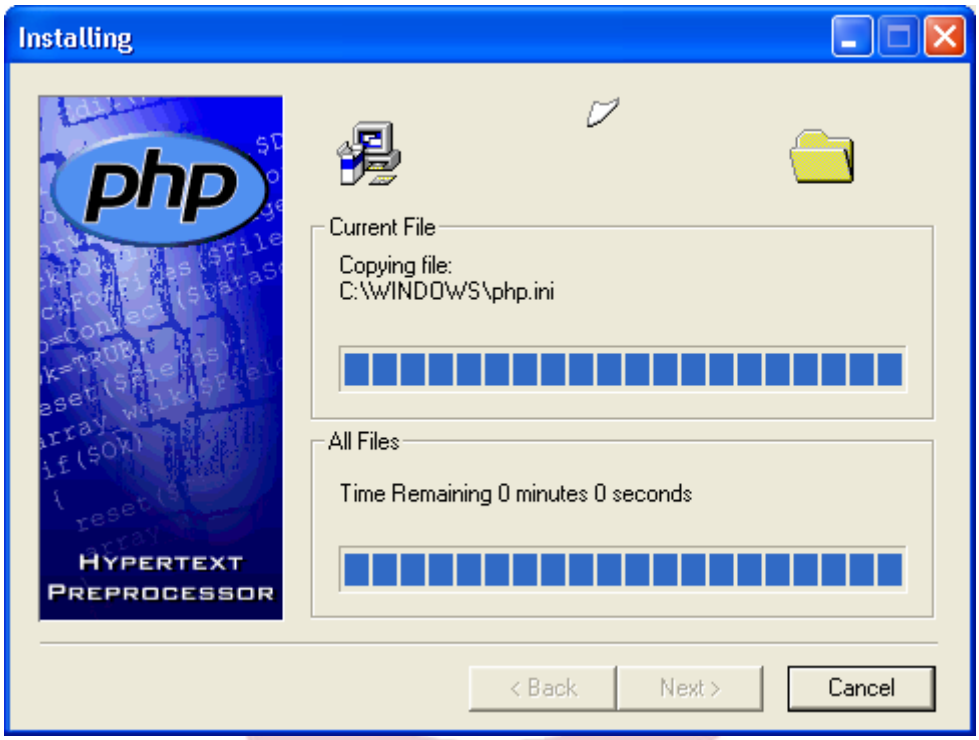


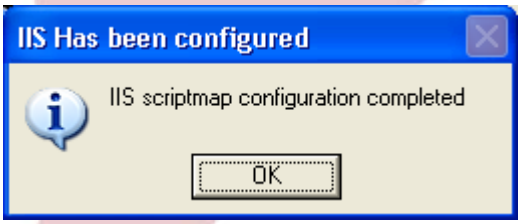
بما أننا نعمل على Windows Xp نختار الخيار رقم 4 ثم Next.

الآن تصبح عملية الإعداد للتحميل جاهزة...



عند ضغط Next يبدأ التحميل ويظهر لنا العداد الخاص بذلك.


Installing
Current File
Copying file:
C:\WINDOWS\php.ini
All Files
Time Remaining 0 minutes 0 seconds
< Back
Next >
Cancel
HYPERTEXT
PREPROCESSOR


IIS Has been configured
IIS scriptmap configuration completed
OK

Installation complete
PHP 4.4.2 has been successfully installed.
Press the OK button to exit this installation.
NT users may need to set appropriate permissions for the various php files and directories. Usually IUSR_MachineName (or the user your web server runs as) will need read write access to the uploadtmp and session directories, and execute access for php.exe and php4ts.dll.
OK

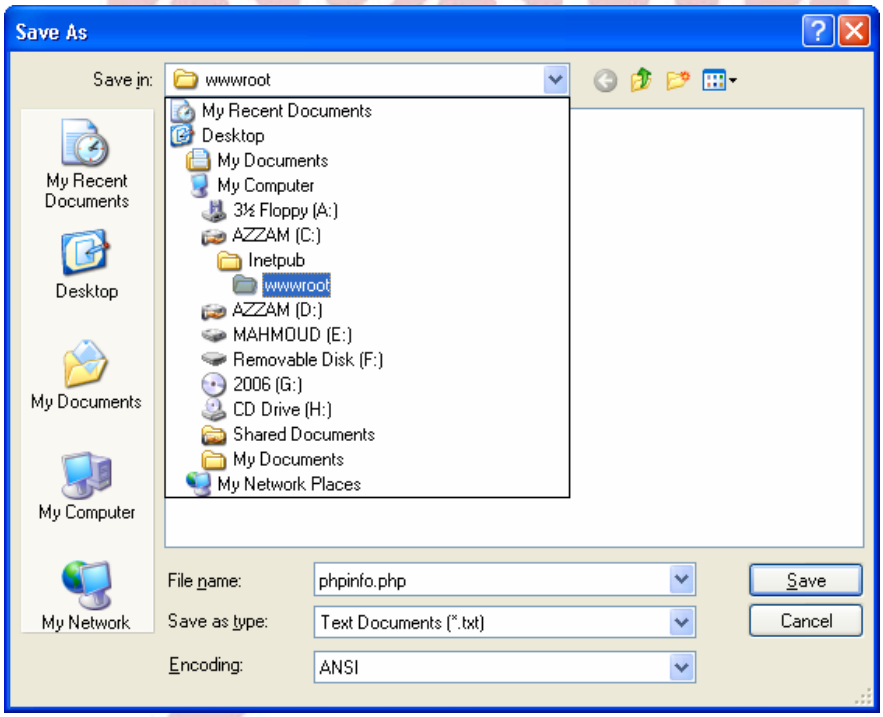
فحص وتجريب الـ PHP على جهازك:

يمكننا التأكد من إتمام عملية التحميل ونجاحها من خلال المثال التالي حيث يمكننا من خلاله فتح ملف المعلومات الخاص بـ PHP 4.4.2. وذلك كما يلي:

نفتح ملف جديد في المفكرة ونكتب به الكود التالي فقط:

```
<?
Echo phpinfo();
?>
```

الآن نحفظ الملف باسم phpinfo.php وذلك حسب الشكل التالي:

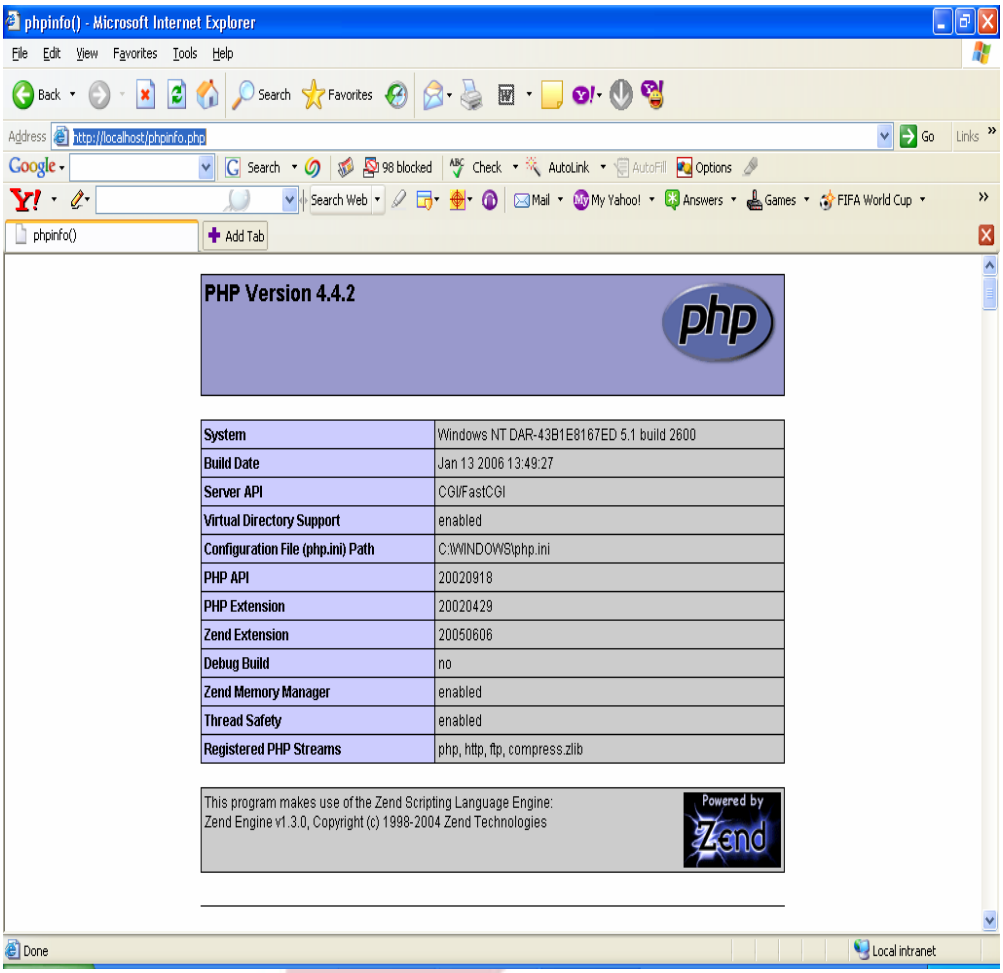


لاحظ المسار الذي تم حفظ الملف به..!!

افتح متصفحك الآن واكتب:

http://localhost/phpinfo.php

ستظهر لك الصفحة التالية:



- في الإصدارات الأقدم ستكون بحاجة إلى بعض التعديلات في الملف php.ini لتحقيق التوافق بين الـ php والمحررات التي ستستعملها. ولكن في الإصدار 4.4.2 لا داعي إلى أي تعديل فهو يقوم بالمطلوب تلقائياً.

زيادة في التأكيد ننشئ ملف جديد على المفكرة ونضع عليه الكود البسيط التالي:

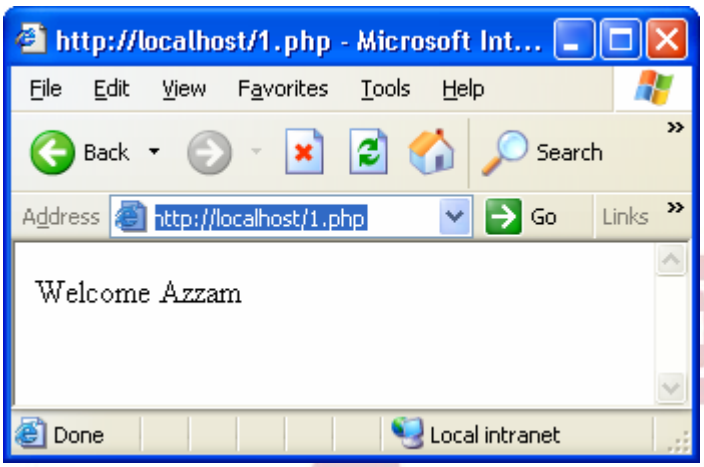
```
<?
echo "Welcome Azzam";
?>
```

وبالطبع نحفظ الملف بصيغة php. وليكن 1.php

نعود إلى المستعرض ونكتب العنوان:

http://localhost/1.php

فتظهر الصفحة على الشكل التالي:



نحن الآن متأكدين من فعالية الـ PHP.

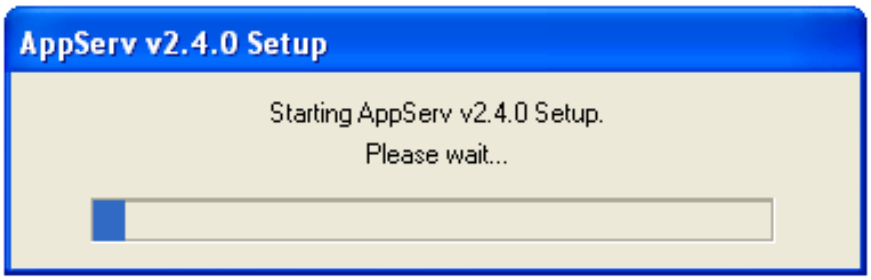
تركيب برنامج السيرفر الشخصي APPSERV

قم بتحميل برنامج APPSERV عن طريق الرابط التالي:

http://prog.arccn.net/modules.php?name=Downloads&d_op=getit&lid=3

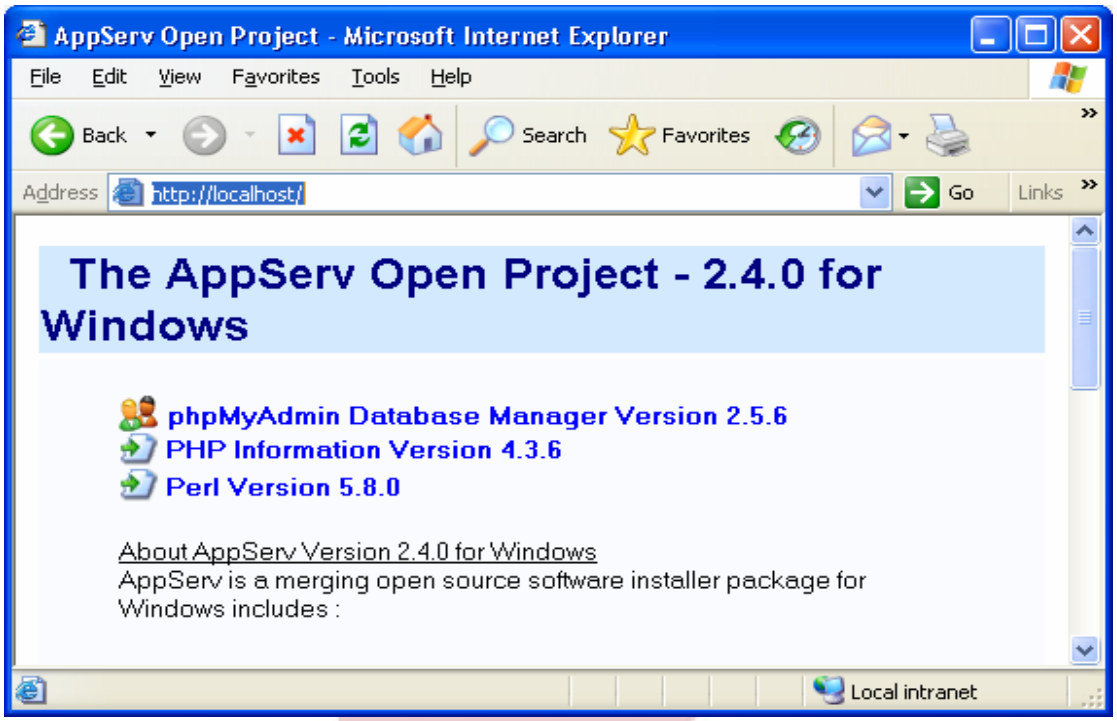
ستظهر لك أيقونة البرنامج على الشكل التالي:

اضغط على أيقونة البرنامج بعد انتهاء التحميل:

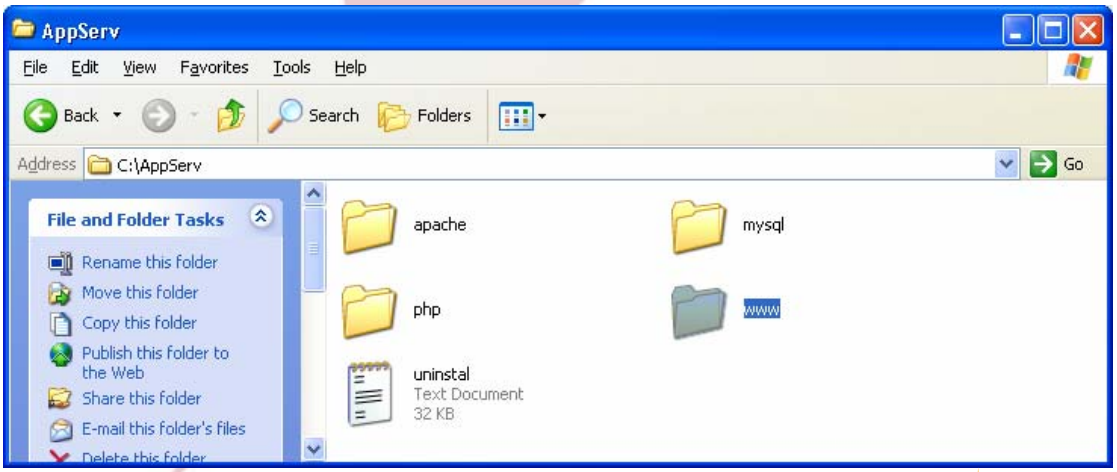


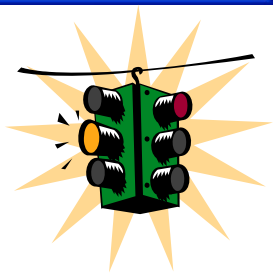
سيبدأ التحميل وأنصحك بعدم تغيير أياً من خيارات معالج التثبيت أي تابع بشكل روتيني إلى نهاية التحميل.

بعد الانتهاء اذهب إلى المتصفح واكتب:



المسار الخاص بملفاتنا من الآن وصاعداً سيكون المجلد www:





سنرى أن ذلك . السيرفر . ضروري جداً في التعامل مع النماذج وربط الصفحات وقواعد البيانات... تذكر ذلك جيداً.... أما بالنسبة لباقي ملفات الـ PHP فيمكنك التعامل مع ما سبق شرحه (PHP 4.4.2).

مدخل إلى الـ PHP

من الضروري أن أكرر إلى أهمية معرفتك عزيزي الطالب بلغات الـ HTML و الـ JAVA قبل دخولك في هذه اللغة، فإذا كنت غير متمكن منهما فأظن أنه حان الوقت لكي تحاول التعرف عليهما بشكل جيد ومن ثم تتابع في تعلم هذه اللغة الممتعة...

هناك وسوم مهمة لبناء ملف PHP علينا معرفتها قبل الدخول في هذه اللغة وهي:

- زوج الوسوم <?php و ?> وهو مشابه لوسم <BODY> في لغة html

```
<?php


?>
```

- زوج المختصر <? و ?> وهو يستخدم بنفس الطريقة السابقة ولكنه يكون بدون الكلمة php في وسم البداية، ويحتاج إلى كمية أقل من الكتابة بالطبع، ولكنه يتعارض مع وسوم xml.
- استخدام زوج الوسوم التالية:

```
<script language="php" >
    echo 'Hi It’s My First Page in PHP’;
</script>
```

ولكن هذه الطريقة غير مستخدمة الآن، حيث أنها تصعب عملية التمييز بين شفرات PHP وباقي ملف HTML، وكذلك بالنسبة لبرامج كتابة ملفات HTML التي تعطي تلوينا للشفرة فأغلبها لا يتعرف على هذا النوع من الشفرة ويعتبره جزء من ملف HTML الاعتيادي.

أفضل الطرق السابقة للتحويل إلى وضعية PHP هو استخدام زوج الوسوم الأول بالطبع، حيث أنه الأكثر استخدامها، ولا يحتوي على أية تعارضات كما أنه يعمل على جميع مترجمات PHP مهما كانت إعداداتها، ولهذا السبب سنستخدمها في جميع الأمثلة التي ستجدها في هذا المقرر.

يتكون كود الـphp من نصوص وكود وعلامات ولغة html وقد لا تحتوي على نصوص html.

لكي يعمل الكود يجب أن يكون امتداد الملف php أو بأي امتداد من إمتدادات الـphp مثلاً php3 و phtml.

عندما نطلب صفحة في الإنترنت يجري اتصالاً مباشراً مع السيرفر هذه العملية تدعي request للسيرفر (يعني طلبية للسيرفر) يقوم السيرفر بتفسير طلبك والبحث عن الصفحة المطلوبة ويرسل إليك الصفحة المطلوبة كجزء مما يسمي response (استجابة) لمستعرض الإنترنت لديك يقوم بعدها المتصفح لديك بأخذ الكود الذي ارجع إليه ويقوم بتجميعه (compile) لكي يصبح صفحة صالحة للعرض هذه العملية التي حصلت تشبه نظرية العميل للخادم(client to server) بحيث أن المتصفح هو العميل والخادم هو السيرفر.

الخادم يقوم بعملية تخزين وترجمة وتوزيع البيانات بينما يقوم العميل (مستعرض الإنترنت لديك) بالعبور إلى السيرفر وإحضار البيانات.

ملاحظة على الكتابة:

يجب أن نكتب في نهاية كل سطر الرمز التالي (;) الفاصلة المنقوطة، وطبعاً يوجد إستثناءات لكل قاعدة، فالمعاملات الشرطية مثل **if** وحلقات التكرار مثل **while** لا نضع لها فاصلة منقوطة في نهاية السطر، أيضاً إن كان السطر طويلاً بحيث لا يمكن مشاهدته على الشاشة بشكل كامل وأردت أن تفرق الأسطر بدون وجود دالة في بداية السطر الثاني فيمكن أن ينتهي السطر بدون (;) ويبدأ السطر الثاني كتكملة للسطر السابق.

المتغيرات في الـ PHP

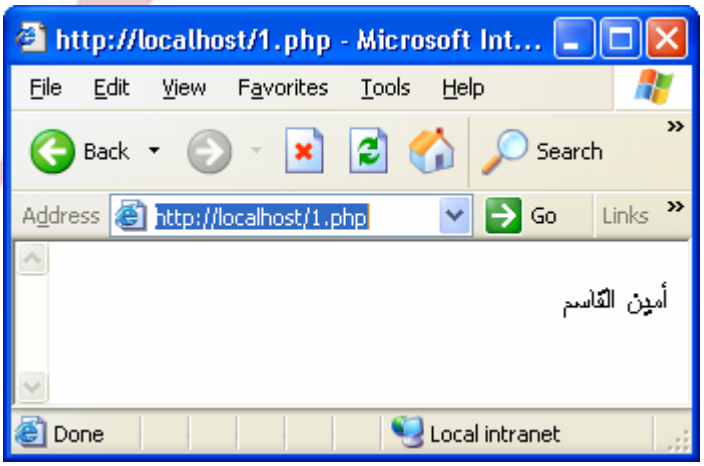
المتغير هو عبارة عن مكان أو محتوى تضع فيه قيمة. أي هو وعاء يمكن أن يحتوي على قيم معينة فهو مكان نحفظ به قيمة (بيانات). إذن كيف نقوم بتعريف متغير؟ وكيف نحدد قيم المتغير ؟! في لغة PHP نقوم بتعريف متغير عن طريق الرمز $ فمثلاً لو قلنا أننا نريد أن نعرف متغيراً اسمه name ويحتوي على القيمة (أمين) فسوف نقوم بالتالي:

```
$name = "أمين";
```

وقد لكن يجب عليك الانتباه بأن أسماء المتغيرات في لغة PHP لها حدود، نعني بذلك انه يوجد أسماء وأحرف غير مسموحه لتعريف المتغيرات، لنقوم بتسهيل الأمر سنذكر المسموح منها وهو (أحرف) و (أرقام) و (_) فقط.

مثال:

```
<html dir="rtl">
<?
$name = "أمين القاسم";
echo $name;
?>
```



هذا في حالة المتغيرات النصية Text، أما في المتغيرات الرقمية Numbers يمكن تعريف متغير (Counter) الذي يحمل القيمة (20) كالتالي:

```
<?
$Counter = 17;
?>
```

وهذه نقطة مهمة وهي لماذا وضعنا علامات التنصيص هذه؟ فالإجابة تكون هي أن القيمة التي وضعناها حرفية أي تتكون من نصوص وهناك أنواع للمتغيرات وعلى ذلك سنفصل ونقول: أن الفرق بين طريقة تعريف المتغيرين النصي والرقمي واضح وهو عدم وجود علامات التنصيص في تعريف المتغيرات الرقمية بينما يجب وضع علامات التنصيص في تعريف المتغيرات النصية.

نقاط هامة في تسمية المتغيرات:

- أسماء المتغيرات في كثير من لغات البرمجة لا تتعدى 255 حرف سواء كانت حروف أو أرقام أو علامات أخرى، وفي لغة الـ PHP لا يوجد حدود على عدد الخانات في تسمية المتغيرات، ولكن في الغالب لن تحتاج إلى اكثر من 15 خانة لتسمية أي متغير، لان المبالغة في تسمية المتغيرات تسبب مشاكل في تذكر المتغيرات وما تحتويه من قيم.
- بداية كل متغير يجب أن تبدأ بحرف (يعني حرف هجائي) أو علامة (_) Underscore، مع تجاهل علامة الـ $ لأنها لا تحسب من اسم المتغير.
- يمكن أن يحتوي اسم المتغير على الحروف أو الأرقام أو علامة (_) فقط، أما العلامات الأخرى مثل (+ , - , * , /) أو الـ & لا يمكن كتابتها في اسم المتغير .
- المتغير ($Name) يختلف عن المتغير ($name) لاختلاف حالة حرف الـ N، ولذلك يجب التأكد من اسم المتغيرات بدقة لتجنب حدوث مشاكل في الوصول إلى متغير معين، وبالتأكيد لو كان لديك أسلوب خاص في تسمية

المتغيرات لسهولة الوصول إليها وتذكرها ستكون كتابة السكربتات أسهل بكثير.

- يستحسن أن تكون أسماء المتغيرات دالة على معانيها، بمعنى أنه لمتغير مثل عداد الزوار يستحسن أن يكون ($counter)، ولمتغير مثل اسم المستخدم ($user) .. الخ .

ABAHE

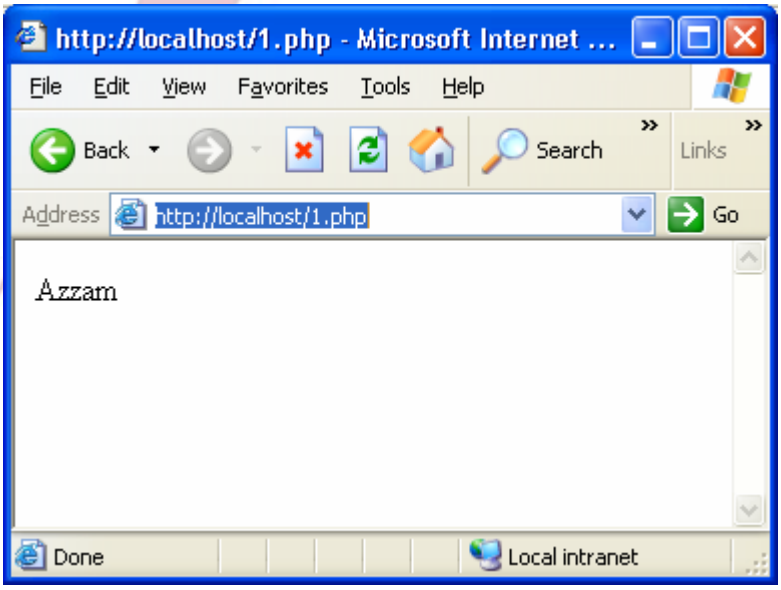
التعامل مع المتغيرات:

فائدة المتغيرات تكمن في طريقة استخدامها في كتابة السكربت، وكما ذكرنا سابقاً أنه لطباعة متغير معين نستخدم أمر الطباعة (echo) أو (print) كما يلي:

```
<?
$name = "Azzam";
echo $name;
?>
```

في البداية سيتم إسناد القيمة (Azzam) إلى المتغير ($name)، وفي السطر الثاني يتم طباعة المتغير، أو بالأحرى القيمة المسندة إلى المتغير.

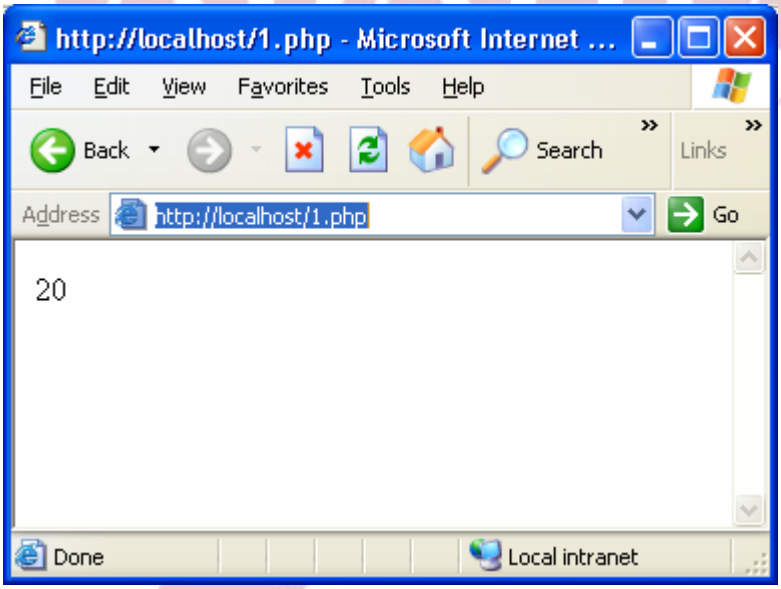
إذا طبقنا السكريبت السابق سنجد الناتج على الشكل التالي:



لاحظ أننا أسمينا الملف 1.php ولاحظ أيضاً المسار الذي كتبناه في المتصفح. ولا ننسى أبداً أننا نقوم بتسجيل ملفاتنا في المجلد: C:\Inetpub\wwwroot.

مثال على القيمة العددية:

```
<?
$Counter = 20;
Print $Counter;
?>
```



أنواع البيانات

في الأمثلة السابقة قمنا بإسناد قيمتين عددية ونصية إلى متغيرين، وبينا الفرق بينهما، وفي لغة الـ PHP بشكل عام يوجد أكثر من هذين النوعين من البيانات ولا يتم تعريف نوعها من قبل المبرمج إنما مترجم الـPHP يقوم بالتعرف عليها لكي يتم إتمام العمليات المختلفة عليها.

لدينا عدة أنواع من المتغيرات وهي: نصوص = string , أرقام = integer أرقام عشريه = Double , المتغير Boolean , مصفوفة = array , كائن = object.

وسوف نتناول الآن الحديث عن البيانات النصية والعددية بشكل عام. أما باقي الأنواع فسنتركه لمراحل متقدمة.

البيانات النصية والعددية:

هي البيانات التي تكون بين علامات التنصيص " " بغض النظر عن محتواها، فيمكن أن تكون حروف أو أعداد أو رموز أو غيرها، ومثال ذلك كما ذكرنا سابقاً:

```
<?
$name = "Ammar";
$number = "5.5";
?>
```

لإضافة المتغيرات التي تحتوي على بيانات نصية مع متغيرات من نفس النوع نحتاج إلى عملية دمج بين المتغيرات، ولعمل ذلك نكتب:

```
<?
$total = $name . $number;
?>
```

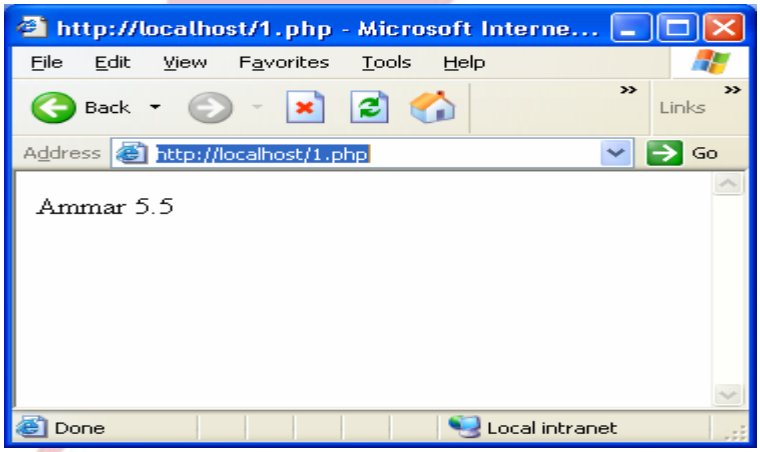
في هذه الحالة سيتم إسناد القيم السابقة إلى المتغير total$ وإذا أردنا وضع مسافة بين المتغيرين نضيف متغير جديد يحتوي على المسافة وهو ($space) ثم نقوم بعملية الدمج بين المتغيرات كالتالي:

```
<?
$space = " ";
$total = $name . $space . $number;
?>
```

وتصبح الصيغة العامة على الشكل التالي:

```
<?
$name = "Ammar";
$number = "5.5";
$space = " ";
$total = $name . $space . $number;
Print $total;
?>
```

والناتج:



يمكننا أيضاً وضع المسافة بين الكلمات بالطريقة التالية:

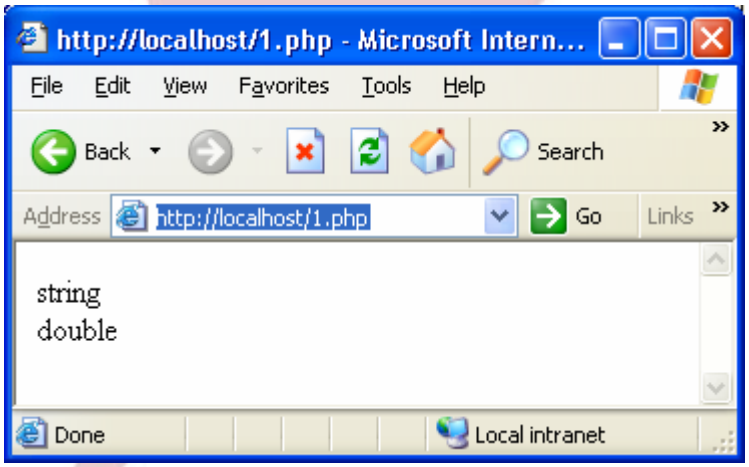
```
<?
$name = "Ammar";
$number = "5.5";
$total = $name . " " . $number;
Echo $total;
?>
```

تسمى لغة PHP اللغة المسامحة ولذلك عندما نضع القيمة داخل المتغير لا نحتاج أن تقوم بتحديد النوع، فالمترجم يقوم بعرفة النوع من تلقاء نفسه، طالما وضعنا علامة $ قبل اسم المتغير.

والأمر بسيط جداً وهو أن تقوم باستخدام الدالة gettype($var); وسوف تقوم هذه الدالة بطباعة نوع المتغير الموجود لديك. ولنأخذ مثال على ذلك:

```
<?
$name = "Ammar";
$number = 5.5;
Print gettype ($name);
print "<br>";
print gettype ($number);
?>
```

وسوف يكون الناتج كما يلي:



والبيانات العددية لها نوعين (الأعداد الصحيحة Integer) و(الأعداد الكسرية Double)، وكمثال على النوعين:

```
<?
$integer1 = 233;
$integer2 = -29;
$double1 = 5.27;
$double2 = -4.6;
?>
```

المعامــلات

التعامل مع البيانات العددية (Numeric):

لدينا ثلاث أنواع من المعاملات في لغة PHP وهي المعاملات الحسابية، معاملات المقارنة، المعاملات المنطقية وسوف نتكلم عنها بالتفصيل.

المعاملات الحسابية:

المعاملات الحسابية الاعتيادية:

المعامل	الاسم	مثال	نتيجة المثال
+	جمع	1+1	2
-	طرح	1-1	0
*	ضرب	2*3	6
/	قسمه	6/2	3
%	باقي القسمه	10%3	1

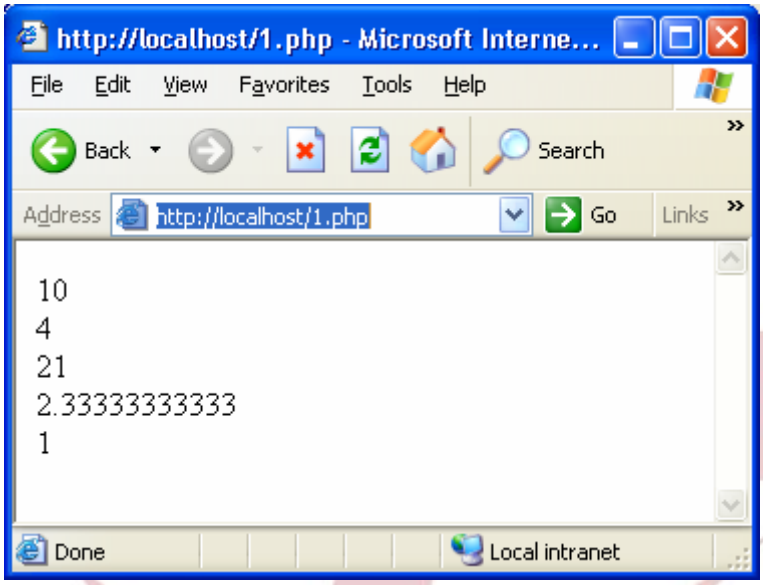
معاملات حسابية إضافية:

المعامل	مثال	تساوي
+=	$x +=5	$x = $x + 5
-=	$x -=5	$x = $x - 5
*=	$x *= 5	$x = $x * 5
/=	$x /= 5	$x = $x / 5
%=	$x % = 5	$x = $x % 5
++	$x++	$x = $x + 1
--	$x--	$x = $x - 1

لنقم بكتابة بعض الأكواد وسوف ترى كيفية سهولة عمل بعض من تمارين العمليات الحسابية الاعتيادية:

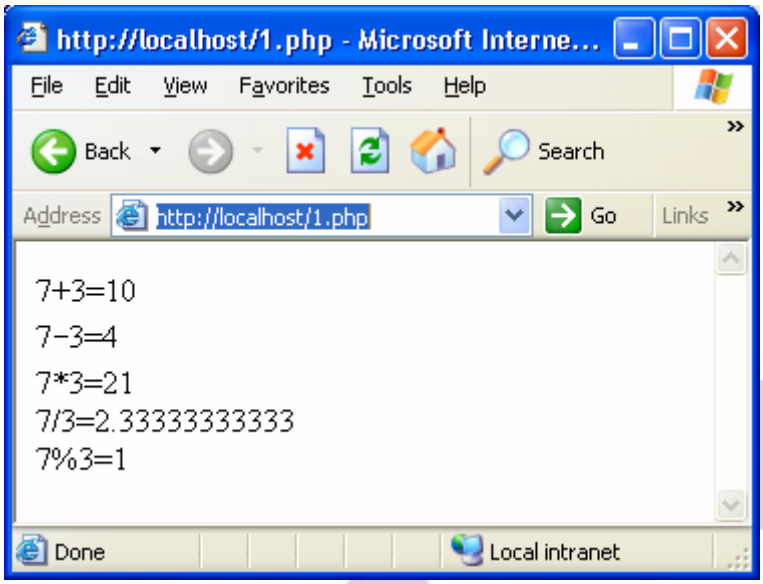
```
<?
$x = 7;
$y = 3;
//العمليات
$a = $x + $y;//عملية الجمع
$b = $x - $y;//عملية الطرح
$c = $x * $y;//عملية الضرب
$d = $x / $y;//عملية القسمة
$e = $x % $y;//عملية القسمة مع باقي
Print $a;
print "<br>";
Print $b;
print "<br>";
Print $c;
print "<br>";
Print $d;
print "<br>";
Print $e;
print "<br>";
?>
```

فتظهر لنا النتائج على النحو التالي:



وإذا أردنا رؤية الناتج كعملية حسابية:

```
<?
$x = 7;
$y = 3;
//العمليات
$a = $x + $y;//عملية الجمع
$b = $x - $y;//عملية الطرح
$c = $x * $y;//عملية الضرب
$d = $x / $y;//عملية القسمة
$e = $x % $y;//عملية القسمة مع باقي
Print $x."+". $y. "=" .$a;
print "<br>";
Print $x."-". $y. "=".$b;
print "<br>";
Print $x."*". $y. "=".$c;
print "<br>";
Print $x."/". $y. "=".$d;
print "<br>";
Print $x."%". $y. "=".$e;
?>
```



أصبحت الفكرة واضحة الآن بالنسبة للعمليات العادية أما العمليات الإضافية فسيتم التعامل معها في سياق المادة وفي الموضع المناسب لها إلا أنني سأدرج مثال لفهم المغزى منها.

أول النقاط هي إضافة المتغير إلى نفسه، بمعنى تعريف عملية حسابية على متغير معين بحيث تخزن القيمة في نفس المتغير، مثلاً لو كان لديك عدد الزوار وتريد في كل مرة أن يزود عدد الزوار بـ 1، يمكنك كتابة ما يلي:

```
<?
$counter = $counter + 1;
?>
```

بالتالي سيتم زيادة المتغير (counter$) بـ 1 في كل مرة يتم فيها تنفيذ السكريبت، وبطريقة أخرى يمكن كتابة السطر السابق كالتالي:

```
<?
$counter = $counter++;
?>
```

والـ ++ تعنـي زيـادة قدرها (1) علـى قيمـة المتغيـر الأصـلية، وكذلك الـ -- تعنـي طرح 1 من القيمة الأصلية .

وفي حالة الرغبة بزيادة أي عدد أخر (غير الواحد) على أي متغير بأسلوب الطريقة الثانية يمكن كتابة ما يلي:

```
<?
$counter +=4;
?>
```

وهـذا يعنـي زيـادة مقـدارها 4 علـى قيمـة المتغيـر الأصـلية، وبالـسالب كـذلك بـنفس الأسلوب.

معاملات المقارنة:

المعامل	المعنى
= =	يساوي
=!	لا يساوي
= = =	يساوي ومن نفس النوع
>	اكبر من
>=	اكبر من ويساوي
<	اصغر من
<=	اصغر من ويساوي

المعاملات المنطقية:

المعامل	الاسم	يكون صحيح عند	مثال	النتيجة
||	أو	أحد الطرفين صحيح	true || false	true
OR	أو	أحد الطرفين صحيح	true OR false	true
&&	و	كل الطرفين صحيحه	true && false	false
AND	و	كل الطرفين صحيحه	true AND false	false
xor	xor	أحد الطرفين صحيح وليس الاثنان	true xor true	false
!	لا	الطرف ليس صحيح	!true	false

عند استخدام المعاملات يقوم المترجم بالقراء من اليمين إلى اليسار في الحالات العادية وعند استخدام معامل واحد، ولكن تخيل لدينا اكثر من معامل في سطر برمجي واحد؟ هنا يقوم المترجم بالاختيار حسب الأهمية، لنأخذ مثال على ذلك، تخيل أن لديك العملية التالية:

4 + 8 فما هو الناتج ؟؟؟ الناتج واضح وليس به أي غموض وهو 12 ولكن تخيل لو لديك العملية التالية 4 + 8 * 2 فما هو ناتج العملية الحسابية؟ الجواب هو 20 لأن المترجم قرأ عملية الضرب أولاً ثم عملية الجمع ويمكن أن نقوم بإجبار المترجم على قراءة عملية الجمع أولا باستخدام الأقواس.

بمعنى أخر يجب أن نفرق بين العمليات الحسابية البسيطة والمعقدة وطريقة التعامل مع كلٍ منها للوصول إلى النتائج السليمة.

في الجدول التالي سوف تقوم بمعرفة الترتيب حسب الأهمية وما هي المعاملات التي تنفذ قبل الأخرى بشكل تلقائي (من الأعلى إلى الأسفل)

المعامل
++ , -- , (cast)
/ , * , %
+ , -
< , <= , => , >
= = , = = = , !=
&&
||
= , += , -= , /= , *= , %= , .=
AND
xor
OR

الثوابت

يمكننا القول أن الثوابت هي متغيرات ولا تتغير أبداً مهما حصل، ونحتاج إلى هذه النوع من المتغيرات في أمور عديدة على سبيل المثال لا الحصر: تخيل أن لدينا أسعار بعض من منتجاتنا، وهذه الأسعار ثابتة ولا تتغير فسوف نقوم بوضعها في الثوابت بالطريقة التالية define ('var' , value).

فالفرق بين المتغيرات والثوابت انه عندما نريد أن نقوم بطباعة الثابت أو استخدامه لا نقوم بوضع علامة $ في الثابت أبداً، وأيضاً من الأسماء الممنوع استخدامها في متغيرات لغة PHP الثوابت المعرفة في نفس اللغة، وهي ثابتة ولا تتغير وللتعرف على هذه الثوابت اكتب الصيغة التالية ومن ثم استعرض الملف الذي سيظهر على المتصفح... ستجد معلومات كثيرة بداخله وما يهمنا هنا هو القسم المتعلق بـ Variables فهو مهم جداً وسيتم سؤالك عن بعض المعلومات فيه:

```
<?
phpinfo();
?>
```

بعض الدوال الهامة:

بعض الدوال الهامة في التعامل مع المتغيرات:

isset: وهي دالة للتأكد من وجود متغير معين، فمثلاً:

```
<?
echo isset($age);
?>
```

سيتم طباعة الرقم 1 إذا كان المتغير ($age) موجوداً أي تم إنشاءه مسبقاً، والعكس إذا كان غير موجود سيتم طباعة الرقم 0، وهذه الدالة يتم استخدامها كثيراً في الشروط.

unset: هذه الدالة تعمل على مسح المتغير من الذاكرة كلياً، فقط قم بعمل التالي:

```
<?
unset($age);
?>
```

وفي هذه الحالة سيتم مسح المتغير ($age) بشكل كامل.

empty: وهذه الدالة معاكسة للدالة isset بحيث لو كتبنا:

```
<?
echo empty($age);
?>
```

سيتم طباعة الرقم 1 في حالة عدم وجود المتغير ($age) أو أن قيمة المتغير تساوي 0 أو (فراغ)، وفي حالة وجود المتغير ($age) لن يتم طباعة أي شيء.

دوال الوقت والتاريخ في الـ PHP

الوقت والتاريخ من الأشياء المهمة جداً لبناء ويب متكامل، فالمستخدم يريد مثلاً معرفة تاريخ اليوم أو تاريخ نشر المقال أو الخبر أو.... فصاحب الويب يريد معرفة تاريخ تسجيل مستخدم معين أو تاريخ دخوله وتاريخ إرساله رسالة إلى الموقع.

كل تلك الأشياء تجعل التاريخ مهم جداً لنا في بناء الويب. والتاريخ والوقت في لغة PHP سهل جداً. مجرد دوال نقوم باستخدامها بالطريقة الصحيحة ونحصل على الناتج المراد.

لكي نقوم بطباعة التاريخ والوقت وإيجاده وتنسيقه نستخدم الدالة date() وهذه الدالة عملها سهل جداً.

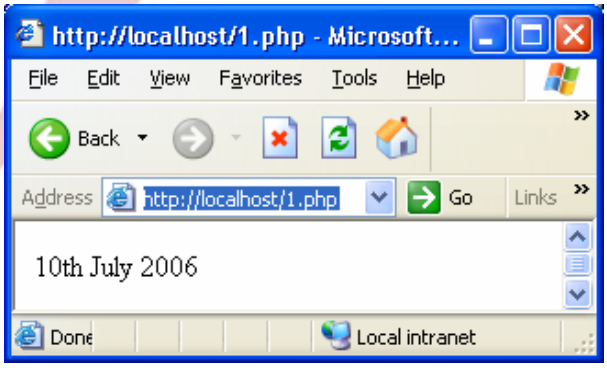
نأخذ قيمتين واحدة أساسية والأخرى افتراضية ولاحظ التالي:

```
date(string format);
```

فلو أردنا طباعة تاريخ اليوم سنقوم بالتالي:

```
<?
echo date('jS F Y');
?>
```

وسوف يكون الناتج هو التالي:



حلل الصيغة السابقة من خلال الرموز المستخدمة لإنشاء التاريخ والوقت.

وهذه الرموز هي:

الرمز	المعنى	مثال
a	الوقت صباحاً أو مساءً	am , pm

A	الوقت صباحاً أو مساءً	AM , PM
d	اليوم في الشهر على شكل أرقام من خانتين الخانة الأولى صفر	من 01 إلى 31
D	اليوم في الشهر ثلاث خانات مختصر على شكل حروف	من mon إلى sun
F	الشهر في السنة مكتوب كاملاً	September
g	الساعة في اليوم على مدى 12 ساعة	من 1 إلى 12
G	الساعة في اليوم على مدى 24 ساعة	من 0 إلى 23
h	الساعة في اليوم على مدى 12 ساعة	من 01 إلى 12
H	الساعة في اليوم على مدى 24 ساعة	من 00 إلى 23
i	الدقائق في الساعة	من 00 إلى 59
I	نهار أو مساء	نهار 1 , مساء 0
j	اليوم في الشهر على شكل أرقام من غير صفر	من 1 إلى 31
l	اليوم في الشهر , مكتوب كاملا على شكل حروف	Monday
L	السنة الكبيسة	كبيسة 1 , غير كبيسة 0
m	الشهر في السنة على شكل خانتان بدأ من الصفر	من 01 إلى 12
M	الشهر في السنة على شكل ثلاث خانات حروف	Jan
n	الشهر في السنة على شكل خانتان لا يبدأ بالصفر	من 1 إلى 12
s	الثواني في الدقيقة على شكل خانتان بدأ من الصفر	من 00 إلى 59
S	الاختصارات للأحرف	TH , ST , ND
t	مجموع الأيام في الشهر	من 28 إلى 31
U	مجموع عدد الثواني من تاريخ 1970 المسمى بي UNIX TIME STAMP	
w	اليوم في الأسبوع على شكل أرقام	Sunday => 0 , Saturday => 6
y	السنة على شكل رقم من خانتان	99 , 98 , 05
Y	السنة على شكل رقم من أربع خانات	2000 , 2005
z	اليوم في السنة على شكل أرقام	من 0 إلى 365

الدالة getdate() وكيفية استخدامها:

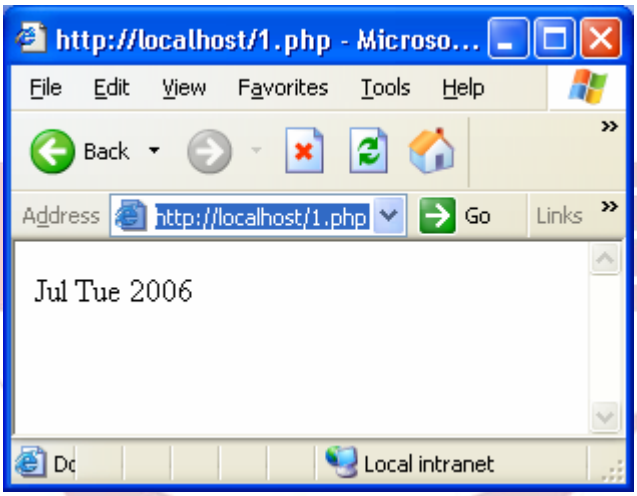
تستخدم هذه الدالة بإعطائها التاريخ على شكل UNIX TIME STAMP وتقوم بإرجاعه على شكل عادي، هذه الدالة تقوم بإرجاع التاريخ على شكل مصفوفة حرفيه.

المفتاح لهذه المصفوفة في الجدول التالي:

key	value
seconds	الثواني . أرقام
minutes	الدقائق . أرقام
hours	الساعات . أرقام
mday	اليوم في الشهر . أرقام
wday	اليوم في الأسبوع . أرقام
mon	الشهر . أرقام
year	السنة . أرقام
yday	اليوم في السنة . أرقام
weekday	اليوم في الأسبوع . على شكل نص كامل
month	الشهر . على شكل نص كامل

مثال

```
<?
Echo gmdate ("M D Y");
?>
```

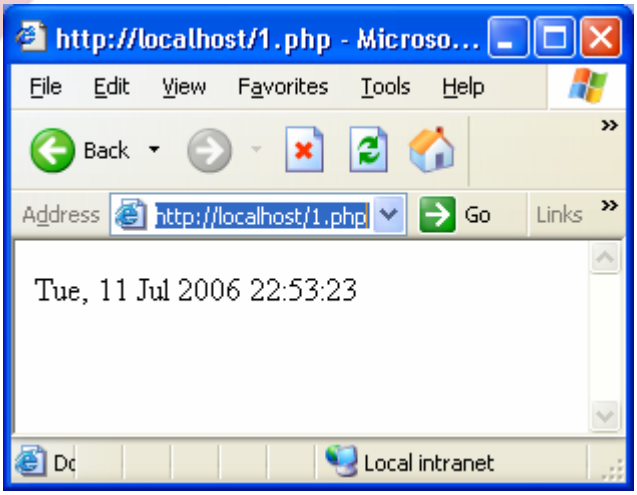


لاحظ أننا استخدمنا علامات التنصيص لكي تنجح العملية عندما قمنا باستخدام أكثر من عامل في الدالة

جرب استخدام الكود التالي:

هذا سوف يعرض لك اليوم والتاريخ والساعة

```
<?
Echo gmdate ("D, d M Y H:i:s")
?>
```



العبارات الشرطية

سبق وأن مرت هذه العبارة في العديد من مقرراتنا الدراسية ولكني سأعيد صياغة مفهومها بطريقة أرجو أن تكون مختلفة هذه المرة. تستعمل عبارة IF للتميز بين ناتجين لعملية ما فإذا توافقت القيمة مع الشرط أدى ذلك إلى تنفيذ أمر ما، أما في حال عدم التوافق أو التطابق أو المساواة . سمها ما شئت . سيتم تنفيذ أمر أخر أي إذا تحقق الشرط المعين افعل كذا وكذا، وإذا لم يتحقق افعل كذا وكذا. وبشكل عملي نمثلها كالتالي:

العبارة الشرطية IF:

```
IF condition is true (إذا كان الشرط صحيحاً)
{
excute this code (قم بتنفيذ هذا الكود)
}
```

سيقوم الـPHP بتنفيذ الكود الذي بين { و } فقط في حال كان الشرط صحيحاً.

أما إذا لم يكن صحيحاً فسيقوم بتجاوزه وتنفيذ الكود الذي يليه.

ويمكنك أيضا أن تقوم بجعلها بسطر واحد ولا تستخدم الأقواس بل تكتب الأمر مباشرة:

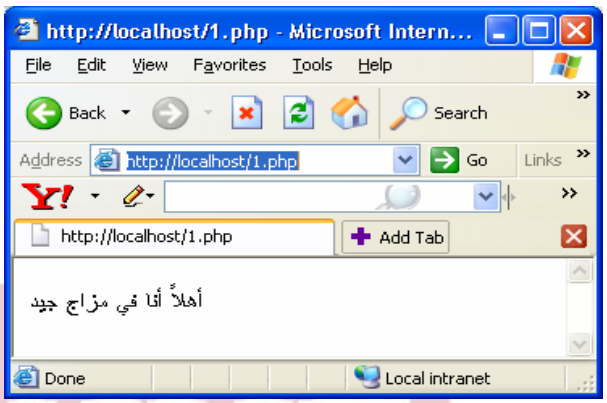
IF condition is true excute function;

لاحظ أنه لابد من استخدام { و } إذا كان الكود يتكون من عدة أسطر أما إذا كان يتكون من سطر واحد فلا داعي لاستخدامها .

سوف نوضح الصورة مع الأمثلة:

مثال(1)

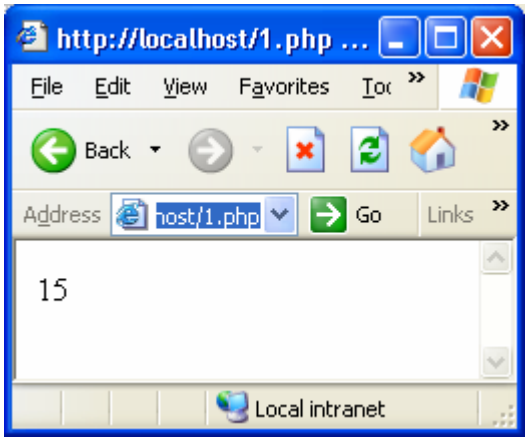
```
<?
$mood="سعيد";
if($mood=="سعيد")
{
print"أهلاً أنا في مزاج جيد";
}
?>
```



لاحظ أننا استخدمنا عامل المقارنة (==) الذي تكلمنا عنه سابقاً، هذا معناه هل الطرف الأيمن يساوي الطرف الأيسر ؟ لا تنسى انه يختلف تماماً عن العلامه (=) لوحدها، فالعلامة (=) معناها تعيين أو تعبئة متغير بقيمة معينة ولكن (==) يستخدم لمقارنة الأطراف. في حال عدم المطابقة لن يظهر شيء على الشاشة.

مثال(2)

```
<?
$S=20;
if ($S=20) echo 15;
?>
```



لاحظ أنه يمكننا استعمال echo أو Print

العبارة الشرطية ELSE:

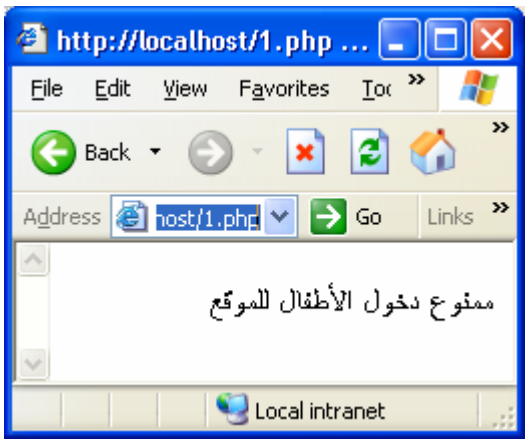
في IF الشرطية نستطيع فقط التحكم في شيء واحد أما العبارة ELSE وتعني إذا تحقق هذا الشرط قم بعمل التالي... وإذا لم يتحقق قم بعمل التالي... وتتم كتابته بنفس الطريقة الأولى مع إضافة بعض الأسطر البرمجية لاحظ طريقة الكتابة التالية:

```
If condtion is true
{
Excute code
}
Else
{
Excute other code
 }
```

وهي تقوم بالتحقق من الشرط فإذا وجدته صحيحاً قامت بتنفيذ الكود الأول وإذا لم تجده صحيحاً ستقوم بتنفيذ الكود الآخر.

سنأخذ المثال البسيط التالي لتوضيح الفكرة:

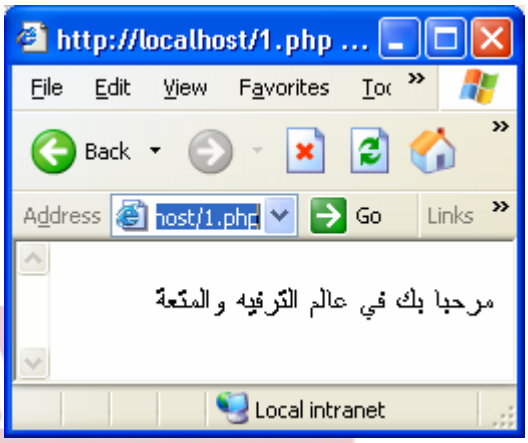
```
<?
$age=12;
If ($age>18)
 {
echo "مرحبا بك في عالم الترفيه والمتعة";
}
else
{
echo "ممنوع دخول الأطفال للموقع";
}
?>
```



لاحظ بأن العمر كان أصغر من 18.

ولكن إذا كان العمر أكبر من 18.

```
<?
$age=22;
If ($age>18)
 {
echo "مرحبا بك في عالم الترفيه والمتعة";
}
else
{
echo "ممنوع دخول الأطفال للموقع";
}
?>
```



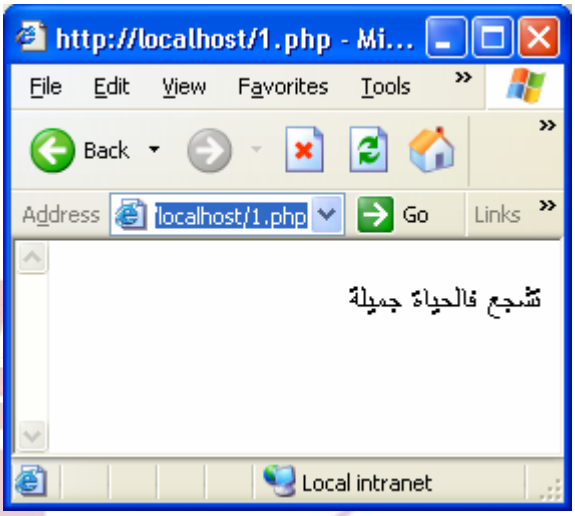
ويمكننا أيضا استخدام الهيكلية التالية:

```
If condtion is true
{
Excute code
}
Elseif
{
Excute other code
 }
Else
{
Excute other code
 }
```

وهي تقوم بتطبيق أكثر من شرط فإذا لم يكن أي شرط من الشروط صحيحاً سيتم تنفيذ الكود الذي يقع بعد كلمه else .

مثال:

```
<html dir="rtl">
<?
$mood="حزين";
if($mood=="سعيد")
{
print"أهلاً أنا في مزاج جيد";
}
elseif ($mood=="حزين")
{
print"تشجع فالحياة جميلة";
}
else
{
print"لست سعيد ولا حزين ولكنك $mood";
}
?>
```

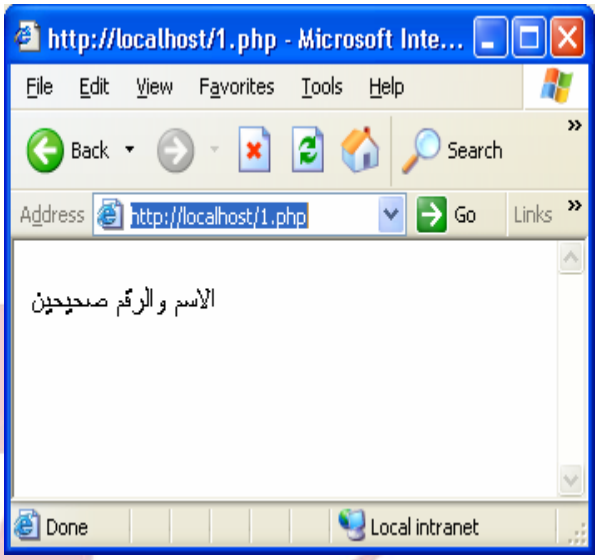


غير في القيمة $mood="حزين";

ولاحظ الفرق في الناتج.

مثال:

```
<?
$h="Azzam";
$f=64;
If  ($h == "Azzam")
{ If ($f== 64)
{
echo "الاسم والرقم صحيحين";
}
else
{
echo ("الرقم غير صحيح");
}     }
else  {
echo "اسم تسجيل الدخول غير صحيح" ;
}
?>
```



حاول تغيير القيم الخاصة بالاسم والرقم وشاهد النتائج.

هذا مجرد مثال بسيط جداً لتداخل الدوال الشرطية حيث يقوم بإجراء اختبار على قيمة معينة ثم يقوم عند تجاوزه ذلك الاختبار بنجاح بإجراء اختبار ثاني فإذا تم تجاوز الاختبار الثاني يتم طباعة الاسم والرقم صحيحين وإذا لم يتم الاجتياز يتم طباعة عبارة الفشل في الاجتياز.

عبـارات التكرار

التكرارات عبارة عن تكرار أمر معين بعدد معين من المرات ولقد أخذنا سابقاً العبارات الشرطية فوجدنا أن الكود الذي نكتبه في العبارات الشرطية لا تنفذ إلا عندما يكون الشرط صحيحاً وأيضا التكرارت فهي تختبر الشرط فإذا كانت قيمته صحيحة فإنها تقوم بعمل الكود المطلوب ثم تقوم بإعادة اختبار القيمة فإذا كان صحيحاً فإنها تقوم بإعادة تنفيذ الكود وهكذا، أما عندما لا يكون الشرط صحيحاً فإنها تتوقف عن تنفيذ الكود ويتم إكمال البرنامج بشكل عادي... هناك ثلاثة أنواع من التكرارات هي:

التكرار while

التكرار do - while

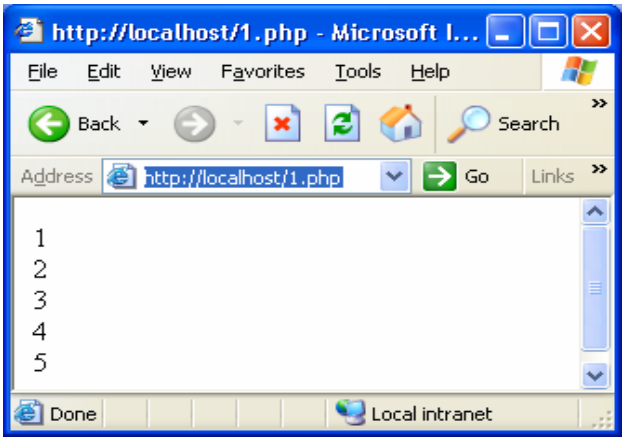
التكرار FOR

إن أول دالة نقوم بأخذها في البداية هي الدالة **while**

لنأخذ بعض الأمثلة على طريقة كتابة عبارة التكرار WHILE تخيل انك تريد طباعة الأرقام من 1 إلى 5 كم سطر سيتطلب منك كتابته؟ وماذا لو كنا نريد تكرار 100 سطر...

هنا تأتي فائدة عبارة التكرار WHILE لكي تساعدنا على تسهيل كتابة الكود:

```
<?
$x=1;
while ($x<=5)
{
echo ($x);
echo ("<br>");
$x++;
}
?>
```



كم سطراً قمنا بكتابته الآن؟ لنحاول أن نقوم بكتابة الأعداد من 1 إلى 100 من غير استخدام التكرار، كم سطر سنحتاج...!! ولكن مع التكرار يمكن أن نقوم بذلك وسوف يأتيك الناتج بنفس عدد الأسطر للمثال السابق.

```
<?
```

```
$x=1;
while ($x<=100)
{
echo ($x);
echo ("<br>");
$x++;
}
?>
```

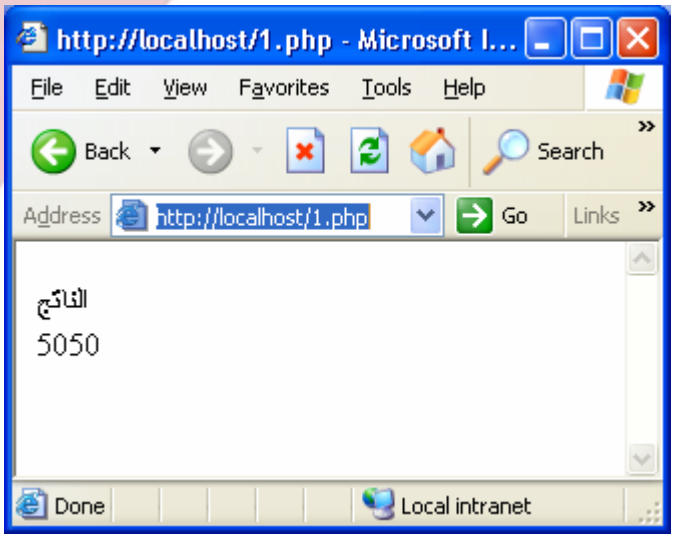
سنشرح الكود الآن:

أولاً: قمنا بتعيين متغير توجد به القيمة 1 واسمه x وبعد ذلك نقوم بكتابة دالة التكرار وشرطها أن يكون المتغير x أقل من أو يساوي 100 ثم نقوم بطباعة المتغير الموجود لدينا ونطبع سطر جديد للترتيب وأخيراً نقوم بزيادة المتغير بواحد ونرجع إلى الأعلى لنرى هل الشرط صحيح أم لا إذا كان صحيحاً يقوم بفعل الطباعة مره أخرى وإذا لم يكن صحيحاً يقوم بالخروج من البرنامج.

مثال:

ما هو ناتج جمع الأعداد من 1 إلى 100 ؟ الجواب هو الكود التالي:

```
<?
$x=1;
$total = 0;
while ($x<=100)
{
$total = $total + $x;
$x++;
}
echo "الناتج";
echo ("<br>");
echo $total;
?>
```



أرجو أن تقوم بشرح الصيغة السابقة.

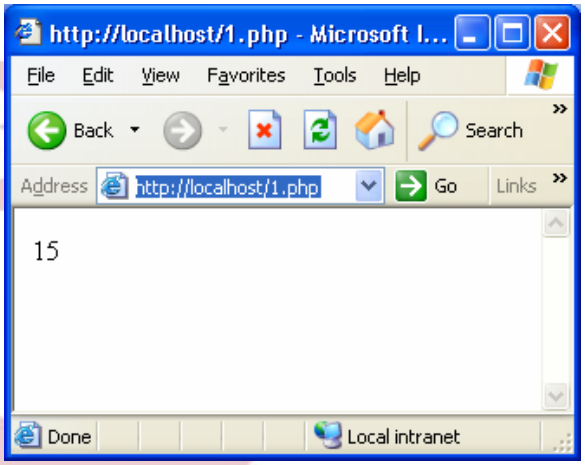
التكرار do - while

هذا التكرار يعمل بنفس طريقه التكرار الأول إلا أنه يوجد بعض الاختلافات البسيطة وصيغته كالتالي:

```
Do
{
        do this code;
}
while (expression);
```

مثال:

```
<?
$f=15 ;
do
{
echo $f;
$f++;
}
while ($f < 10) ;
?>
```



سيقوم التكرار بتنفيذ السطر الموجود بين القوسين أولا ثم يقوم بتنفيذ باختبار الشرط فإذا كان الشرط صحيحاً قام بإعادة العملية الموجودة بين القوسين وهي إضافة واحد على المتغير $f وهكذا حتى يكون الشرط غير صحيح فيتم التوقف.. لاحظ أننا في التكرار الأول قمنا باختبار الشرط قبل صناعة أي عمل بينما في التكرار الثاني قمنا بتنفيذ الكود أولا ثم قمنا بإجراء الاختبار.

التكرار FOR:

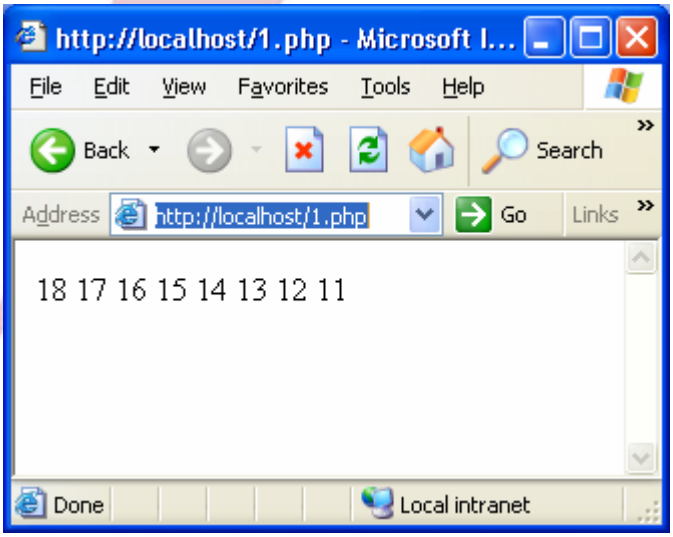
يختلف هذا التكرار عن سابقيه لكن وظيفته هي نفس وظيفتهما وهي تكرار الأوامر عند حصول شي معين.

الصيغة:

أداء عمليه على set counter ; اختبار القيمة test value ; عداد counter For (
العداد)
{
code شفرة
}

مثال:

```
<?
For ($u = 18 ; $u>10 ; $u--)
{
echo $u."\t\t";
}
 ?>
```



يتكون هذا التكرار من ثلاثة أقسام .... القسم الأول نضع فيه متغير يحتوي على قيمة حيث سيبدأ التكرار العمل من عند هذه القيمة والقسم الثاني نكتب فيه

الشرط الذي سيقوم التكرار بفحصه (والذي هو كالمعتاد اختبار لقيمة المتغير في القسم الأول) والقسم الثالث نضع فيه العمل الذي سيجري على المتغير عند كل تكرار ثم نقوم بكتابة كود التي سيقوم بتنفيذها التكرار بين القوسين.

كأننا نقول للـ php بشكل عامي أن يقوم في البداية بإعطاء المتغير $u القيمة 18 وقبل أن يقوم بتنفيذ الكود عليه أن يقوم بتحليل الشرط فإذا كان الشرط صحيحاً فإنه يقوم بإنقاص واحد من المتغير $u ويتم تنفيذ الكود حتى يصبح المتغير $u قيمته 9 فيقوم الـPHP آنذاك بالخروج من التكرار والذهاب إلى الكود الذي يلي القوسين.

الصور في لغة الـ PHP

استخدام لغة php لا يقتصر على إصدار ملفات html أو php بل يسمح لك أيضاً بإنشاء الصور بأنواعها سواء كانت متحركة أو ثابتة.

توجد مكتبة خاصة لعمل هذه الصور وتسمى مكتبة GD والاسم الكامل لها هو (GD Library). ولها إصدارات كثيرة وهي متوافقة مع الأنواع JPEG و GIF و PNG. وكلها تعمل بنفس الطريقة.

:JPEG

هي اختصار لـ (Joint Photographic Experts Group). ونستخدمها كثيراً إذا أردنا حفظ الصور وهي مفيدة إذا كانت في الصورة ألوان كثيرة وتدرجات. هذه النوعية ليست جيده عند استخدام رسوم الخطوط والنصوص والأجزاء التي تحتوي على لون جامد واحد.

:PNG

وهي اختصار لـ (Portable Network Graphics) وخصائص هذه النوعية متماثلة مع النوعية GIF وهي تعتبر بديلاً للنوعية GIF. وسبب هذا البديل هو أن PNG أكثر جودة من النوع GIF.

:GIF

وهي اختصار لـ (Graphics Interchange Format) وتستخدم في مواقع الانترنيت لأنها خفيفة وتستطيع عمل الصور المتحركة من خلالها.

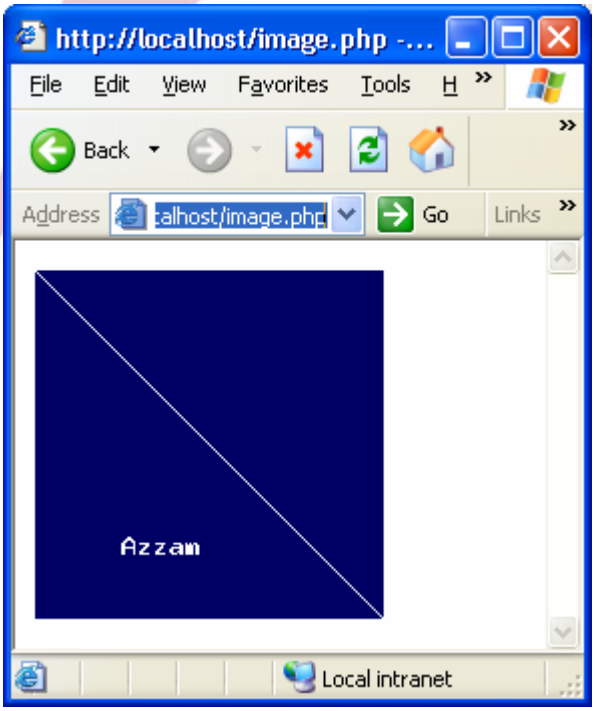
لإنشاء الصور في لغة PHP نتبع 4 خطوات وهي مفيدة جداً وتسهل علينا العمل

1. إنشاء الخلفية المراد العمل عليها
2. رسم المجسمات أو الكتابات على هذه الخلفية
3. تصدير هذه الصور أو العمل
4. تنظيف العمل

لنقم بعمل مثال بسيط على هذه الطريقة لنفهم كيفية العمل:

```
<?
$height = 200;
$width = 200;
$image = ImageCreate ($width , $height);
$white = ImageColorAllocate ($image , 255,255,255);
$black = ImageColorAllocate ($image , 0 , 0 , 100);
@imageFill ($image , 0 , 0 , $black);
@imageLine ($image , 0 , 0 , $width, $height , $white);
@imageString ($image , 8 , 50 , 150, 'Azzam',  $white);
header ( 'content-type: image/png' );
@imagepng ($image);
?>
```

بعد تنفيذ هذا العمل سوف يكون الناتج هو التالي:



أول عمل قمنا بفعله هو إنشاء صوره لنقم بالرسم عليها، وهناك طريقتين لعمل هذه الطريقة الأولى باستخدام الدالة ImageCreate وهي تأخذ مدخلان العرض والطول بالترتيب وسوف تقوم بإرجاع الصورة التي قمنا بعملها

الطريقة الثانية هي إنشاء خلفية العمل من صوره موجودة مسبقاً لدينا بأي نوع كانت باستخدام الدالة ImageCreateFromPng أو ImageCreateFromJpeg أو ImageCreateFromGif وهي تأخذ مدخل واحد وهو اسم الصورة التي نريد العمل عليها.

بعد ذلك نرسم الأشكال التي نريدها على الصورة، أولاً يجب أن نقوم باختيار الألوان التي نريدها وهي مكونه من 3 ألوان وهي الأحمر والأخضر والأزرق، ويمكن استخدام هذه الألوان عن طريق الدالة ImageColorAllocate وهي تأخذ 4 مدخلات وهي الصورة واللون الأحمر واللون الأخضر والأزرق وقد عملنا في وهذه الدالة تقوم بإرجاع اللون الذي سنستخدمه لاحقا.

```
$white = ImageColorAllocate ($image , 255,255,255);
$black = ImageColorAllocate ($image , 0 , 0 , 100);
```

الطريقة الثانية هو رسم ما نريد ويكون على أربع نقاط نحتاج إليها: الصورة، الإحداثيات، اللون، معلومات النص. ويمكننا مشاهدة عملنا في مثالنا هنا:

```
@imageFill ($image , 0 , 0 , $black);
@imageLine ($image , 0 , 0 , $width, $height , $white);
@imageString ($image , 8 , 50 , 150, 'Azzam',  $white);
```

الدالة ImageFIll تقوم بتعبئة الصورة وتأخذ 4 مدخلات وهي:

الصورة والإحداثي السيني والإحداثي الصادي واللون.

الدالة ImageLine تقوم برسم خط وتأخذ 6 مدخلات وهي:

الصورة وبداية الإحداثي السيني وبداية الإحداثي الصادي ونهاية الإحداثي السيني ونهاية الإحداثي الصادي واللون.

الدالة ImageString تقوم بكتابة نص على الصورة وتأخذ 6 مدخلات: الصورة ونوع الخط والإحداثي السيني والإحداثي الصادي والنص واللون.

ثم قمنا بإصدار الصورة ويكون العمل على خطوتين: الخطوة الأولى يجب أن نخبر المتصفح أننا نريد تصدير صور بدلاً من مجرد نصوص أو HTML ويمكننا عمل ذلك عن طريق الدالة Header.

```
header ( 'content-type: image/png' );
```

الخطوة الثانية نقوم بإخراج الصورة إلى المتصفح باستخدامنا للدالة ImagePng أو ImageJpeg أو ImageGif وقمنا بعمل ذلك في ملفنا بالطريقة التالية:

```
@imagepng ($image);
```

التعامل مع النمـاذج

النماذج في الويب أو صفحات الإنترنت عبارة عن استمارات تقوم بتعبئتها ثم عند إرسالها لخادم الويب (السيرفر) يتلقاها برنامج يقوم بإجراء العمليات عليها مثل JavaScript أو ASP أو php .

فعندما تقوم بإنشاء بريد إلكتروني جديد يتوجب عليك أن تقوم بتعبئة نموذج التسجيل والذي يتضمن الاسم وكلمة المرور والبريد المراد إنشائه كل هذا نقوم عمله عن طريق النماذج FORMS .

يتم تخزين هذه القيم في المتغيرات التي يتم كتابتها في الخاصية(name) ويتم إرسالها عند ضغط زر . إرسال البيانات . (submit) إلى الصفحة التي سوف تقوم بمعالجة هذه البيانات والتي يتم تحديدها في الخاصية ACTION وإجراء العمليات عليها مثل تخزينها مثلاً في قاعدة البيانات أو إرسالها إلى البريد الإلكتروني وذلك عن طريق الـphp

خصائص النماذج:

يجمع النموذج جميع خصائص المضيف لكننا هنا سنتطرق إلى اثنين منهما وهما ACTION و METHOD التي تستخدم بكثرة وهي مهمة لنا في دروسنا القادمة.

ACTION: وظيفة هذه الخاصية أن تخبر السيرفر مكان الصفحة التي يقوم بإرسال معلومات النموذج إليها أو عنوانها أياً كان نوعها وطبعاً في حالتنا ستكون الصفحة الثانية هي الصفحة التي تحتوي على سكربت الـphp .

```
<FORM  ACTION ="TEST.PHP">
…..
</FORM>
```

METHOD: هذه الخاصية تقوم بإخبار النموذج طريقة إرسال المعلومات إلى الصفحة الهدف وفي الواقع هناك طريقتين مشهورتين ومعروفتين لإرسال المعلومات هما GET و POST.

```
<FORM ACTION ="test.php" METHOD = "GET">
```

أو

```
<FORM ACTIN = "test.php" METHOD ="POST">
```

ويوجد طرق أخرى لإرسال المعلومات وهي:
(CONNECT;HEAD;OPTIONS:DELETE:TRACE) وغيرها ولكن لا تستخدم إلا بشكل نادر.

تقوم الخاصية GET بإخبار مستعرض الإنترنت لديك بأن يقوم بإضافة المعلومات التي تمت كتابتها في النموذج إلى متصفح الإنترنت لديك وتكون طريقة كتابته كالتالي:

1-كتابه عنوان الصفحة المصدر.
2-اتباعها بعلامة استفهام.
3-كتابة العناوين والقيم.

http://localhost/test.html?name=value

إن النموذج يتكون من عناصر (مربع علامة . مربع نص . زر اختيار) ولكل من هذه العناصر عنوان خاص بها (name) ولكل منها قيمه خاصة بها (value).
وهي مشابهة للمتغيرات ويمكن أن يحتوي عنوان الصفحة على أكثر من عنوان (name) واكثر من قيمة (value) ويقوم بالتعريف عنهما باستخدام المعامل (&).

مثال:

http://localhost/test.html?animal=cat&age=30

تسمى الإضافة التي تظهر بعد علامة الاستفهام (query String) نتيجة الاستعلام الحرفية.

العنوان دائما يكون باللغة الإنجليزية (name) ونعامله كأنه اسم متغير من المفترض تعريفه في الصفحة الهدف (التي سنكتبها بالـPHP).

أما POST فوظيفتها هي نفس وظيفة الـ get ولكنها لا ترسل المعلومات في عنوان صفحة الإنترنت بل تقوم وضعها في الـ body التابع لـ http response بالإضافة إلى أنه يستطيع إرسال البيانات بكمية أكبر من الـGET.

الفروقات بين GET و POST:

متى نستخدم GET ومتى نستخدم POST هناك بعض من الفروقات بين الطريقتين:

- الطريقة POST يوجد بها حماية اكثر من الطريقة GET.
- الطريقة POST تقوم بإرسال بيانات بكمية اكبر من الطريقة GET.
- الطريقة GET أسرع من الطريقة POST , لذلك نراها تستخدم في محركات البحث مثل google و yahoo و msn وغيرها.

ليست هذه كل الفروق بين العبارتين ولكن تعتبر هذه هي الفروق الأساسية.

كتابة النماذج:

في الواقع أن أدوات التحكم الخاصة بالنماذج هي عبارة عن مربعات النصوص العادية (التي يدخل فيها المستخدم اسمه وعنوانه) وأزرار (والتي يقوم المستخدم فيها باختيار شي معين (مثل الوجبة المفضلة لديه أو المشروب المفضل إليه) ومربعات الاختيار (التي تتيح للمستخدم أن يختار ما يشتهي ويحب من الخيارات المعروضة) وأيضا القوائم التي تساعدك على اختيار أكثر من شي أو شي واحد.

في أغلب هذه الأشياء يتم استعمال الوسم

<INPUT>

1 - TYPE= type

نحدد نوع الكائن إذا كان زر راديو أو مربع نص عادي أو مربعات الاختيار.

2 - NAME= name

تقوم فيها بإعطاء اسم لمتغير يتم حفظ القيمة فيه.

3 - VALUE= value

سيتضح وظيفته أكثر عندما ندرج عليه أمثله إذ أن عمله يختلف من أداة إلى أخرى.

وصيغته كالتالي:

<INPUT TYPE= type NAME= name VALUE= value other attribute>

هناك نوعين من الأزرار هي SUBMIT و RESET

1- الـ submit يقوم بإرسال المعلومات.

2- الـ reset يقوم بمسح البيانات في جميع الأدوات في النموذج لإعادة إدخالها من جديد .

أمثلة توضيحية:

سنقوم في هذه التطبيقات بصنع برامج بسيطة تتكون من ملفين، الملف الأول يحتوي على كود HTML يقوم بتكوين النموذج والملف الثاني يقوم باستقبال النتائج وطباعتها.

اكتب الكود التالي في محرر النصوص.

```
<html dir ="rtl">
<FORM METHOD  = "GET" ACTION= "edu.php">
ما هو مستوى تعليمك ؟
<br>
<INPUT TYPE = "text" NAME = "education" value = "">
<br>
<INPUT TYPE= submit  VALUE="إرسال">
<INPUT TYPE= reset  VALUE="مسح">
</form>
</html>
```

احفظ الملف كصفحة HTML . وقم بتسميته (edu1.html) .

اكتب الكود التالي في ملف جديد في محرر النصوص:

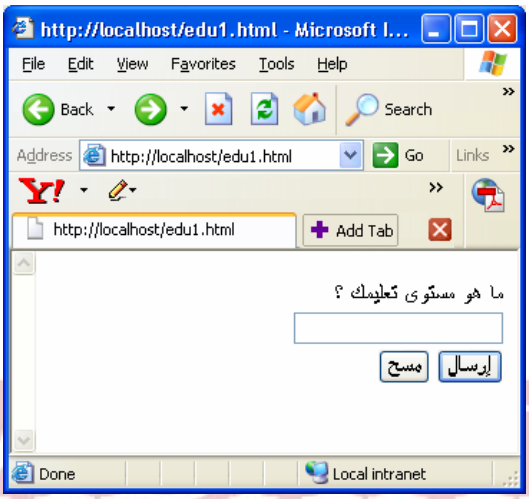
```
<html dir ="rtl">
<?php
Echo "أنا في المستوى ال"."$education";
?>
```

احفظ الملف كـ php . وقم بتسميته edu.php.

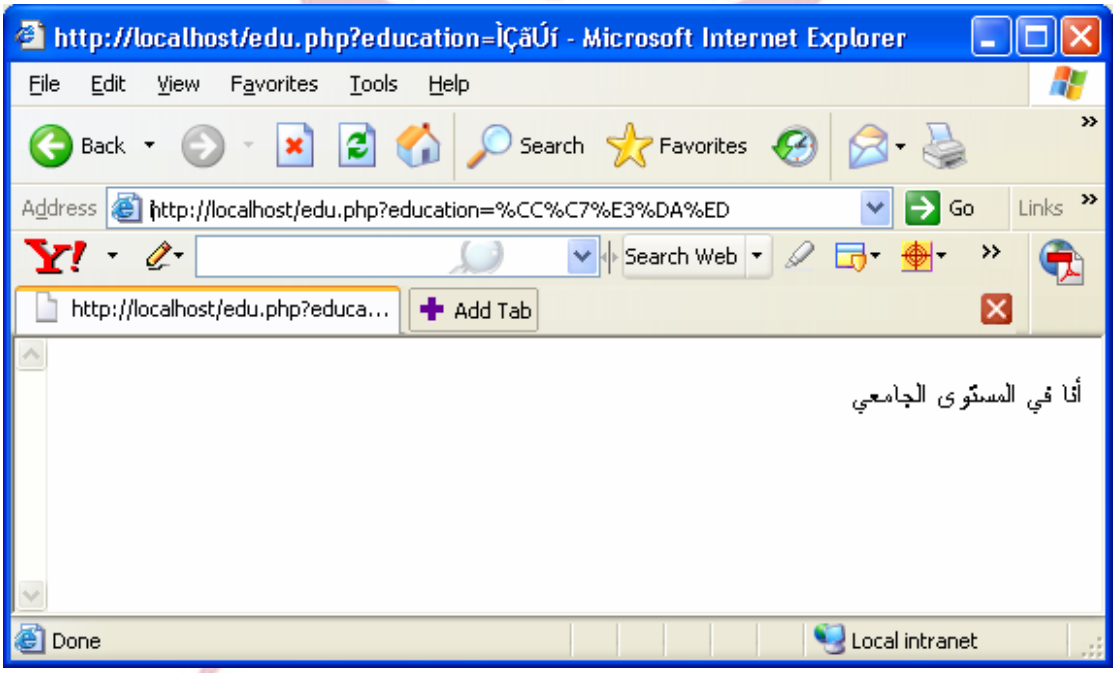
طبعاً أنت تعلم المسار الذي يجب أن تحفظ ملفاتك به...!!

من مستعرض الإنترنت نكتب العنوان التالي: http://localhost/edu1.html

سيظهر لك الملف الأول على الشكل:



نكتب في مربع مستوى التعليم المستوى الذي نريد ثم نضغط إرسال فتظهر لنا النتيجة على الشكل التالي:



لقد قمنا في البداية بعمل صفحة تتكون من نص ومربع نص وزر يقوم بعملية إرسال البيانات ثم حددنا بداية النموذج بواسطة الوسم <FORM> وقمنا بتحديد المكان الذي سيتم إرسال البيانات إليه بواسطة

ACTION="edu.php"

وقمنا بإنشاء مربع النص بواسطة الوسم INPUT واخترنا الـ

TYPE="text"

كما قمنا بوضع قيمة خالية بواسطة القيمة:

Value= ""

ثم وضعنا الناتج الذي يسجله المستخدم في مربع النص في المتغير education

ثم أضافنا زر بواسطة:

TYPE = SUBMIT
VALUE = "إرسال"

وزر آخر

Type = reset
Value = "مسح"

بعد أن قمنا بإدخال البيانات وضغط زر الإرسال قام النموذج بإرسال البيانات إلى الصفحة المحددة في الخاصية ACTION وقامت الصفحة المحددة باستقبال النتائج الموجودة في النموذج وهي نتيجة واحدة في مربع نصوص تم حفظ قيمته في المتغير education. وتمت طباعتها بواسطة الدالة echo.

أما إذا كنت تريد أن تكتب رسالة متعددة الأسطر فإنك تحتاج إلى أداة تحكم تختلف تماماً عن مربع النص العادي وهي مربعات النصوص الكبيرة التي يمكنك فيها من إدخال نصوص كبيرة الحجم ومتعددة الأسطر.

تستخدم هذه الأداة وسم فتح ووسم إغلاق

<TEXTAREA>
</TEXTAREA>

ويمكنك تحديد حجمها بواسطة تحديد الصفوف بالخاصية rows والأعمدة بالخاصية cols.

تمرين عملي:

اكتب الكود التالي في محرر النصوص.

```
<html dir="rtl">
<form action="2.php" method="GET">
<h3>ادخل الاسم
<input type="text" name="user">
<br>أدخل عنوانك بالتفصيل<p>
<textarea name="address" rows="5" cols="40">
</textarea></h3>
<br>
<input type="submit" value="ارسل البيانات">
</form>
</html>
```

احفظ الملف باسم 22.html

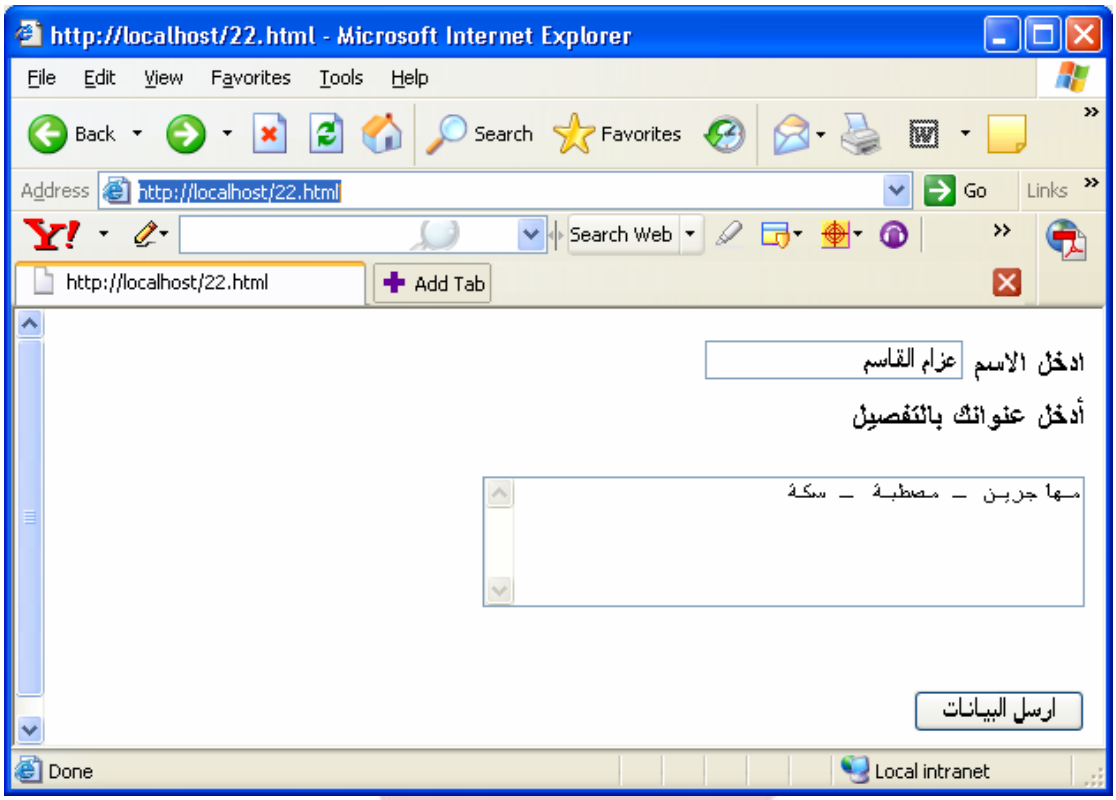
اكتب الكود التالي في ملف جديد في محرر النصوص:

```
<html dir="rtl">
<body><h2>
<?
print"أهلاً بك يا <b>$user</b><P>";
print"عنوانك المفصل هو:<P><b>$address</b>";
?>
</body>
</html>
```

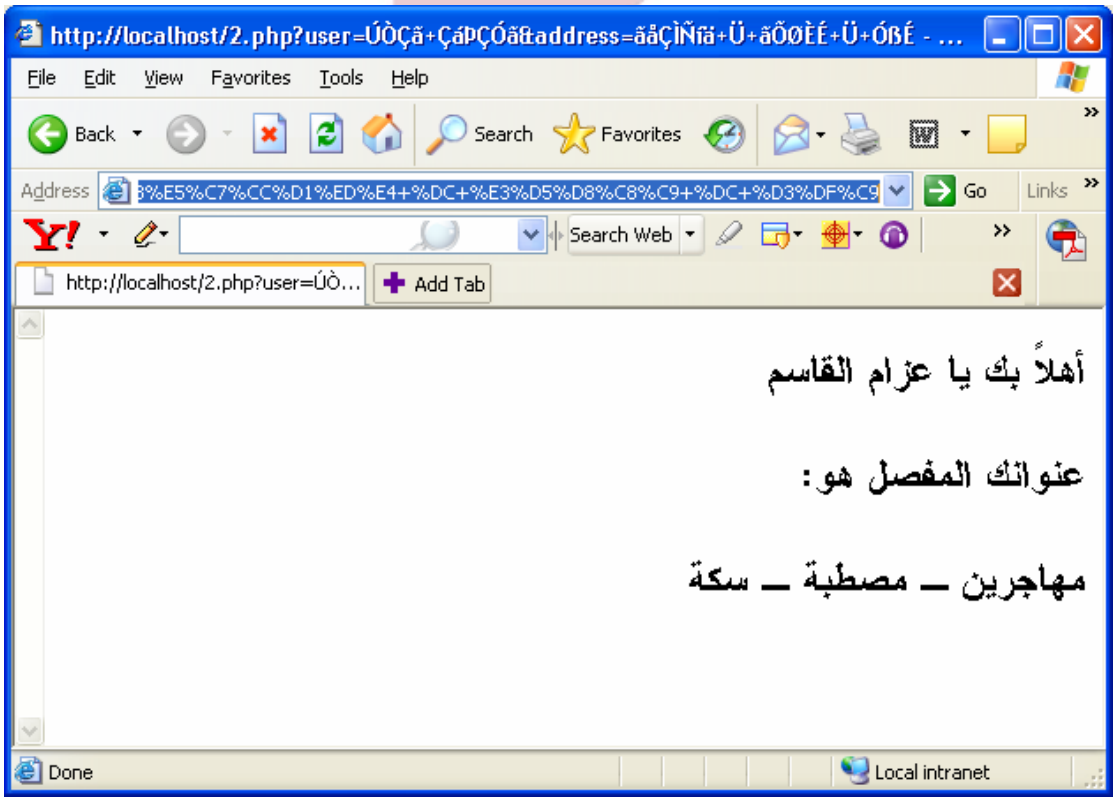
قم بحفظ الملف باسم 2.php

شغل البرنامج. http://localhost/22.html

أدخل البيانات واضغط زر الإرسال.



شاهد النتيجة



تمرين عملي:

سنقوم من خلاله بالتعامل مع أزرار الخيارات وربط ثلاثة صفحات معاً:

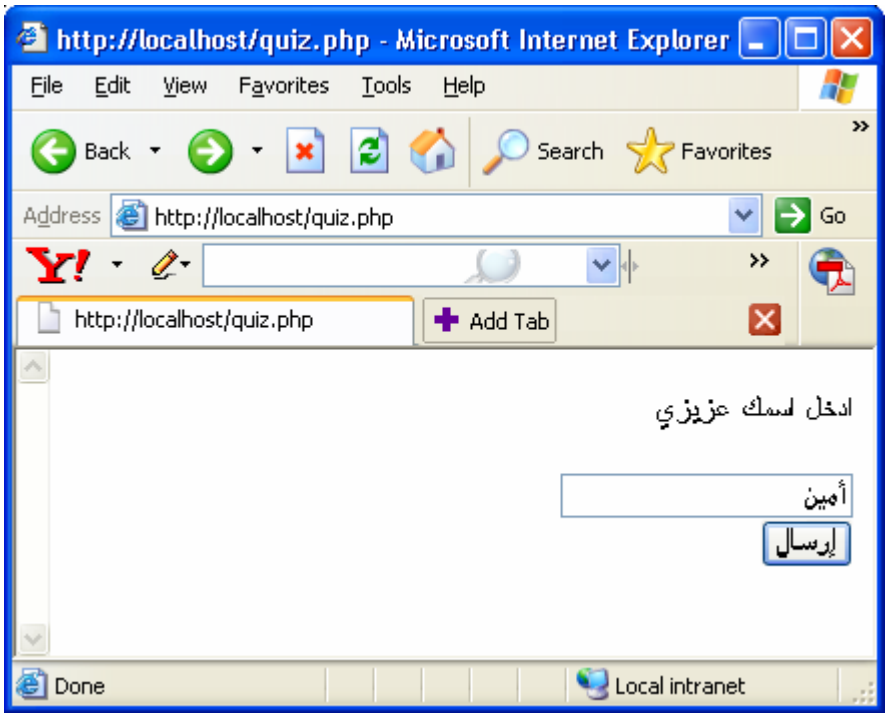
ستكون الأولى للإدخال الاسم والثانية للإجابة على بعض الأسئلة والثالثة لعرض النتيجة.

اكتب الكود التالي في محرر النصوص.

```
<html dir="rtl" align="center"><p>ادخل اسمك عزيزي </p>
<form method="POST" action="quiz2.php">
<input type="text" name="name" size="20"><br>
<input type="submit" value="إرسال"></p>
</form>
```

احفظ هذا الملف باسم quiz.php

وسيظهر على الشكل التالي:



نفتح ملف جديد على المفكرة ونكتب الصيغة التالية:

```
<html dir ="rtl">
<?
If  (isset($name)) {
Echo "مرحبا بك يا" ." ". $name ;
Echo '
<br>
<form method="POST" action="quiz3.php" dir="rtl">
<input type=hidden name = thename value = "$name">
من هو أول الخلفاء الراشدين ؟ </p>
<p dir="rtl"><input type="radio" value="أبوبكر الصديق" name="khlifa">أبوبكر الصديق </p>
<p dir="rtl"><input type="radio" value="عمر بن الخطاب" checked name="khlifa">عمر بن الخطاب</p>
<p dir="rtl">من هو الفاروق ؟</p>
<p dir="rtl"><input type="radio" name="faroq" value="عمر بن الخطاب">عمر بن الخطاب</p>
<p dir="rtl"><input type="radio" name="faroq" value="سالم" checked>سالم بن عامر</p>
<input type ="submit" value = "إرسال" dir="rtl">
</form>' ;
}
else
{
echo "غير مصرح لك بدخول هذه الصفحة" ;
}
?>
```

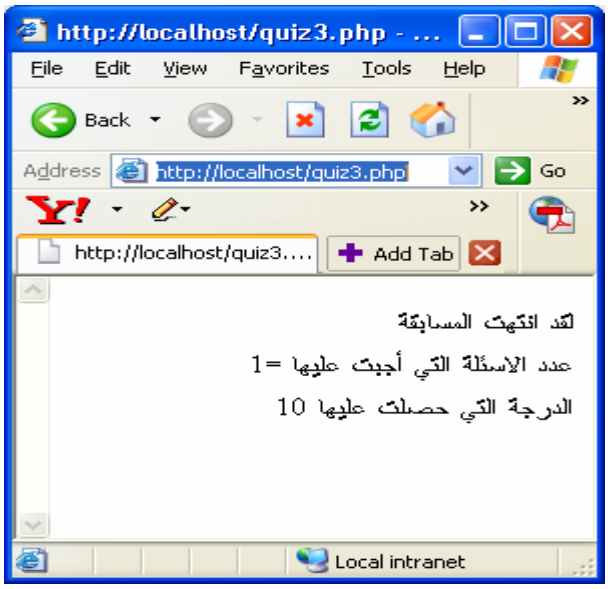
احفظ الملف باسم quiz2.php

وستظهر النتيجة على الشكل التالي:

لاحظ أن اسم اللاعب المسجل في الصفحة السابقة قد ظهر في بداية الصفحة والآن يمكن الإشارة على الإجابة الصحيحة ومن ثم ضغط إرسال لتظهر لنا النتيجة على الصفحة الثالثة والتي سنقوم بإعدادها على الشكل التالي بعد إنشاء ملف جديد على المفكرة وتسميته باسم quiz3.php نكتب الكود التالي:

```
<html dir="rtl">
<?
If  ((isset($thename)) && (isset($khlifa)) && (isset($faroq)))
{
Echo "لقد انتهت المسابقة"." ";
$range=0;
$co = 0;
      if ($khlifa =="أبوبكر الصديق") {
      $range=$range+10;
      $co = $co +1;
      }
      if ($faroq =="عمر بن الخطاب")
         {
         $range=$range+10;
         $co=$co+1;
         }
       if ( $range < 10)
       {
       echo "ليس هناك أي إجابة صحيحة";
       }
       else
       {
       echo "<br>"."عدد الاسئلة التي أجبت عليها =" . $co ;
       echo "<br>" . "الدرجة التي حصلت عليها " . $range ;
       }
}
?>
```

والآن ستظهر لنا النتيجة على الشكل التالي وذلك حسب الإجابات الصحيحة:

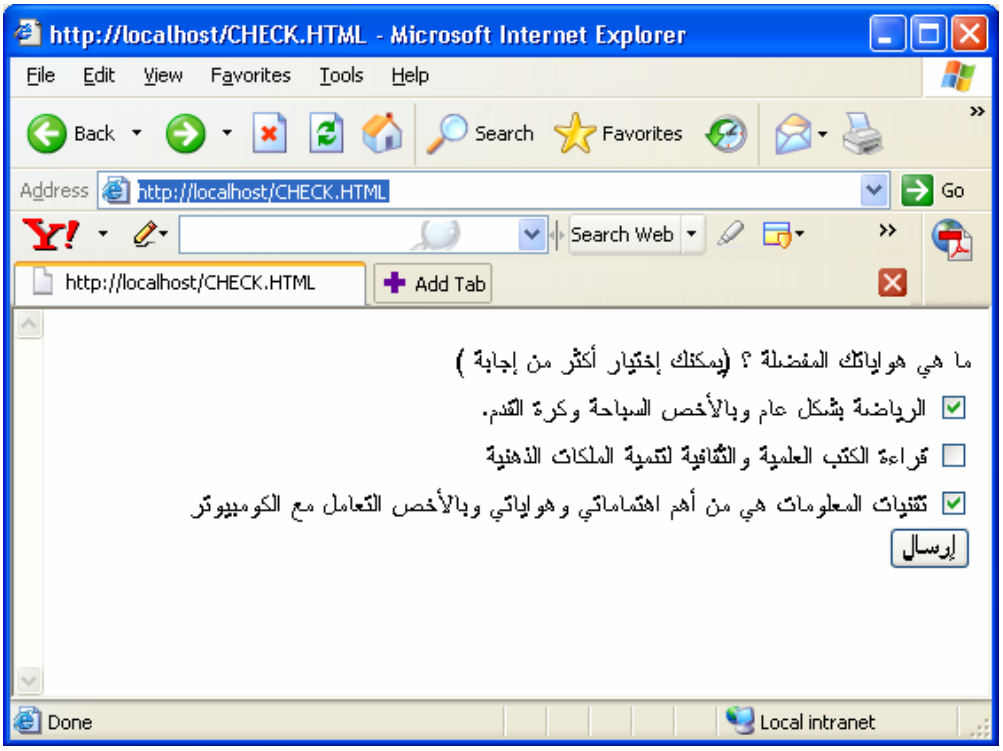


تطبيق عملي:

الملف الأول check.htm:

```
<html dir="rtl">
<FORM ACTION="CHECK.PHP" METHOD = "POST">
ما هي هواياتك المفضلة ؟ (يمكنك إختيار أكثر من إجابة)
<br>
<INPUT TYPE="CHECKBOX" NAME = "academy[]" value= "رياضة"
CHECKED>
الرياضة بشكل عام وبالأخص السباحة وكرة القدم.
<br>
<INPUT TYPE="CHECKBOX" NAME = "academy[]" value= "مطالعة" >
قراءة الكتب العلمية والثقافية لتنمية الملكات الذهنية
<br>
<INPUT TYPE="CHECKBOX" NAME = "academy[]" value= "كومبيوتر"
CHECKED>
تقنيات المعلومات هي من أهم اهتماماتي وهواياتي وبالأخص التعامل مع الكومبيوتر
<br>
<input type= submit value = "إرسال">
</FORM>
</html>
```

النتيجة:

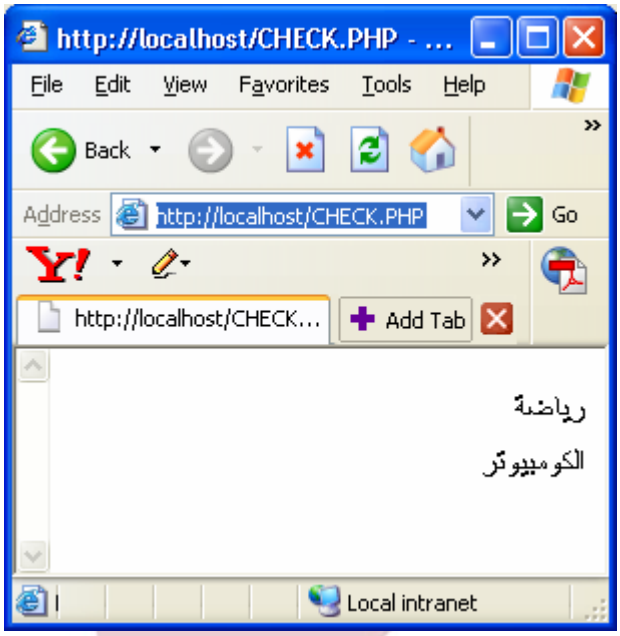


الملف الثاني check.php:

```
<html dir="rtl">
<?
```

```
Echo  "$academy[0] <br>" ;
Echo  "$academy[1] <br>" ;
Echo  "$academy[2] <br>" ;
?>
</html>
```

النتيجة:



القوائم (Lists Or drop down menus):

سنستخدم وسمين من وسوم لغة html وهما: <select> لنقوم بإنشاء القائمة و <OPTION> ونستخدم الخاصية MULTIPLE إذا كنا نريد إتاحة الفرصة للمستخدم أن يختار أكثر من قيمة ونقوم بوضع القيمة التي يختارها المستخدم في متغير بواسطة الخاصية NAME أو في مصفوفة متغيرات.

تطبيق عملي:

الملف الأول test.htm:

```
<html dir="rtl">
<form action = "test.php" method = "post">
ما هي الدولة التي تنتمي إليها ؟
<br>
<select name = "city">
<option> سوريا </option>
<option> مصر </option>
<option>لبنان </option>
<option>فلسطين </option>
<option>الكويت </option>
<option> قطر </option>
<option>السعودية </option>
<option>الامارات </option>
<option>دولة أخرى </option>
</select>
<BR>
عرف عن نفسك!!
<Br>
<select name="dis[]" multiple>
<option>مثقف</option>
<option>جامعي</option>
<option>متدين </option>
<option>مدخن</option>
<option>رياضي </option>
<option>سمين </option>
<option>هادئ </option>
<option>سريع الغضب </option>
</select>
<br>
<INPUT TYPE=SUBMIT VALUE="إرسال">
</html>
```

الملف الثاني test.php:

```
<html dir="rtl">
<?
Echo "إن الدولة التي أنتمي إليها هي " . " " .
$city;
Echo "<br><br>";
Echo "وهذه هي  بعض مواصفاتي ";

Echo "<br><br>";
Echo "$dis[0] <br>";
Echo "$dis[1] <br>";
Echo "$dis[2] <br>";
Echo "$dis[3] <br>";
Echo "$dis[4] <br>";
Echo "$dis[5] <br>";
Echo "$dis[6] <br>";
?>
</html>
```

حلل التمرين السابق وحاول أن تُضيف إليه بعض التعديلات....!!

استخدام حقل كلمات السر (Password fields):

لكي تجعل المعلومات أكثر حماية من التعرض إلى السرقة أو غير ذلك يمكنك استخدام حقول كلمات السر والتي هو عبارة عن مربع نص بسيط يقوم بإظهار النص على شكل نجوم ****** أو ●●●●●● .

في الواقع مع ذلك فإنك لا تكون قد أديت حماية إذا كان الأسلوب المستخدم في إرسال بيانات المستخدم هو الأسلوب get إلا إذا كنت تستخدم تشفير البيانات ويكون أكثر جودة إذا استخدمت الأسلوب post وأيضاً لن يكون محمياً من الهاكر إذا لم تكن تستخدم SSL (Secure Socket Layer) لكي تقوم بتنشيط تشفير البيانات.

تطبيق عملي:

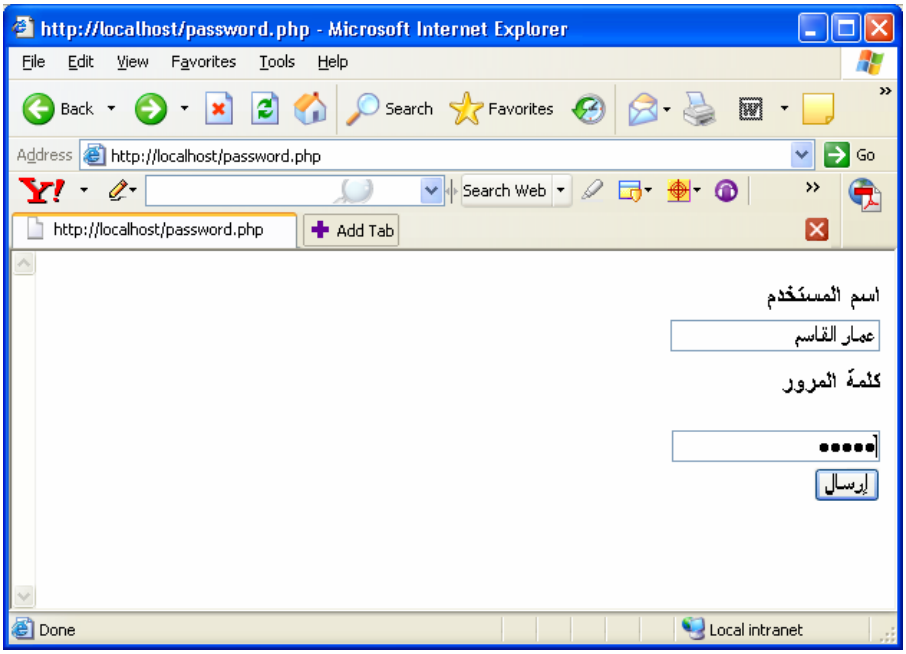
الملف الأول password.php:

```
<html dir="rtl">
<body>
<form method=post action="password1.php">
<h3><b>اسم المستخدم
<br>
<input type="text" name ="user">
<br>
كلمة المرور</h3></b>
<input type="password" name ="pass">
<br>
<input type = submit  value="إرسال">
</form>
<body>
</html>
```

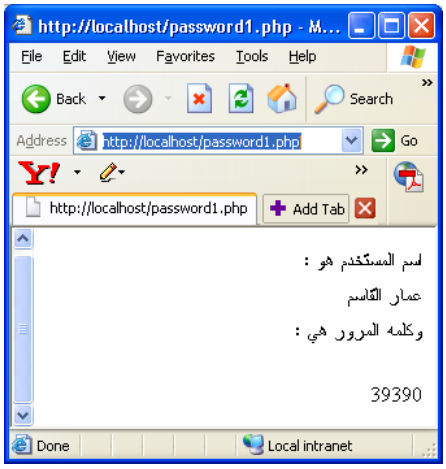
الملف الثاني password1.php:

```
<html dir="rtl">
<?
Echo "اسم المستخدم هو: ";
Echo "<br>$user<br>";
Echo "وكلمه المرور هي:";
Echo "<br><br>$pass"
?>
</html>
```

النتائج:



وعند ضغط إرسال سيظهر لنا:



المصفوفات

أنواع المصفوفات:

يوجد في لغة PHP نوعان من المصفوفات مصفوفات رقمية ومصفوفات حرفية في البداية سوف نقوم بتعلم المصفوفات الرقمية لأنها الأساس إلا أن جميعهم يؤدون نفس العمل.

المصفوفات الرقمية:

المصفوفات الرقمية هي المصفوفات التي يتم استعمالها عن طريق الأرقام فعندما أريد قيمه من هذه المصفوفة فسوف أقوم بالاستعلام عنها باستخدام الرقم الذي يدل عليها، إذن نعرف أن لكل قيمه في المصفوفة رقم يدل عليها وغالبا يبدأ من العدد 0 وطبعاً. والقيم الموجودة بالمصفوفة نسميها عناصر والأرقام الموضوعة لكل قيمة في المصفوفة نسميها فهرس أو المفتاح.

إنشاء المصفوفات الرقمية:

سأوضح مفهوم المصفوفة وكيفية إنشاءها من المثال التالي:

لنفترض أننا نريد كتابة مواد مقرر الدبلوم الدولي العالي في تقنيات المعلومات داخل مصفوفة... سنقوم بإنشاء مصفوفة اسمها مواد، ونضع داخلها أسماء المقررات..

افتح المفكرة وقم بكتابة التالي:

```
<?
```

```
$items1 = " أساسيات تقنيات المعلومات";
$items2 ="رياضيات الحاسب";
$items3 =" الحواسيب والدارات المنطقية تجميع";
$items4 ="قيادة الحاسب";
$items5 ="صيانة الحواسيب";
$items6 ="هندسة الشبكات";
$items7 ="الشبكة الدولية للمعلومات وتطبيقاتها";
$items8 ="تكنولوجيا الأعمال";
$items9 ="Adobe Photoshop";
$items10 ="Visual basic";
$items11 ="HTML";
$items12 ="Java Script";
?>
```

ويمكن أيضاً كتابتها على الشكل:

```
<?
$items = array ( " أساسيات تقنيات المعلومات" , "رياضيات الحاسب" , " الحواسيب والدارات المنطقية تجميع" , "قيادة الحاسب" ,
"صيانة الحواسيب" , "هندسة الشبكات" , "الشبكة الدولية للمعلومات وتطبيقاتها" , "تكنولوجيا الأعمال" , "Adobe Photoshop" ,
"Visual basic" , "HTML" , "Java Script");
?>
```

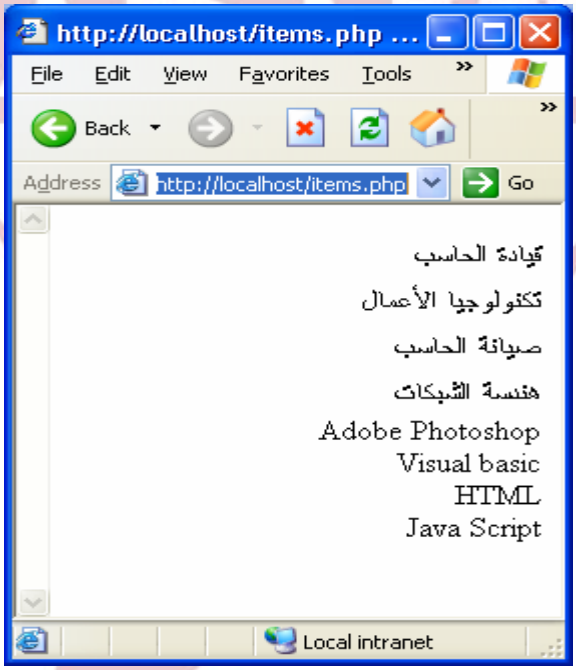
طباعة المصفوفات الرقمية:

لطباعة المصفوفة علينا الوصول إلى كل عنصر على حده وطباعته على حده انظر المثال التالي:

```
<html dir="rtl">
<?
$items = array ("قيادة الحاسب","تكنولوجيا الأعمال","صيانة الحاسب","هندسة الشبكات","Adobe
Photoshop" , "Visual basic" , "HTML" , "Java Script");
echo $items[0];
echo "<br>";
echo $items[1];
echo "<br>";
echo $items[2];
echo "<br>";
echo $items[3];
echo "<br>";
echo $items[4];
echo "<br>";
echo $items[5];
```

```
echo "<br>";
echo $items[6];
echo "<br>";
echo $items[7];
echo "<br>";
echo $items[8];
?>
</html>
```

النتيجة:



نستطيع أن نستخدم دالة تكرار لتقوم بعمل الطباعة بدلاً من أن نقوم بكتابة بكل سطر على حده:

```
<html dir="rtl">
<?
$items = array ("قيادة الحاسب","تكنولوجيا الأعمال","صيانة الحاسب","هندسة الشبكات","Adobe
Photoshop" , "Visual basic" , "HTML" , "Java Script");
for ($i = 0; $i<8; $i++)
{
echo $items[$i];
echo "<br>";
}
?>
```

لاحظ أن النتيجة نفسها ولكن بأسطر وجهد أقل بكثير..

هنـاك العديـد مـن الـدوال التـي يوفرهـا لنـا الـPHP لفـرز المـصفوفات. أكثرهـا استخداماً:

الدالة Sort():

تقوم هذه الدالة بفرز المصفوفة هجائياً اعتماداً على الأحرف الكبيرة أولاً ثم الصغيرة .. تتطلب هذه الدالة اسم المصفوفة التي سيتم عليها الفرز:

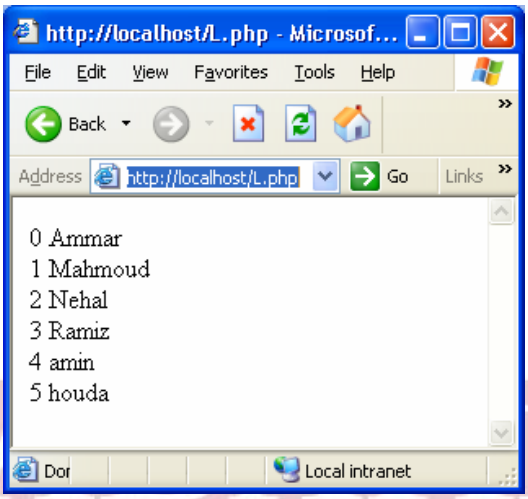
```
Sort (ArrayName);
```

أنشئ المصفوفة بالشكل التالي:

```
$Lolo=array
("amin","Mahmoud","houda","Nehal","Ammar","Ramiz");
```

فإذا أردنا فرزها عن طريق الدالة sort() فإننا نقوم باستخدامها كالتالي:

```
<?
$Lolo =array
("amin","Mahmoud","houda","Nehal","Ammar","Ramiz");
sort($Lolo);
While (list($e,$r) = each ($Lolo))
{
echo "$e\t\t$r<br>";
}
?>
```



لاحظ أن الـPHP قام بالفرز اعتماداً على الأحرف الكبيرة أولاً ثم قام بالفرز بعدها اعتماداً على الأحرف الصغيرة.

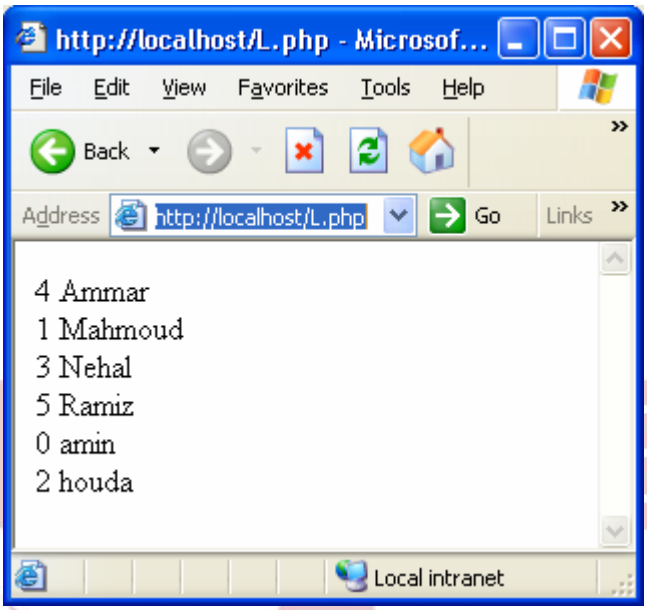
الدالة Arsort():

هذه الدالة تعمل نفس عملية الدالة sort() ولكن هناك اختلاف بسيط فالدالة asort تقوم بإظهار الترتيب الصحيح لعناصر المصفوفة حسب توضعها داخلها حسب الصيغة التالي:

لنأخذ المثال السابق ونبدل rsort بـ arsort:

```
<?
$Lolo =array
("amin","Mahmoud","houda","Nehal","Ammar","Ramiz");
asort($Lolo);
While (list($e,$r) = each ($Lolo))
{
echo "$e\t\t$r<br>";
}
?>
```

لاحظ الفرق:

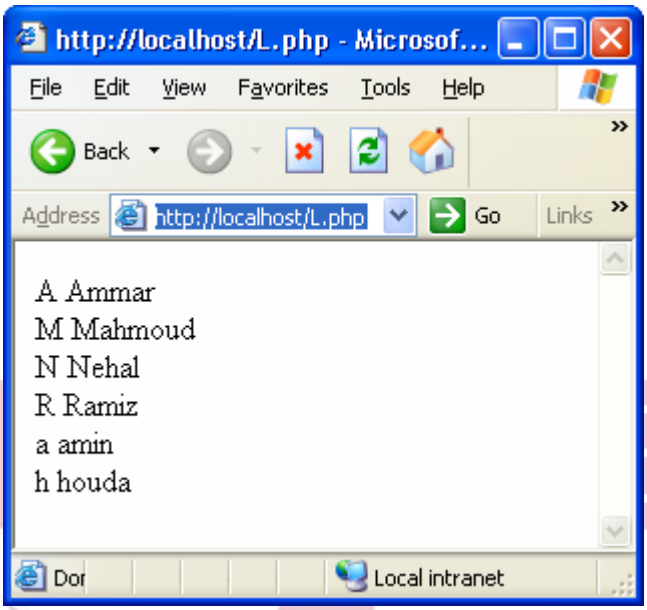


ولاحظ الترقيم....

كما يمكن للدالة أيضاً أن تستبدل فهرسة الحروف بالأرقام.

لاحظ....

```
<?
$Lolo=array
("a"=>"amin","M"=>"Mahmoud","h"=>"houda","N"=>"Nehal",
"A"=>"Ammar","R"=>"Ramiz");
asort($Lolo);
While (list($e,$r) = each ($Lolo))
{
echo "$e\t\t$r<br>";
}
?>
```


http://localhost/L.php - Microsof...
File Edit View Favorites Tools Help
Back
Address http://localhost/L.php
Go
Links
A Ammar
M Mahmoud
N Nehal
R Ramiz
a amin
h houda
Local intranet

:array_push()

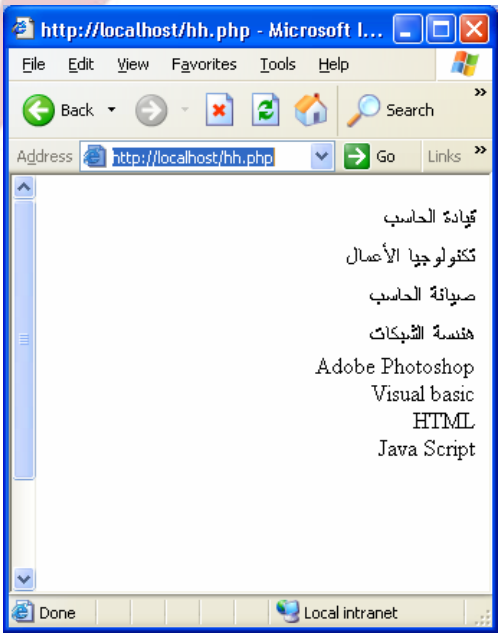
نستطيع إضافة عنصر على المصفوفة عن طريق الدالة array_push() كالتالي:

array_push (ArrayName اسم المصفوفه,Elemnt1, Elemnt2, Elemnt3,…..)

نضع في القسم الأول من الدالة اسم المصفوفة التي نريد إضافة العنصر لها ونضع في القسم الثاني عنصر واحد أو أكثر وهي التي سيتم إضافتها للمصفوفة.

وإذا عدنا إلى المصفوفة السابقة:

```
<html dir="rtl">
<?
$items = array (" هندسة","صيانة الحاسب","تكنولوجيا الأعمال","قيادة الحاسب
الشبكات","Adobe Photoshop" , "Visual basic" , "HTML" , "Java Script");
for ($i = 0; $i<8; $i++)
{
echo $items[$i];
echo "<br>";
}
?>
```

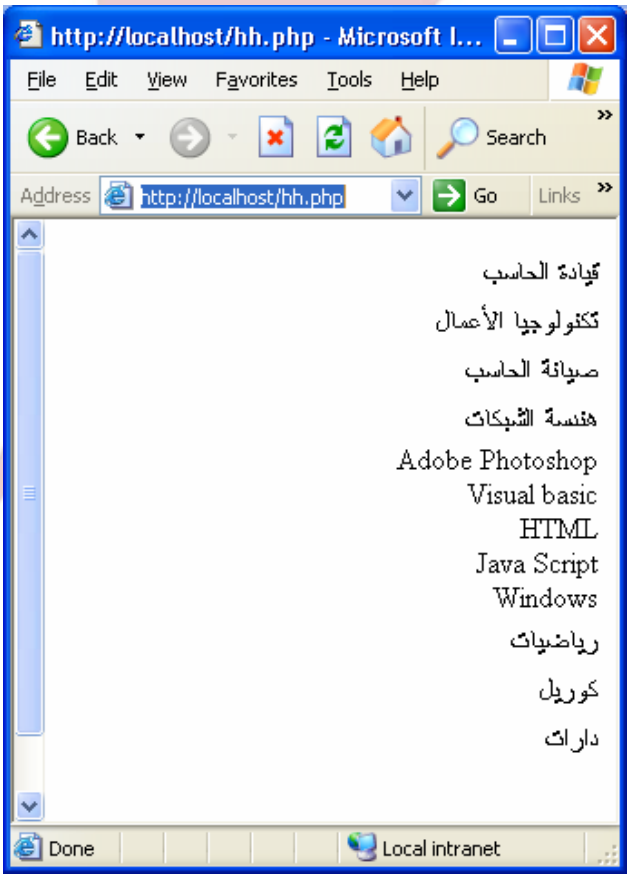


وإذا أردنا أن نضيف عنصر جديد لها ستقول طبعاً هذا سهل... ولكن كيف ستأخذ الترقيمات وأين ستبحث عن السطر الخاص بها وأين ستضع الإضافات....

لنستريح من كل ذلك ونعدل الصيغة كما يلي: (طبعاً سيكون الوضع مريح جداً عندما يبلغ عن البرنامج 200 سطر مثلاً..)

```
<html dir="rtl">
<?
$items = array ("هندسة الشبكات","صيانة الحاسب","تكنولوجيا الأعمال","قيادة الحاسب","Adobe
Photoshop" , "Visual basic" , "HTML" , "Java Script");
array_push ($items,Windows,دارات,كوريل,رياضيات);
for ($i = 0; $i<14; $i++)
{
echo $items[$i];
echo "<br>";
}
?>
```

لاحظ الآن...



array_pop():

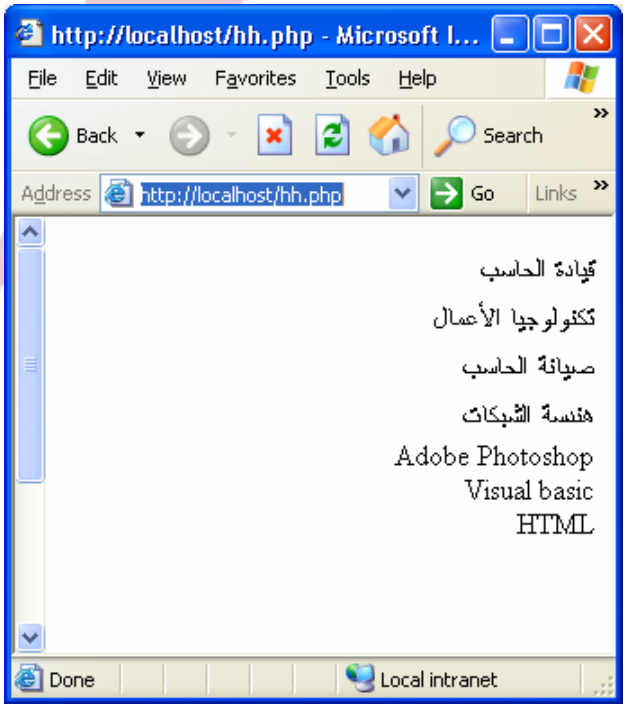
أما إذا أردنا حذف عنصر من المصفوفة فإننا نقوم بتعريف المصفوفة من جديد أو يمكننا استخدام الدالة array_pop التي تقوم بحذف آخر عنصر من المصفوفة والتي تتطلب فقط اسم المصفوفة.

Array_pop(ArrayName اسم المصفوفه)

مثال:

```
<html dir="rtl">
<?
$items = array ("هندسة الشبكات","صيانة الحاسب","تكنولوجيا الأعمال","قيادة الحاسب","Adobe
Photoshop" , "Visual basic" , "HTML" , "Java Script");
array_pop($items);
for ($i = 0; $i<14; $i++)
{
echo $items[$i];
echo "<br>";
}
?>
```

لاحظ الآن....



لقد اختفى العنصر الأخير من المصفوفة.

الدالة Implode:

تقوم بإضافة قيمة بين عناصر المصفوفة .

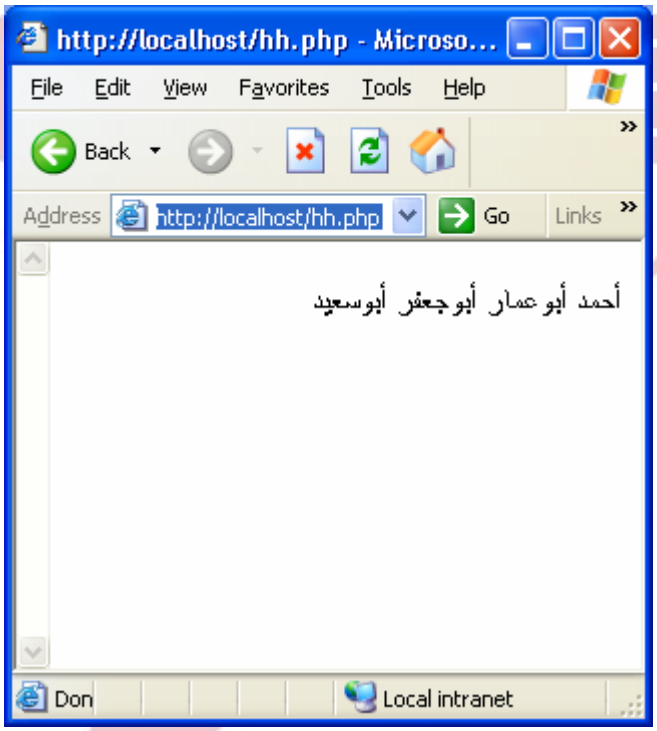
مثال:

```
<html dir="rtl">
<?
$items =array ("أحمد","عمار","جعفر","سعيد");
echo $items[0];
echo "<br>";
echo $items[1];
echo "<br>";
echo $items[2];
echo "<br>";
echo $items[3];

?>
```

فلو أردنا إضافة كلمة أبو إلى عناصر المصفوفة...

```
<html dir="rtl">
<?
$items =array ("أحمد","عمار", "جعفر", "سعيد");
$r =implode (" ".".أبو",$items);
echo $r;
?>
```



لاحظ الفرق ولا حظ أنني طبعت النتائج بطريقة مختلفة...

الدالة explode:

تقوم بحذف قيمة من مصفوفة وذلك لا يعني حذف عناصر من المصفوفة.

التحكم بالنصوص وإدراج الملفات

التحكم بالنصوص:

إن التحكم بالنصوص ضروري جداً إذا ما أردنا التعامل مع تلك النصوص فنحن نضع معلومات التسجيل في موقع ما تم نقوم بإرسالها إليهم، سواء الاسم أو البريد أو..... ولكنها تمر بعد فلترتها طبعاً فكافة المواقع لا تضمن ما يقوم بإدخاله المستخدم، هل هو المطلوب أم لا.

الدالة trim:

تستخدم الدالة trim لإلغاء المسافات من بداية ونهاية النص، لاحظ المثال التالي:

```
<html dir="rtl">
<?
$name = " Azzam    ";
$name = trim($name);
echo $name;
?>
```

الدالة chop و ltrim:

الدالة ltrim تقوم بنفس عمل trim ولكن تلغي المسافات من الجزء الأيسر، أما الدالة chop فتقوم بإلغاء المسافة من الجزء الأيمن.

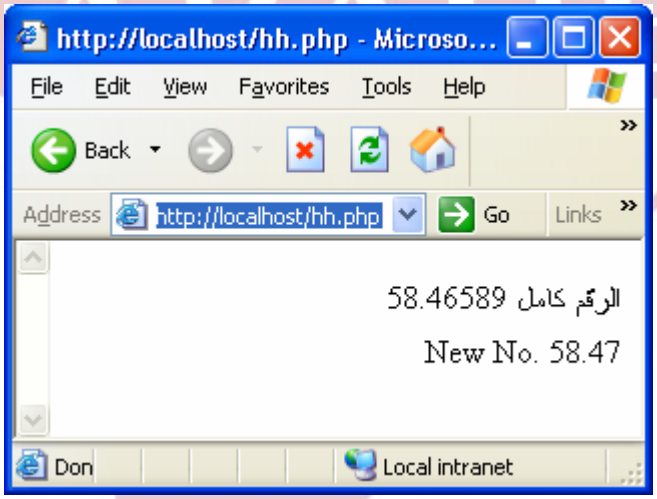
الدالتين printf و sprintf:

تقومان هاتين الدالتين بنفس عمل الدالة print والفرق بينهما وبين العبارة print أنها تقوم بطباعة النص بطريقه مختلفة.

فعلى سبيل المثال إذا كنا نريد أن نطبع العدد التالي 58.46589 ولكن دون إظهار كامل الرقم أي رقمين بعد الفاصلة فقط. فسوف نكتب التالي:

```
<html dir="rtl">
<?
$total = 58.46589;
echo " الرقم كامل ".$total;
echo "<br>";
printf ("New No. %.2f" , $total);
?>
```

لاحظ الآن...



ملاحظة: كل خواص التحويل تبدأ بعلامة % ولطباعة هذا الرمز إلى المتصفح استخدم %%. بعض الخواص الأخرى في التحويل:

النوع	المعنى
b	يترجم على شكل عدد صحيح ويطبع على شكل ثنائي
c	يترجم على شكل عدد صحيح ويطبع على شكل على حروف
d	يترجم على شكل عدد صحيح ويطبع على شكل عشري
f	يترجم على شكل عدد عشري double ويطبع على شكل عشري (float)
s	يترجم على شكل نص ويطبع على شكل نص

الدالة strtoupper:

تقوم بجعل كل الحروف كبيرة خاصة باللغة الإنجليزية أي تقوم بتحويل a إلى A وهكذا.

الدالة strtolower:

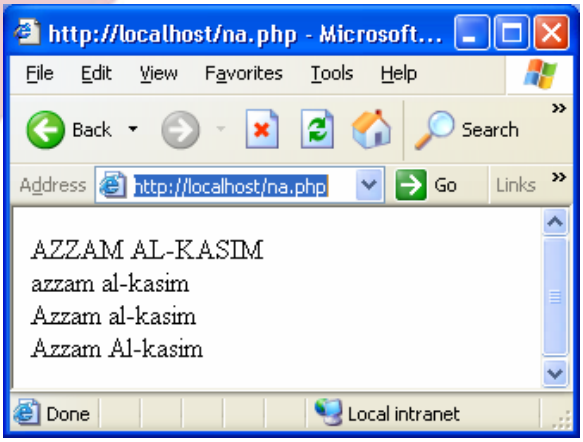
عكس الدالة السابقة فهي تقوم بجعل كل الحروف صغيرة خاصة باللغة الإنجليزية أي تقوم بتحويل A إلى a وهكذا.

الدالة ucfirst: تقوم بتحويل أول حرف من الجملة للشكل الكبير.

الدالة ucwords: تقوم بتحويل أول حرف من كل كلمة في النص إلى الحالة الكبيرة.

لاحظ التطبيق التالي:

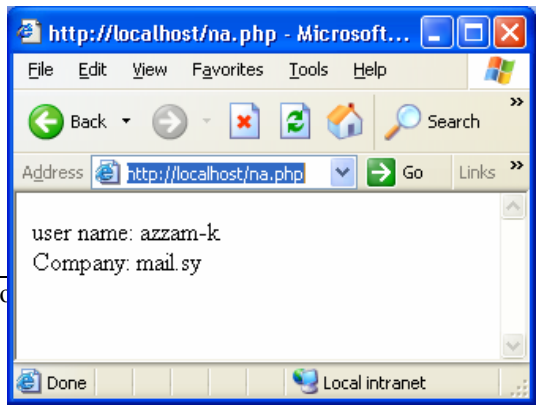
```
<?
$name = 'azzam al-kasim';
echo strtoupper ($name);
echo '<br>';
echo strtolower ($name);
echo '<br>';
echo ucfirst ($name);
echo '<br>';
echo ucwords ($name);
echo '<br>';
?>
```



الدالة explode:

الدالة explode تستخدم لفصل النص عن بعضه البعض وتقوم بإرجاع الناتج على شكل مصفوفة:

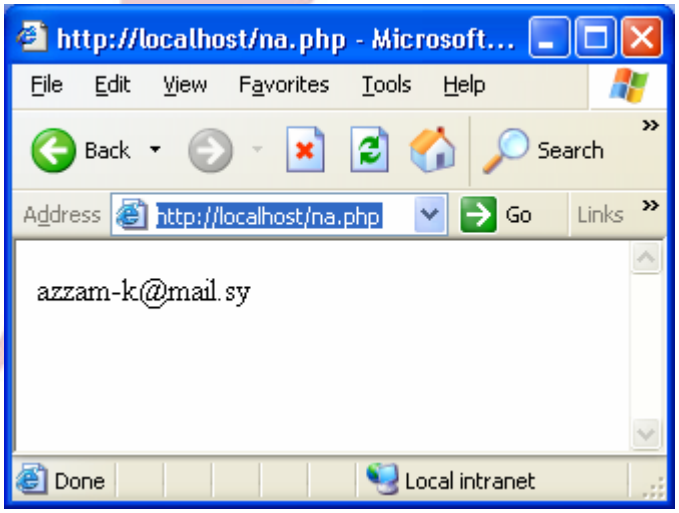
```
<?
```



```
$mail = 'azzam-k@mail.sy';
$site = explode ('@' , $mail);
echo 'user name:'." ".$site[0];
echo '<br>';
echo 'Company:'." ".$site[1];
?>
```

الدالتين **implode** و **join**: متطابقتان تماماً لجمع نص مع بعضه البعض:

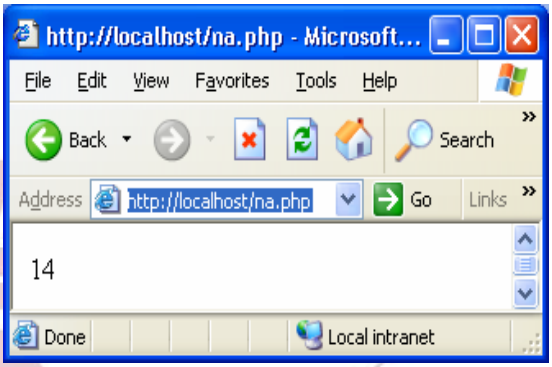
```
<?
$mail[0] = 'azzam-k';
$mail[1] = 'mail.sy';
$new_email = implode ( '@' , $mail);
echo $new_email;
?>
```



الدالة strlen:

تستخدم هذه الدالة لمعرفة طول نص معين وطريقتها كالتالي:

```
<?
$text1 = "azzam al-kasim";
$result = strlen ($text1);
echo $result;
?>
```

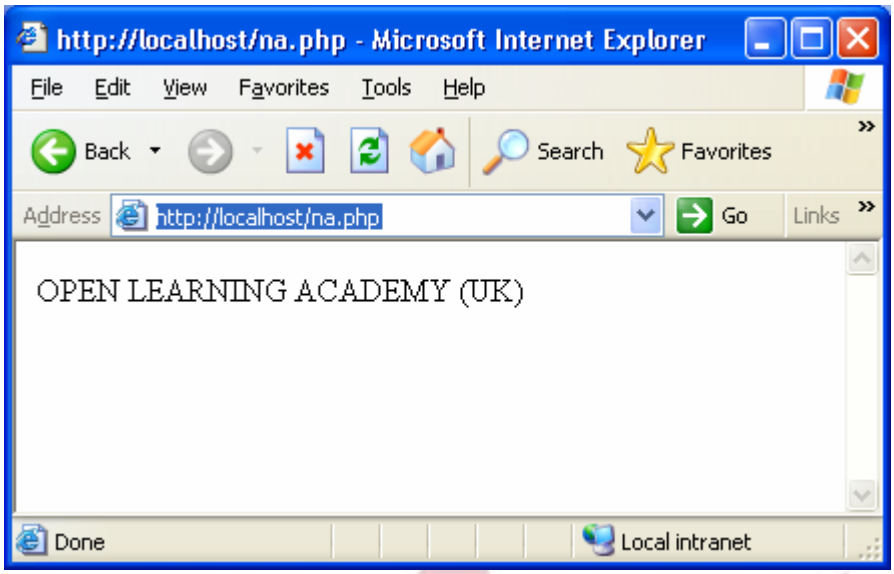


الدالتين strstr و strchr:

متطابقتين تماماً وحساستان لحالة الأحرف، تستخدم strchr للبحث عن الأحرف ولكن الأولى تؤدي نفس العمل وهي الأفضل والمناسبة دائماً طريقة هذه الدالة كالتالي:

```
<?
$text = "VIRTUAL AND OPEN LEARNING ACADEMY (UK)";
$find = "OPEN";
$result = strstr ($text, $find);
echo $result;
?>
```

لاحظ أن الجملة أو النص هو قيمة $text أما الكلمة المُراد البحث عنها فتمثلها قيمة $find وفي مثالنا هي OPEN فإذا وجدت الدالة كلمه مطابقة فسوف تقوم بإرجاع الكلمة ثم باقي الجملة وتهمل ما قبل الكلمة التي عثرت عليها لاحظ ناتج مثالنا:



الدالة stristr: مطابقة لـ strtsr والفرق هو أنها غير حساسة لحالة الأحرف أما **الدالة strrchr** فهي أيضاً مطابقة والفرق الوحيد هو انه إذا كانت الكلمة مكرره مرتين في الجملة فسوف تقوم بإرجاع النص من آخر مكان تكررت فيه الجملة على عكس strstr فهي تقوم بإرجاع النص من أول مكان تكررت فيه الجملة.

دالة str_replace:

تقوم هذه الدالة بإيجاد نص وتغييره وهي مفيدة جداً وطريقتها كالتالي:

```
<html dir="rtl">
<?
$text = "فلسطين الحبيبة تناديكم يا أهل العزة والكرامة ";
echo $text;
echo "<br>";
$oldwords = array ("الحبيبة","تناديكم", "والكرامة");
$newwords = array ("الجريحة","تستصرخكم","والمروءة");
$newtext = str_replace($oldwords, $newwords, $text);
echo $newtext;
?>
```

النتيجة:

انتبه... فقد طبعنا الجملة قبل التعديل ومن ثم بعد التعديل...

أعتقد أن الدالة واضحة...

إدراج الملفات:

عند تصفحك لمواقع الانترنيت الضخمة والمعروفة تجد أنه في الغالب أن رأس الصفحة وتذيلها متكرر بشكل عام في كل الصفحات والتغيير يحدث فقط في محتوى الصفحة وذلك يتم من خلال إدراج ملفات خاصة بالترويسة والتذييل لنرى الآن كيفية الإدراج.

الدالة require:

هذه الدالة تستخدم لإدراج الملفات وطريقة كتابتها كالتالي:

Require (string file) لنأخذ مثال عليها لتوضيح الصورة.

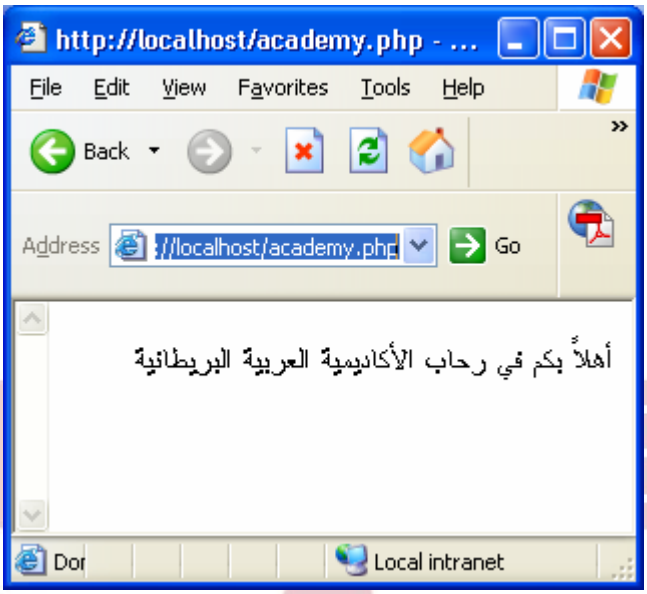
افتح المفكرة واكتب الصيغة التالية ثم احفظها باسم head.php:

```
<?
    echo "أهلاً بكم في رحاب الأكاديمية العربية البريطانية";
?>
```

افتح ملف جديد في المفكرة واكتب الكود التالي ثم احفظه باسم academy.php:

```
<html dir="rtl">
<?
    require ( 'head.php' );
?>
```

اطلب الملف academy.php من متصفحك وشاهد النتيجة:



نستنتج أن الدالة require تحتاج إلى ملف لكي يكون مدخلها وعندما ننفذ الكود فكأننا كتبنا الملف head.php داخل الملف الأساسي.

يمكن إدراج أي ملف تريد بأي امتداد تريد، ولكن من المستحسن ادرج ملفات PHP أو ملفات HTML، ولكن لا تنسى أنه إذا كان الملف HTML فلن يتم تنفيذ كود PHP بداخله، يجب أن يكون PHP لينفذ.

الدالة include:

هذه الدالة لها نفس عمل الدالة require تماماً، الفرق الوحيد بينهما هو كيفية إخراج الخطأ فلو أخطأت في الدالة include فسوف يكون الخطأ هو تحذير ويكمل تنفيذ عمل الملف (Warning) أما إذا أخطأت في الدالة require فسوف يكون الخطأ رئيسي ويتوقف عمل تنفيذ الملف أي الخروج من الكود بشكل كامل (Fatal Error).

قواعد البيانات

قاعدة البيانات MySQL:

هناك كثير من قواعد البيانات الموجودة الآن، ولكن أغلب مبرمجين PHP يستخدمون هذه القاعدة لخصائصها: فهي مجانية . مفتوحة المصدر . سريعة . سهلة. ولكن أهم عيوبها أنها لا تدعم العلاقات إلا بطريقة معينة.

تصميم قاعدة بيانات:

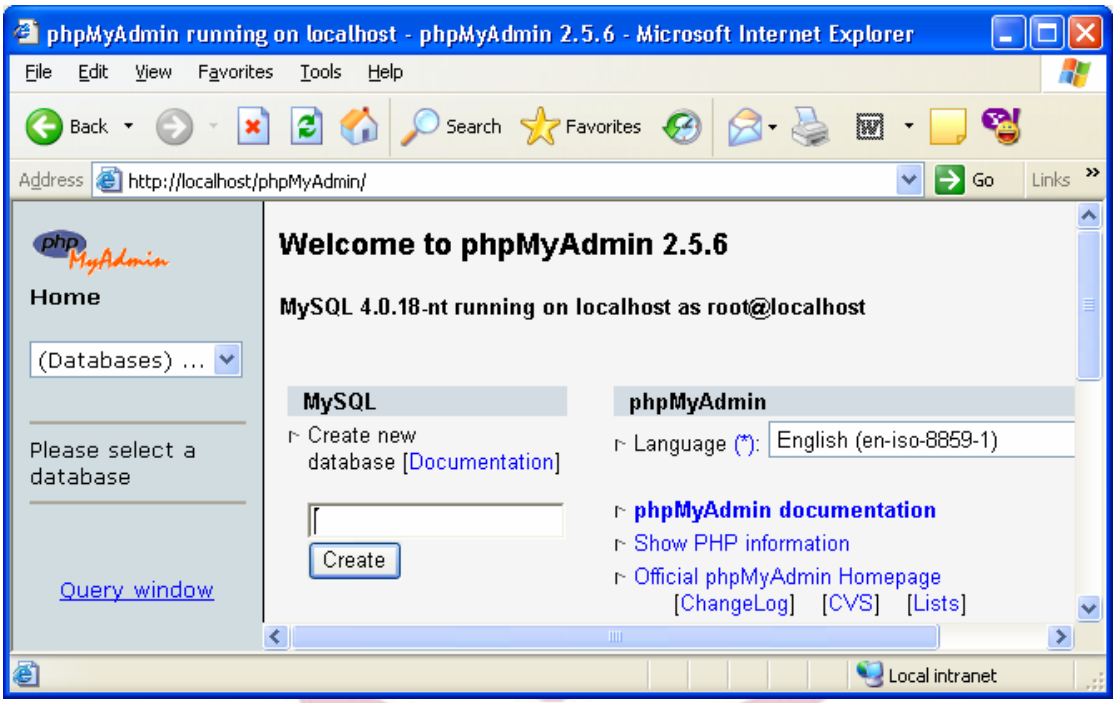
في تصميم قاعدة البيانات يجب علينا أن نقوم بأشياء كثيرة، أولها أن نقوم بتحليل ومعرفة ماذا نريد من هذه القاعدة، كيفية عملها، كيفية ترتيب المعلومات المفيدة لنا بالقاعدة كما يجب أن نقوم برسم القاعدة على الورق أولاً لكي نستنتج هل منطقنا صحيح في تصميم هذه القاعدة أم لا. ويجب أن نحل مشكلة التكرار في قاعدة البيانات..

هنا نقوم بتشغيل السيرفر الشخصي نذهب إلى المتصفح ونكتب التالي:

http://localhost/

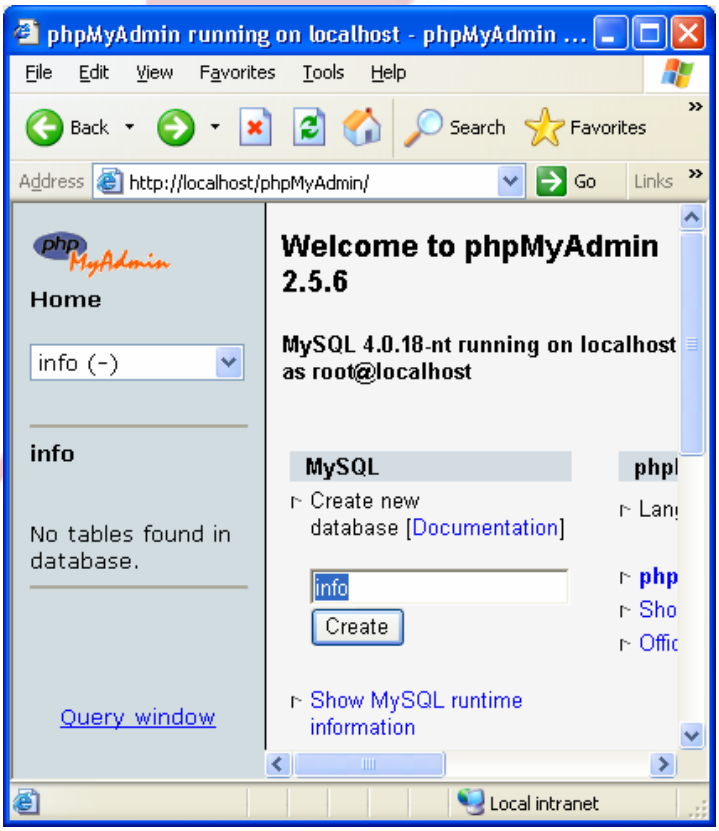
بعد أن تفتح الصفحة نضغط على الرابط التالي:

ستظهر الآن الصورة التالية وهي لبرنامج PHPMyAdmin وهو برنامج يتحكم في قاعدة البيانات ويسهل لك عملية الإنشاء والحذف وغيرها:



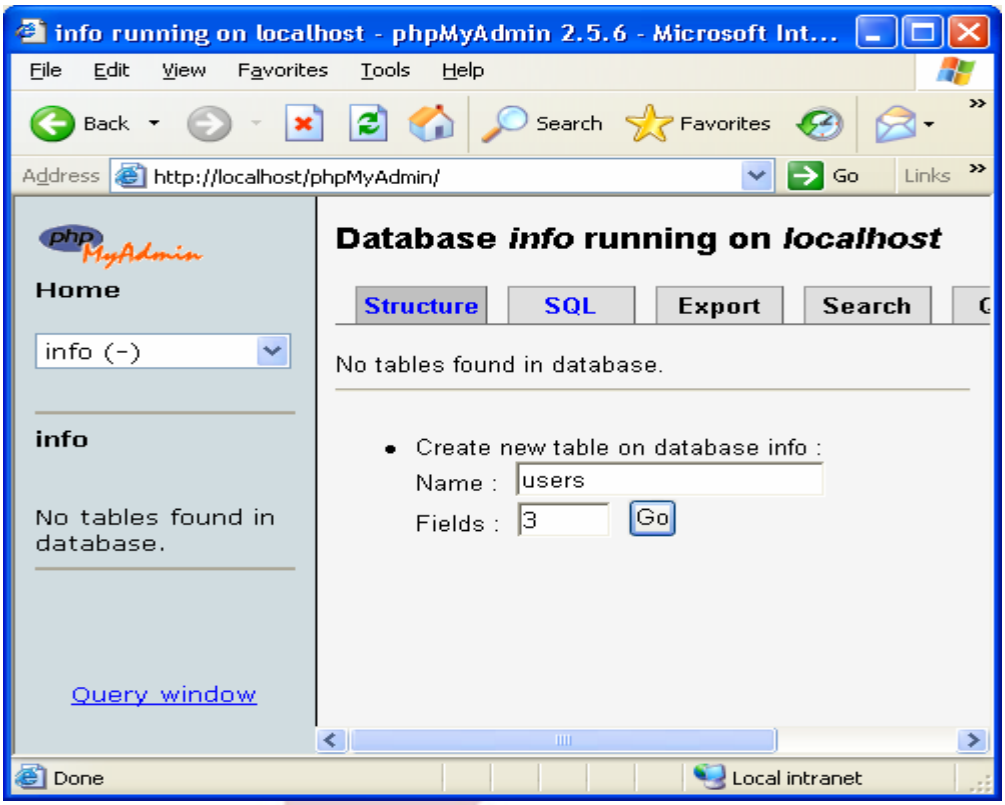
لنبدأ الآن بإنشاء قاعدة بيانات:

لنشئ قاعدة بيانات اسمها info باتباع الخطوات التالية:

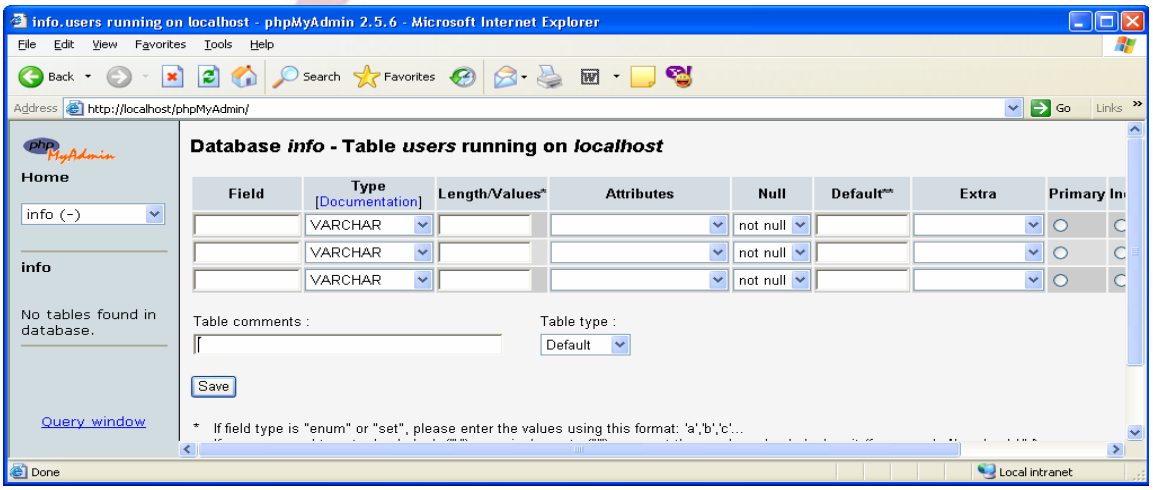


نكتب info في المكان المخصص ثم نضغط Create.

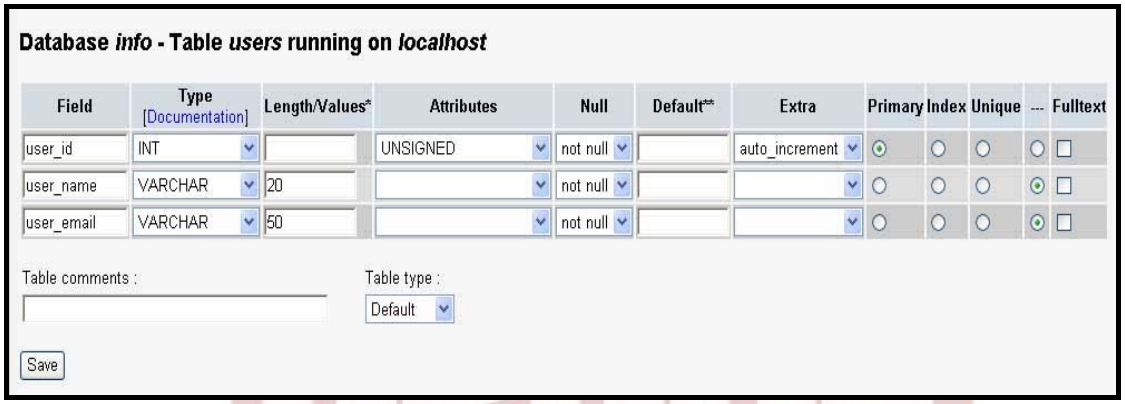
وسوف تنقل إلى الصفحة التالية ونحدد اسم الجدول وعدد الأعمدة به:



تحت Create new table on database info نقوم بإنشاء جدول في قاعدة البيانات ونسميه users ونضع به 3 حقول ولتكن (رقم المستخدم . اسم المستخدم . البريد الإلكتروني للمستخدم) نضغط الآن على الزر Go وسوف ترى التالي:

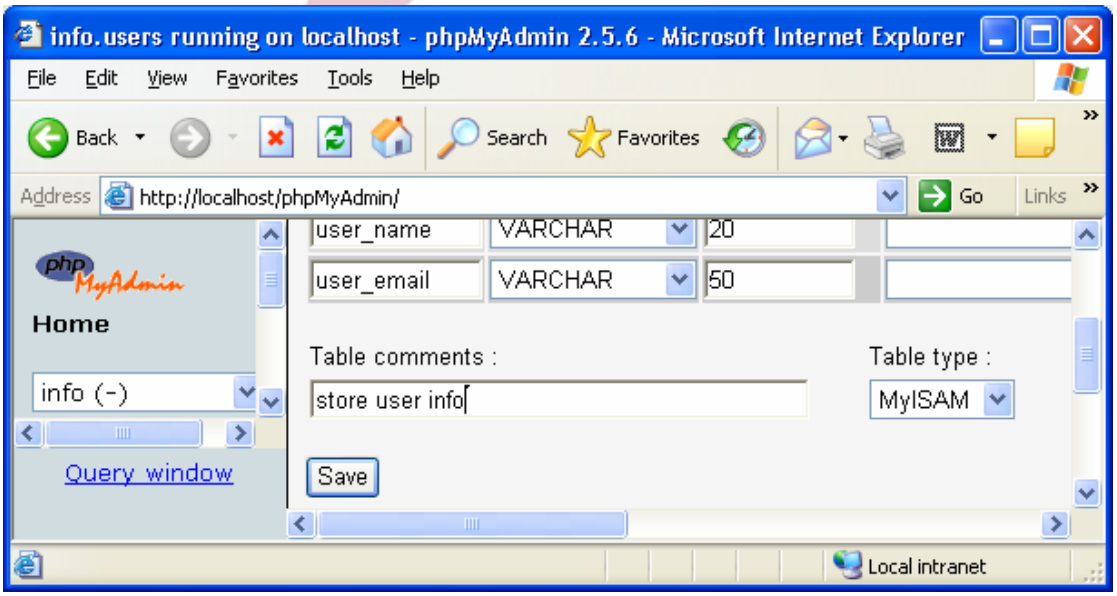


نقوم بملئها حسب الشكل أدناه:

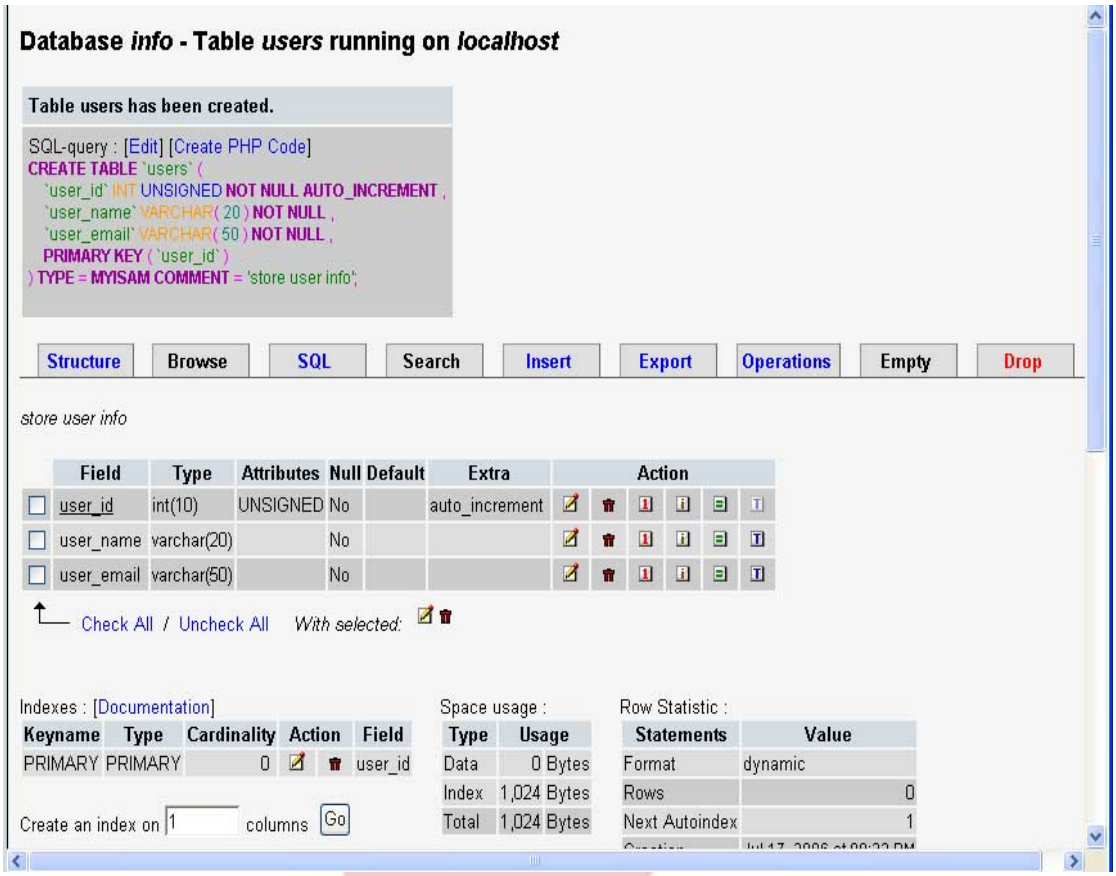


لاحظ أن أول حقل نجعل له الخصائص التالية unsigned أي لا يمكن أن يحتوي على عدد اقل من الصفر أو سالب auto increment أي يزيد لكل عضو يضاف جديد تلقائياً ويولد له رقم مستخدم فريد من نوعه وأخيراً الخاصية Primary أي يكون مفاتح رئيسي PRIMARY KEY

الحقلين الأخيرين نجعل لهما طول بالنسبة لعدد الأحرف. الأول 20 لنقل أن اسم المستخدم لن يزيد عن 20 حرف وبريده الإلكتروني عن 50 حرف وأخيراً نقوم بوضع اسم في Table comments ونحدد نوع الجدول ونضغط على save لاحظ الصورة:



إذا تم العمل بنجاح سوف ترى الصفحة التالية:



لاحظ أن الحقل user_id تحته خط وذلك يعني انه من النوع PRIMARY KEY

أرى من المفيد هنا أن نتعرف على أنواع بيانات الأعمدة في قواعد البيانات:

الوقت والتاريخ	الأنواع العشرية	الأنواع الرقمية
DATE	FLOAT	TINYINT
TIME	DOUBLE	SMALLINT
DATETIME	REAL	MEDUIMINT
TIMESTAMP	DECIMAL	INT
YEAR	NUMERIC	INTEGER
		BIGINT

الخيارات والتعددية	الكتابات و BLOB	النصوص العادية
ENUM	TINYBLOB	CHAR
SET	TINYTEXT	VARCHAR
	BLOB	
	TEXT	
	MEDUIMBLOB	
	MEDUIMTEXT	
	LONGBLOB	
	LONGTEXT	

تستخدم الأعمدة النصية لإدراج بيانات محرفية أو خليط من البيانات المحرفية والرقمية، و هناك عدة أنواع للأعمدة النصية:

:char

لهذا النوع من الأعمدة طول أعظمي يبلغ 255 محرف. ويكون استخدامه كالتالي:col_name char (size) وهذا النوع من الأعمدة هو عمود ذو طول ثابت أي إذا تم إدراج قيمة ما، عدد محارفها أقل من العدد الأعظمي للعمود، فسيتم حشو باقي الحقل بفراغات من اليمين.

عيب هذا النوع من الأعمدة هو أنه يقوم بحجز مساحات كبيرة في قاعدة البيانات و معظمها يكون عبارة عن فراغات.

أما أهميته فهو أنه مفيد لإنشاء أعمدة كلمات المرور Passwords .

:varchar

الطول الأعظمي: 255 محرف.

الاستخدام col_name varchar(size) :

ويتميز هذا النوع بأنه ذو طول متغير أي إذا قمت بتعريف عمود من النوع varchar(15) وخزنت القيمة Die Hard فيه فلن يتم حشر فراغات إلى يمين القيمة المخزنة بل يقوم بإزالة الفراغات من نهاية السلسلة المحرفية. و لكن MySQL تضيف محرفاً واحداً إلى كل عمود من النوع varchar حيث يتم فيه تخزين طول الحقل.

:Text

الطول الأعظمي هو : 65535

الاستخدام col_name text : أيضاً هو من النوع متغير الطول، يمكن أن يتم إنشاء فهارس على أول 255 محرف من العمود الذي من النوعtext كما يمكن أن يتم استخدام الفهارس من النوعFULLTEXT .

:enum

يستخدم هذا النوع من الأعمدة من أجل تحديد خيار واحد من بين عدة خيارات موجودة، ويسمح هذا النوع من الأعمدة باستخدام 65535 قيمة.

الاستخدام col_name enum('val_1' , 'val_2' , …) default 'val_1' :

هناك أنواع أخرى من الأعمدة النصية وهي:

tinytext , mediumtext , longtext , set

الأعمدة الرقمية:

يستخدم هذا النوع من أجل إدراج بيانات رقمية أو عددية, و أهم أنواعها:

:int/integer

الاستخدام [col_name integer(size) [zerofill] [unsigned] :

إن التعليمة الموضوعة داخل القوسين [ ] هي تعليمة اختيارية، ويستخدم هذا النوع من الأعمدة غالباً من أجل أعمدة الترقيم التلقائي.

:float

الاستخدام [col_name float(M,D) [zerofill] :

حيث M هي عدد الخانات التي سيتم حجزها، و D هو عدد الفواصل التي ستحدد من هذه الخانات.

وهذا النوع لا يمكن أن يكون بدون إشارة، ويستخدم كما يلي:

column_name float (7,3)

أي أن العمود الذي اسمه column_name نوعه float وعدد خاناته 7 ومنها 3 للفاصلة أي أن أكبر رقم يمكن أن يخزن به هو 9999.999

وهناك أنواع أخرى للأعمدة ذات النوع الرقمي وهي:

tinyint , mediumint , bigint , double , real , decimal

أعمدة التاريخ و الوقت:

و له عدة أنواع هي:

Date:

الاستخدام col_name date :

ويكون تخزين الوقت في هذا النوع من الأعمدة على الشكل التالي:

(YYYY-MM-DD) حيث أن القيم المسموح بها هي بين 1000-01-01 إلى 9999-12-31

datetime:

الاستخدام col_name datetime :

يكون شكل التنسيق هو(YYYY-MM-DD HH:MM:SS)

أما القيم المسموح بها فهي بين: 1000-01-01 00:00:00 و

9999-12-31 23:59:59

ومن سيئات هذين النوعين هو أنه ستقوم أنت بإضافة التاريخ لذلك فمن الأفضل استخدام النوع الثالث و هو:

timestamp:

الاستخدامcol_name(size)

هذا النوع من الأعمدة هو متعدد الاستخدامات كما أنه يقوم تلقائياً بتسجيل تاريخ

ووقت أحدث التغييرات سواء أكان هذا التغيير إدراجاً أو تحديثاً أما الوسيط size فهو يأخذ القيم الزوجية التي بين العددين 2 و 14 حيث يكون التنسيق كما يلي:

Size	Format
2	YY
4	YYMM
6	YYMMDD
8	YYYYMMDD
10	YYMMDDHHMM
12	YYMMDDHHMMSS
14	YYYYMMDDHHMMSS

وهناك نوعين آخرين من أنواع أعمدة التاريخ و الوقت و هما time , year.

أنواع الجداول:

هناك عدة أنواع للجداول في MySQL وهي:

MyISAM: وهو النوع الافتراضي الذي يضعه MySQL للجدول في حال عدم تعيينك لنوع الجدول الذي تقوم بإنشائهK وهذا النوع من الجداول سريع جداً ومستقر.

Heap: هذا النوع من الجداول تكون مقيمة في الذاكرة أي أنها غير مخزنة في أي مكان فيزيائي لذلك فإنها تتبخر في حال انقطاع التغذية و لكن كونها تتوضع على الذواكر فهي بغاية السرعة والفائدة منها هو إمكانية بناء جداول مؤقتة لتتوضع على الذواكر من أجل الاستعلامات السريعة.

و هناك أنواع أخرى هامة هي Gemini , BDB , InnoDB و لكل منها مزايا عديدة ومختلفة بحيث يتوقف النوع الذي ستختاره على نوع الوظيفة التي سيقوم بها الجدول الذي تقوم ببنائه. ولكنها غير مرغوبة وسنعتمد على أن كل جداولنا هي من النوع MyISAM .

الاستعلام في قواعد البيانات:

طريقة الاستعلام بلغة SQL وهي بسيطة جداً، وسوف تكون مختصرة لنرى مع بعضنا البعض الآن:

الأمر INSERT:

ونعني به الإضافة وهو كالتالي:

```
$sql = "INSERT INTO table_name ( v1 , v2 )
        VALUES ( 'va1' , 'va2' ) ";
```

المربع الأول نعني به اسم الجدول والمربع الثاني أسماء الأعمدة التي نريد إدراجها ولو كان العمود حقل يتم زيادته تلقائياً auto increment فلا نحتاج أن نضع له قيمة، فقط نقوم بوضع علامة التنصيص "

الأمر SELECT:

هذا الأمر يعني الاختيار (أي نستعلم عن معلومات معينه في قواعد البيانات) وهو كالتالي:

```
$sql = " SELECT * FROM table_name ";
```

نعني بالعبارة * أي كل شي ولو أردنا اختيار صف يحوي قيمه معينه نريد مطابقتها سوف نقوم بالتالي:

```
$sql = " SELECT * FROM table_name WHERE col_1 = 'col_value' ";
```

الأمر DELETE:

نعني بهذا الأمر المسح إذا أردنا أن نقوم بمسح صف فسوف نستخدم هذه العبارة كالتالي:

```
$sql = " DELETE FROM table_name WHERE col_name = 'value' ";
```

يوجد هناك أوامر أخرى كثيرة ولمعرفة المزيد عنها يمكنكم زيارة العنوان التالي:

http://www.mysql.com/

الاتصال بقاعدة البيانات في لغة PHP (الدالة mysql_connect و mysql_pconnect):

تقوم هذه الدوال بالاتصال مع قاعدة البيانات ومدخلاتها ثلاث أشياء اسم الخادم اسم المستخدم لقاعدة البيانات كلمة المرور لقاعدة البيانات بالترتيب لاحظ التالي:

```
$conn = mysql_connect( 'localhost' , 'root' , '' );
```

بالنسبة للدالة mysql_pconnect فهي تقوم بنفس العمل ولكن الاتصال بها لا ينقطع بانتهاء الاستعلام وإنما يبقى السكريبت متصلاً بقاعدة البيانات.

اختيار قاعدة البيانات (الدالة mysql_select_db):

تقوم هذه الدالة باختيار قاعدة البيانات التي تريد العمل عليها, المدخل الأول اسم قاعدة البيانات والمدخل الثاني هو حلقة الاتصال بقاعدة البيانات. وهي تعمل بالشكل التالي:

```
$conn = mysql_connect ('localhost','root',' ');
$db = mysql_select_db ('zzzz',$conn);
```

لاحظ انك لو استخدمت الدالة mysql_pconnect فلن تحتاج إلى ادخل حلقة الاتصال.

إرسال الاستعلام (الدالة mysql_query):

هذه الدالة الجميلة تساعدك على إرسال استعلام إلى قاعدة البيانات مدخلها الأول الاستعلام المطلوب والمدخل الثاني حلقة الاتصال لاحظ التالي:

```
$conn = mysql_connect ('localhost','root',' ');
$db = mysql_select_db ('zzzz' , $conn);
$sql = "select * from zzzz";
$result = mysql_query($sql , $conn);
```

استقبال الاستعلام وطباعته (الدالة mysql_fetch_array والدالة mysql_num_rows):

الدالة mysql_num_rows تقوم بأخبارك كم صف تم إرجاعه من الدالة mysql_query , لاحظ المثال التالي:

```
$conn = mysq1_connect ('localhost','root',' ');
$db = mysq1_select_db ('zzzz',$conn);
$sq1 = "select * from zzzz";
$result = mysq1_query($sq1 , $conn);
$number = mysq1_num_rows ( $result );
```

ولو أردنا أن نقوم بعرض النتائج فسوف نقوم باستخدام الدالة:

mysql_fetch_array وهذه الدالة تعطيها ناتج الاستعلام من الدالة mysql_query وهي تقوم بطباعته عن طريق أي دالة تكرار وتقوم الدالة بإرجاع الناتج على شكل مصفوفة حرفية مفتاحها key هو اسم العمود في قاعدة البيانات والقيمة value هو ما يقابل هذا العمود.

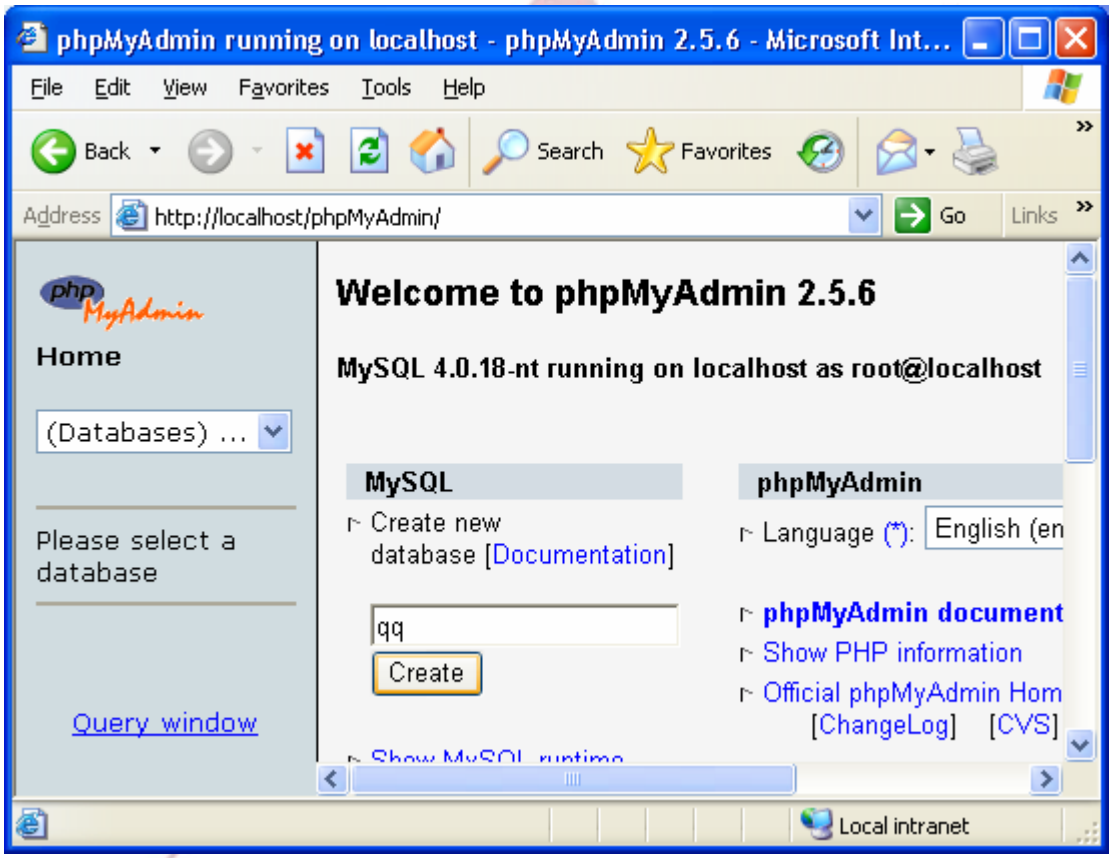
مثال عملي بسيط:

حساب لمدير أو مسؤول قاعدة البيانات:

تعلمنا فيما سبق كيف يمكننا إنشاء قاعدة البيانات من خلال **phpmyadmin** والآن سننشئ اسم القاعدة فقط وسنتمم إنشاء الجدول admin من خلال ملف config.php وملف install.php.

تعلمنا سابقاً...

إذن سننشئ قاعدة البيانات أولاً ومن ثم ملف الـ Config.



سنكتب الآن الكود الخاص بالملف config.php

```
<?
// المستضيف لا تعدل عليه
$db_host="localhost";
// للسيرفر المحلي لا تعدل عليه
$db_user="root";
// للسيرفر المحلي دعه فارغ
$db_pass="";
// اسم قاعدة البيانات اختياري
$db_name="qq";
$reslut_connect= mysql_connect("$db_host","$db_user","$db_pass")
or die ('هناك خطأ في بيانات الاتصال بقاعدة البيانات');
mysql_select_db ($db_name,$reslut_connect) or die
('هناك خطأ في بيانات الاتصال بقاعدة البيانات');
?>
```

اقرأ الكود بعناية وحلله... نفذ الكود السابق كما هو..

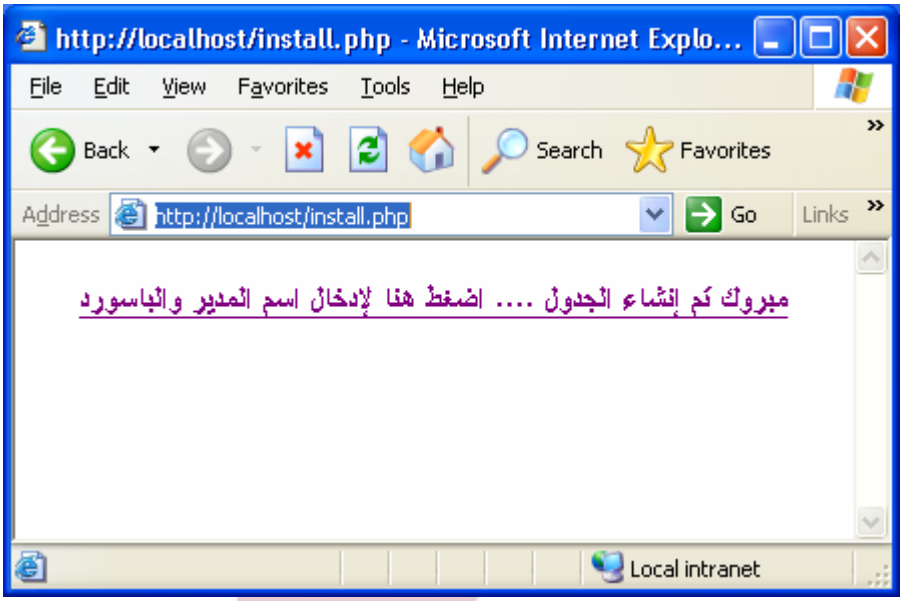
والآن نحن بحاجة إلى ملف تحميل لتثبيت الجدول السابق في قاعدة البيانات وهو: install.php.

ويكون الكود الخاص به على الشكل التالي:

```
<?php
include("config.php");
$abosaleh_admin_create = mysql_query("  CREATE TABLE `admin` (
  `user` varchar(250) NOT NULL default '',
  `password` varchar(250) NOT NULL default '',
  `id` int(11) NOT NULL auto_increment,
   PRIMARY KEY  (`id`)
  ) TYPE=MyISAM;");
// نضع هنا شرط للتأكد من اتمام انشاء الجدول او هناك مشكلة
if($abosaleh_admin_create){
Echo "</font><center><b><a href='create_login.php'> مبروك تم إنشاء الجدول
اضغط هنا لإدخال اسم المدير والباسورد ....</a> ";
}else{
Echo "<center><b>لم يتم انشاء الجدول تأكد من الكود الخاص بملف الـconfig";
}
?>
```

في حال كانت كافة الخطوات السابقة صحيحة سيظهر لنا التالي عن تشغيل ملف install.php. وإلا ستظهر رسالة الخطأ لم يتم إنشاء الجدول تأكد من الكود الخاص بملف الـ config .

اقرأ الكود بعناية وحلله...



لن نضغط على الرسالة أعلاه فنحن بحاجة إلى ملف يحتوي على فورم نقوم من خلاله بتسجيل اسم المدير وكلمة المرور الخاصة به... وذلك الملف هو: **create_login.php**

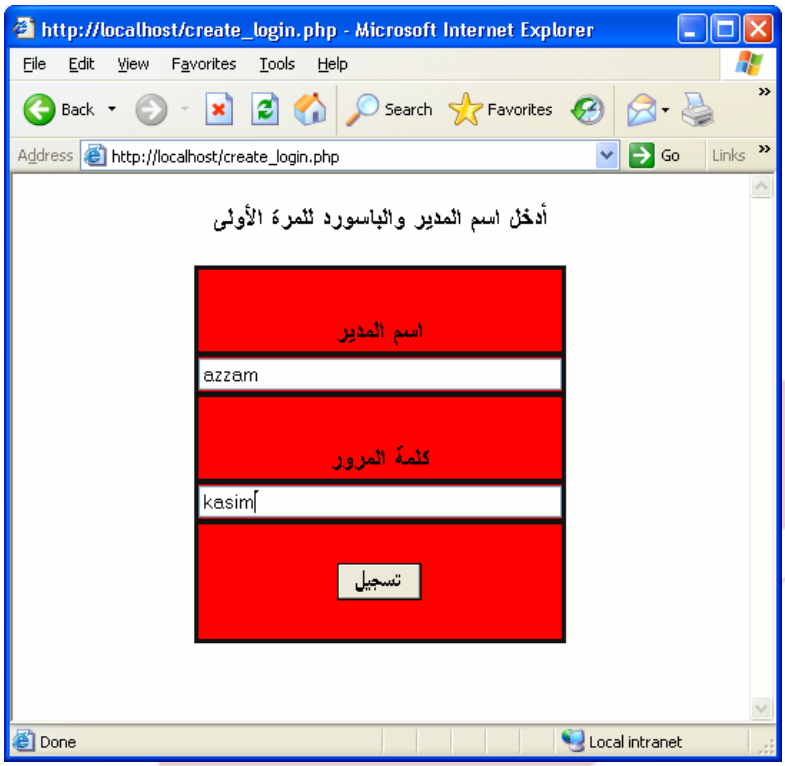
وذلك حسب الكود التالي:

```
<?
ECHO "<div align='center'>";
 <h3><b><p>أدخل اسم المدير والباسورد للمرة الأولى <br></p></h3></b>
 <table border='3' cellpadding='0' cellspacing='0' style='border-collapse: collapse; ' bordercolor='#111111' width='33%' id='AutoNumber1' bgcolor='red'>
   <tr><form method='post' action='login.php?Next=outinstall'>
    <td width='100%' align='center' style='border-style: 1; border-width: medium'>
    <br><h3><b>
    اسم المدير</td></h3></b>
```

```
  </tr>
  <tr>
    <td width='100%' align='center' style='border-style: 1; border-white: medium'>
    <span lang='en-us'><input type='text' name='AdminName' size='38'></span></td>
  </tr>
  <tr>
    <td width='100%' align='center' style='border-style: 1; border-width: medium'>
    <br><h3><b>
    كلمة المرور</td></h3></b>
  </tr>
  <tr>
    <td width='100%' align='center' style='border-style: 1; border-width: medium'><input type='text' name='AdminPass' size='38'></td>
  </tr>
  <tr>
    <td width='100%' align='center' style='border-style: none; border-width: medium'>
    <p dir='rtl'><span lang='en-us'>
    <br>
    <input type='submit' name='save' size='38' value=' تسجيل ' style='line-height: 150%; border-style: 2; border-width: 1'><br>
 </span></td>
  </tr>
 </table>
 </center>
</div></form>";
?>
```

اقرأ الكود بعناية وحلله...

في حال سلامة كافة الخطوات السابقة عن الضغط على الرابط في صفحة الملف install.php ستظهر لنا النتيجة التالية...



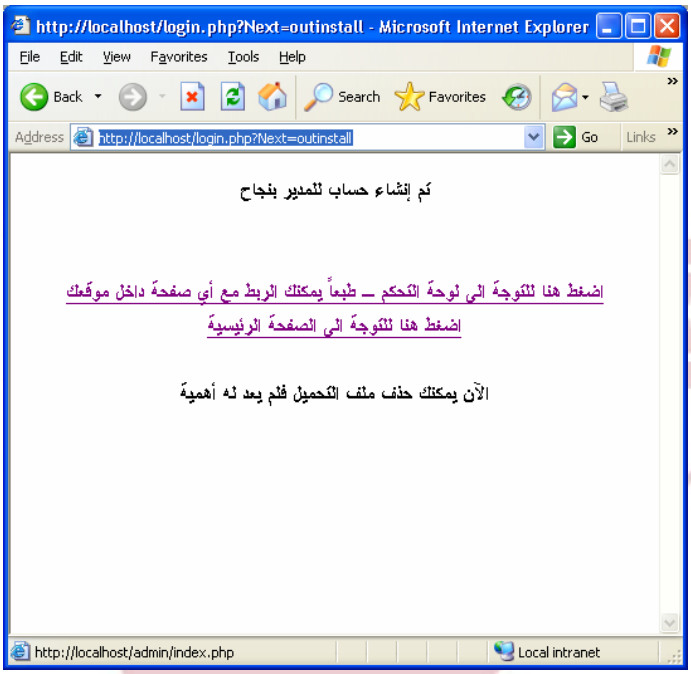
حتى الآن كل ما فعلناه هو إنشاء جدول في قاعدة بيانات وربطها بفورم مناسب لها ولكن الخطوة الأهم الآن هي كيف سيتم تسجيل الاسم وكلمة المرور في قاعدة البيانات (الجدول)... نحن هنا بالطبع بحاجة إلى داله مهمة من دوال الـ sql وتذكرها دائماً هي INSERT INTO وعملها إدخال البيانات المدرجة بين أقواس.

لذا نضع الكود التالي في ملف **login.php**

```
<?
require('./config.php');
$Re=mysql_query("INSERT INTO admin(user,password)"."VALUES
('$AdminName','$AdminPass')");
if($Re){
echo("<center><b> تم إنشاء حساب للمدير بنجاح<br><br></font><br>
    </font><a href='admin/index.php'> اضغط هنا للتوجة الى لوحة التحكم . طبعاً
يمكنك الربط مع أي صفحة داخل موقعك</a><br>
    <a href='index.php'>اضغط هنا للتوجة الى الصفحة الرئيسية</a><br><br>");
echo("<center><b>الآن يمكنك حذف ملف التحميل فلم يعد له أهمية <br><br>");
    }else{
echo("<br><br><cnter><b> لم يتم انشاء حساب للمدير <br><br>");
    }
?>
```

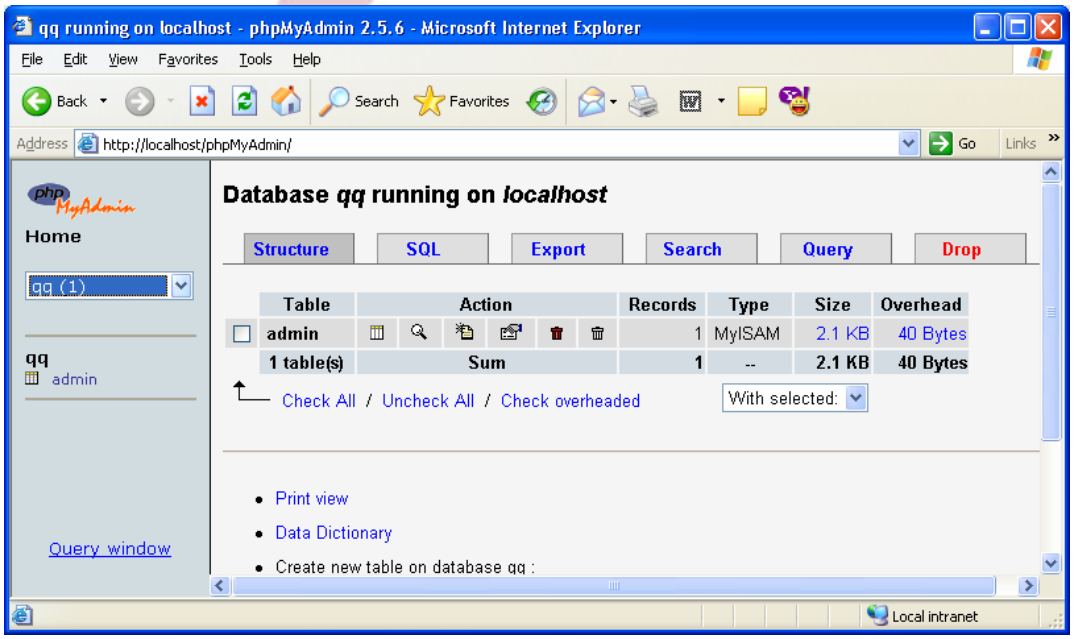
اقرأ الكود السابق بعناية وحلله...

اضغط تسجيل الآن في الصفحة السابقة وشاهد النتيجة....

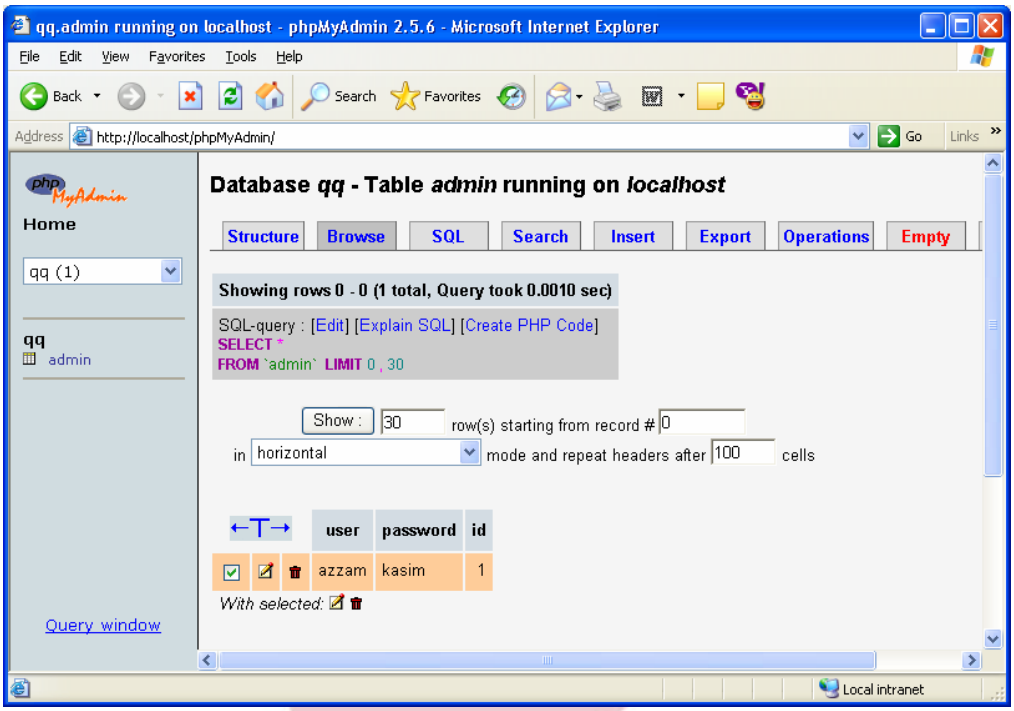


بقيت خطوة واحدة الآن وهي كيف سنتأكد من أن البيانات قد حُفظت في قاعدة البيانات والأهم هل تم فعلاً إنشاء الجدول المطلوب....

للتأكد من ذلك نعود ثانية إلى البرنامج **phpmyadmin** ونقوم بفتح قاعدة البيانات qq والتي سبق وأنشئناها... فسوف نشاهد....



لاحظ أن جدول admin قد أنشئ... استعرض الجدول وشاهد...



هل لاحظت الآن بأن الأعمدة التي طلبنا إنشائها موجودة وأن اسم المدير وكلمة المرور قد تم حفظها في قاعدة البيانات.

طبعاً التعامل مع قواعد البيانات رغم كونه ممتع إلا أنه بحرٌ لا قرار له وسوف أترك لك عزيزي الطالب هذه المتعة فكلما حاولت إنشاء قواعد بيانات وربطها كلما اقتربت من الاحتراف وهذا منوط بمدى تفهمك لما سبق فقد حاولت تبسيط الموضوع قدر الإمكان... والآن جاء دورك...

التعامل مع الملفات والمجلدات

سوف يستخدم برنامجك الملفات لكي يقوم بتخزين معلومات الإعداد للسكريبت، أو يقوم بتخزين البيانات لقراءتها وكتابتها، أو لكي يقوم بحفظ البيانات المؤقتة.

فتح وإغلاق الملفات:

Fopen: تستخدم هذه الدالة ثلاث عوامل هي مسار الملف (path) والوضع له (للقراءة، للكتابة .....) بالإضافة إلى المسار...

المتغير $fp الذي يخزن فيه مكان الملف وما إذا كان قابلاً للفتح أو لا أو يعمل أو لا يعمل، والنتيجة التي تتخزن في المتغير $fp هي رقم وهو صفر إذا كان الملف لا يعمل أو واحد إذا تم فتح الملف بنجاح..

إلا أنه يجب مراعاة القيم التالية عن وضع الكود..

القيمة	الوصف
r	تفتح الملف للقراءة فقط ويكون المؤشر في بداية الملف
r+	يفتح الملف للقراءة والكتابة ويضع المؤشر في بداية الملف
w	يفتح الملف للقراءة فقط، أي بيانات موجودة سيتم مسحها، إذا لم يكن الملف موجودا سيحاول PHP إنشاؤه
w+	يفتح الملف للقراءة والكتابة، أي بيانات موجودة سيتم مسحها، إذا لم يكن الملف موجودا سيحاول PHP إنشاؤه
a	يفتح الملف للإضافة فقط، إذا لم يكن الملف موجوداً سيحاول PHP إنشاؤه، سيكون المؤشر في نهاية الملف
a+	يفتح الملف للقراءة وللإضافة، إذا لم يكن الملف موجوداً سيحاول PHP إنشاؤه، سيكون المؤشر في نهاية الملف
b	يستخدم لفتح وقراءه ملفات الصور على نظام أو سيرفرات الويندوز فقط.. أما اليونكس فالعوامل السابقة تتعامل مع ملفات الصور بشكل عادي..

مثال بسيط:

سنحاول إنشاء ملف وورد مثلاً ...

```
<?
$fp= fopen("azzam.doc","w");
?>
```

وهكذا يمكننا إنشاء الملف الذي نريد...

أما إذا كان الملف قد تم إنشاءه مسبقاً ففي هذه الحالة سيتم حذف محتوياته ويصبح ملف فارغ جديد...

Fread: تقوم هذه الدالة بقراءة واستخراج البيانات الموجودة في الملفات ووضعها بمتغير وهي تأخذ معاملين المعامل الأول هو الملف والثاني هو عدد الحروف المراد قراءتها..

مثال:

```
$fp=fopen("data.txt","r");
$data=fread($fp,10);
```

Fgetc: تستخدم هذه الدالة لقراءة حرف واحد من الملف في كل مرة، وهي تستخدم معاملاً واحداً وتقوم بإرجاع حرف واحد من الملف أو (False) عند الوصول إلى نهاية الملف ..

Feof: تقوم هذه الدالة بمعرفة إذا ما كنا قد وصلنا إلى نهاية الملف عند قراءته وتقوم بإرجاع (true) عند الوصول إلى نهاية الملف أو حصول خطأ ما.

مثال:

أنشئ ملفاً وسمه file.txt واكتب فيه أكثر من سطر ثم قم بإنشاء ملف PHP وسمه بأي اسم وضع فيه الشفرة التالية:

```
<?
$fp= fopen("file.txt","r");
While (!feof($fp))
{
 $char=fgetc($fp);
    echo $char;
} ?>
```

Fgets: تساعدنا في قراءة عدد محدد من البايتات وهي تأخذ معاملين، المعامل الأول هو الملف والثاني هو عدد الحروف المراد قراءتها +1.

الأول : هو إذا تم قراءة عدد البايتات المحدد ة

Fputs: تقوم بنفس وظيفة الدالة fwrite وتأخذ نفس معاملاتها ونفس طريقتها ..

File: تحتاج هذه الدالة إلى معامل واحد هو مسار الملف وعملها هو قراءة ما بداخل الملف وتخزينه سطراً سطراً في مصفوفة حيث أن هذه المصفوفة تقوم بأخذ كل سطر في الملف كأنه عنصر لوحده وتظل السطور سطوراً (أي أن المصفوفة تحتفظ بالمعامل للسطر الجديد (\n) بداخلها) وهي تقوم بفتح وقراءة وإغلاق الملف تلقائيا ...

وكغيرها من الدوال فإنها تستطيع قراءة صفحات الإنترنت الخارجية ..

مع ذلك يستحسن أن لا تقوم باستعمال هذه الدالة لقراءة الملفات الطويلة لأنها تقوم باستخدام قدر كبير من الذاكرة المحجوزة للـ PHP وقد تستخدمها كلها ...

مثال:

```
<?
$fcontents = file ('file.txt');
while (list ($line_num, $line) = each ($fcontents)) {
   echo "<b>Line $line_num:</b>  $line  <br>\n";
}
?>
```

Fpassthru: تقوم هذه الدالة بقراءة محتويات الملف بداية من النقطة التي توقف منها المؤشر الوهمي عند أي عملية قراءه أخرى، وتقوم بالتوقف عند نهاية الملف وتقوم بإغلاق الملف من تلقاء نفسها لذلك لا داعي لإغلاق الملف بواسطة الدالة fclose بعد استخدامك لهذه الدالة، وتقوم الدالة بقراءة المحتويات وطباعتها بشكل قياسي.

مثال:

```
<?
$fp=fopen("file.txt","r");
fpassthru($fp)
?>
```

Readfile: تقوم هذه الدالة بقراءة جميع محتويات الملف وتحتاج إلى مسار الملف فقط وتقوم بقراءة كامل محتويات الملف ثم طباعتها بشكل قياسي وتقوم بإرجاع عدد البايتات التي تم قراءتها أو (false) عند حدوث خطأ ما

```
<?
Readfile ("file.txt");
?>
```

File_exists: تقوم هذه الدالة بالقيام بالتأكد ما إذا كان الملف موجوداً أم لا وهي تحتاج معامل واحد وهو مسار الملف، وتقوم بإرجاع true (1) إذا كان الملف موجوداً و false إذا كان الملف غير موجود:

```
<?
$Th=File_exists("file.txt");
echo $Th ;
?>
```

Filesize: تقوم هذه الدالة بإرجاع حجم الملف بالبايتات أو false عند حصول خطأ...

```
<?
$Th=Filesize("file.txt");
echo $Th ." "."Bit";
?>
```

Fileowner: تقوم بإرجاع رقم المعرف (ID) لمالك الملف ...

Filegroup: تقوم بإرجاع رقم المعرف (ID) لرقم المجموعة التي يعتبر مالك الملف ضمنهم ..

Filetype: تقوم بإرجاع رقم نوع الملف وقد تعود بإحدى هذه القيم (file ، dir ، char ،fifo ، link ،block) والذي يهمنا منهم هو file و dir ...

Is_dir: وتقوم بإرجاع True إذا كانت قيمه المسار هو مجلد ..

Is_file: وتقوم بإرجاع True إذا كانت قيمه المسار هو ملف ..

Copy (): تقوم بأخذ قيمتين حرفتين وتشير إلي مصدر الملف الرئيسي الذي يوجد فيه الملف والمصدر الهدف الذي سيتم نسخ الـPHP إليه ...

```
<?
if (!copy($file, $file.'.bak')) {
   print ("failed to copy $file...<br>\n");
}
?>
```

Rename: نستطيع الآن استخدام هذه الدالة لإعادة تسمية الملف وهي تحتاج إلى قيمتين حرفتين وهي المصدر الملف أو مكانه واسمه الرئيسي ثم الاسم الجديد الذي تريد إعادة التسمية به ..

```
<?
Rename ('file.txt','newfile.txt');
?>
```

مثلما تعاملنا مع الملفات في الـPHP فإننا نتعامل مع المجلدات ، فهناك دوال للمجلدات تتطلب مقبض المجلد ، وهناك دوال تحتاج فقط إلي القيمة الحرفية فقط وبدلاً من الإطالة دعنا نقوم بالدخول في الموضوع مباشرة

Opendir: تقوم بفتح المجلد وإعطائنا مقبض المجلد.

Closedir(): تقوم بإغلاق المجلد المفتوح وتحتاج فقط إلى مقبض المجلد.

Readdir: تقوم بقراءة المدخل الحالي للمجلد.

Rewindir: تقوم بإرجاع المدخل من الصفر.

Chdir: للانتقال إلى مجلد آخر، وتتطلب المسار للمجلد الذي تريد الانتقال إليه.

Rmdir: تقوم بمسح مجلد، ولكن يجب أن يكون المجلد خالياً من أي ملفات أو مجلدات، وتتطلب مسار المجلد الذي تريد مسحه.

Mkdir: تقوم بإنشاء مجلد جديد وتتطلب أن يكون هذا المجلد غير موجود مسبقاً وتحتاج إلى قيمتين وهما اسم المجلد الجديد مع مساره، والترخيص المطلوب له.

Dirname: تقوم بإعطائنا اسم المجلد الحالي الذي فيه الملف، وتحتاج إلى مسار الملف.

معالجة الأخطاء

إن مصطلح الـdebbug هو من المصطلحات الشائعة في عالم البرمجة، وهو يشير إلى كيفية إصلاح أخطاء البرنامج وتوقعها قبل حدوثها، هناك أنواع من الأخطاء تحدث بسبب المبرمج وهناك أنواع من الأخطاء تحصل بسبب المستخدم، في العادة يجب أن يكون المبرمج متآلفاً مع مصطلح تتبع الأخطاء وإصلاحها.

قد يكون من أهداف تتبع الأخطاء الحماية بقدر أهميه البرنامج الجاري العمل عليه أو الموقع فكلما كان الموقع مهماً كان وجوب حمايته أكبر.

رسائل الخطأ في الـPHP لها طريقتها وتقنيتها الخاصة التي تسير عليها فهي ليست مثل الجافا وليست مثل cgi فالـ PHP لا تقوم بإرسال الخطأ إلى السيرفر بل تقوم بكتابة رسالة خطأ في مكان الخطأ.

قد يكون هناك أخطاء يصعب تتبعها أو معرفة مكانها في الأصل، وقد يكون هذا بسبب أنك تستخدم الـPHP في صناعة موقع ديناميكي وتشرك معها الجافا سكريبت وتضع علامات التعليق الخاصة التي تقوم بإخفاء الأخطاء في الجافا مما قد يجعلك تشعر بالحيرة وتجن أين مكان الخطأ.

أنواع الأخطاء:

هناك أنواع من الأخطاء منها الإملائية (Syntax Error) ومنها المنطقية ومنها أخطاء تحدث في وقت التنفيذ:

ومثال الأخطاء الإملائية:

```
<?
Eco "1";
// من المفترض أن تكتب التالي:
Echo "1";
```

```
?>
```

هذا سيعطيك رسالة خطأ Parse error

ومن الأخطاء الإملائية نسيان الفاصلة المنقوطة (semi-colon) في نهاية الدالة:

```
<?
Echo "hello"
// من المفترض أن تكتب التالي:
Echo "hello";
?>
```

هنا سوف يعطيك الـPHP رسالة خطأ لكن العجيب أنه لن يعطيك إياها بشكل صحيح فرسالة الخطأ تشير إلى أن السطر الرابع يحتوي على الخطأ بينما الخطأ هو في السطر الثاني.

وهناك خطأ آخر يحصل بسبب نسيان الـbrace (وهي الأقواس):

```
<? Php
for ($loop = 0 ; $loop < 5 ; $loop ++ )
{
Echo “”;
?>
```

إذا كنت قد نسيت إغلاق القوس فهذا من الأخطاء الشائعة، والأخطاء الإملائية لا يمكن حصرها، إنها أشبه بقواعد اللغة، لكن أكثر الأخطاء الإملائية الشائعة في برامج الـPHP

1 – نسيان الأقواس . مثال:

```
<?
for ($loop = 0 ; $loop < 5 ; $loop ++ )
{
for ($loop1 = 0 ; $loop1 < 10 ; $loop1 ++ )
{
for ($loop = 0 ; $loop < 5 ; $loop ++ )
{
code ….
}
}
```

في المثال السابق ينقصنا قوس إغلاق التكرار الأخير (})

2 – نسيان الفاصلة المنقوطة . مثال:

```
<?
Echo 10
<?
```

3 – خطأ إملائي في اسم function . مثال:

```
<?
Htmlspecialchar($I);
?>
```

سيعطيك رسالة خطأ:

Fatal error: call to Undefined function: htmlspecialchar().

وتصحيحها أن تكون :

```
<?
Htmlspecialchars($I);
?>
```

4 – نسيان إغلاق النص . مثال:

```
<?
Echo “arabbuilder;
?>
```

نسيان الـ(") في نهاية الكلمة . وسيعطيك Parse error

الأخطاء المنطقية (Logical Errors):

إن الأخطاء المنطقية هي الأكثر صعوبة في التتبع فقد تجد برنامجك يعمل بشكل صحيح وبكل سلامة ولكنه عند نقطة ما لايتم تنفيذها كما تريد أنت، لنضرب مثلاً على خطأ منطقي بسيط جداً، لنفرض أنك قمت بعمل نموذج مكون من مربع نص وزر، عند ضغطك لهذا الزر فأنت تريد أن يتم كتابة كلمة كبير إذا كان الرقم أكبر من 30 وكلمة صغير إذا كان الرقم أصغر من 30 لنقم بكتابة الكود للمثال الأول:

```
<?
echo "ادخل عمرك:" ;
echo '<br>
<form method = "post" action = "age.php">
<input type= "text" name = "age">
<br>
<input type= submit value = "هل أنا كبير أم صغير ؟" >
</form>' ;
?>
```

في ملف age.php اكتب الكود التالي:

```
<?
If ($age<30) echo "أنت صغير";
If ($age>30) echo "أنت كبير";
?>
```

سيعمل السكريبت بشكل صحيح .. ولكن ربما تخطأ أنت في كتابة العلامات المنطقية (التي باللون الأحمر) فتأتي النتائج بشكل خاطئ .

ومن الأخطاء المنطقية الأخطاء التي تقع في وقت التشغيل (Run times error) والتي تكون قد تقوم بإيقاف برنامجك بشكل كامل.

مثال:

```
<?
$t=0;
$r=1;
$f=$r/$t;
?>
```

وعندها سينتج لك الرسالة التالية

Warning: Division by zero in (path) on line (line number)

هناك نوع آخر من الأخطاء المنطقية (unexpected) وهو لا يقوم بإيقاف البرنامج نهائياً بل يقوم بإخراج رسالة الخطأ في مكان الخطأ أو قد يقوم بتنفيذ البرنامج وإخراج البيانات بشكل غير صحيح أو قد لا يقوم بإخراج بيانات.

أخطاء التكرارات:

قد يكون لديك أيضا تكرار فيه خطأ ولا يقوم بالتوقف نهائياً مثل هذا التكرار:

```
$c=1;
$t=true;
while ($t=true)
{
$c++;
}
```

لم نقم بعمل شي يوقف التكرار مثل أن تضع شرط يختبر قيمة المتغير ($c) ثم يقوم بإيقافه عند تعديه رقم معين وعلى ذلك فإن التكرار سيستمر بشكل غير متوقف ولن يعمل البرنامج.

عدم إرجاع قيمه من function:

مثال:

```
<?
Function ($d)
{
$d =$d+$d;
}
```

الخطأ هنا أننا لم نستخدم الـreturn لكي ننهي الدالة أو قد تكون الدالة تحتوي على أكثر من قيمة ولم نقوم بتحديد القيمة النهائية للدالة.

الخلط في المعاملات الحسابية والمنطقية:

مثال:

```
If ($y=10) echo 12 ;
```

والمفترض أن تكون:

```
If ($y= =10) echo 12 ;
```

لتفادي الأخطاء:

التعليقات: إن من الأفكار الجيدة للتقليل من الأماكن التي تبحث فيها عن الخطأ هو وضع تعليقات لوصف وظيفة دالة معينة . مثال:

```
<?
// هذه الكود يقوم بطباعة كلمة أحمد
Echo "أحمد" ;
?>
```

الدوال: وأيضاً من الأفكار الجيدة أن تقوم بتقسيم وظائف البرنامج على دوال بحيث أن لكل دالة وظيفتها المعينة:

```
<?
/*
      +- ----------------------------------------------------------+
      |      هذه الدالة تقوم بقسمه العدد على 2          |
      +- ----------------------------------------------------------+
* /
function ($U)
{
$U=$U/2;
return $U ;
}
?>
```

Regular Expressions

هذه التقنية تساعدك على تفادي الأخطاء في صفحتك عند حدوثه مثل أن يقوم مستخدم ما بكتابة بريد إلكتروني غير صحيح (مثال: a@y@.k.d) هذا البريد غير صحيح ولأجل أن تقوم بمنع حصول أي خطأ مثل ذلك وتقييد العبارات التي يدخلها المستخدم فإنك تقوم باستخدام الـ RE/ (Regular Expressions) إنك بالأصح تجعل قواعد للكلمات التي يدخلها المستخدم فمثلاً تجعل المستخدم لا يدخل سوي

أرقام أو حروف فقط أو شكل معين من الكلمات، تقوم أولاً بإنشاء نمط للكلمة التي تريد المستخدم أن يقوم بإدخالها.

استخدام عبارة echo:

هو من أقدم الأساليب وكان يستخدم مثلاً في فحص بعض متغيرات نموذج فمثلاً أنت لديك نموذج يقوم بإرسال معلومات إلى النموذج وقد تستخدم في اختبار الأخطاء المنطقية التي يستصعب متابعتها في الكود. مثال:

```
<?
Echo “this is: $name”;
Echo “<br>”;
Echo “this is: $Email”;
//كود يقوم بمعالجة معلومات المتغيرين
//طباعة المتغيرين بعد اداء عملية المعالجة ورؤية النتائج
Echo “this is after: $name”;
Echo “<br>”;
Echo “this is after: $Email”;
?>
```

فحص كود الـhtml: قد تستخدم كود جافا سكريبت ويتم إخفاء الأخطاء وسط علامات التعليقات فعليك حينئذ فحص كود الـhtml لرؤية إن كان هناك بعض الأخطاء المخفية أم لا.

تجاهل الأخطاء:

لنفترض أنك تعلم أن الدالة التي صنعتها بها أخطاء ولكنك تريد تجاهل هذه الأخطاء فكل ما عليك أن تقوم بوضع @ أمام الدالة لكي يتم تجاهل الخطأ عند حدوثه.

مثلاً نحن نعلم أن القسمة على الصفر من الأشياء الغير مقبولة في الـPHP وأنت صنعت دالة تقوم بالقسمة على صفر ولن يتم تنفيذها لأنها بالأصل خطأ ولكنك تريد أن يقوم PHP بتجاهلها فكل ما عليك أن تفعله هو وضع @ أمام الدالة.

مثال:

```
<?
function amail ($y)
{
$y=$y/0;
return $y;
}
$s= @amail(44);
echo $s;
?>
```

أمثلة متنوعة

إنشاء كود عشوائي . رموز تأكيد التسجيل:

أثناء عملية التسجيل في بعض المواقع تمر بمرحلة الصورة العشوائية والتي تحتوى على أرقام وحروف يجب عليك إدخالها كما في الصورة حتى تتم عملية التسجيل.

فكيف يتم إنشاء مثل هذه الصورة التي يتم انتقاء الأحرف والأرقام فيها بصورة عشوائية؟ وكيف يتم تمييز الحروف التي تظهر على الصورة بينما تقوم أنت بإدخالها من لوحة المفاتيح ؟

الآن سوف نشرح كيفية عمل ذلك بواسطة php وشرح الأكواد خطوة خطوة.

- تكوين الصورة، ويكون ذلك كالتالي:

```
Header("Content-Type: image/png");
```

- الآن نضع كود الجلسة كما يلي:

```
session_start();
$new_string;
session_register('new_string');
```

- الكود الخاص بإنشاء الصورة :

```
$img = ImageCreate(200, 40);
$white = ImageColorAllocate($img, 255, 255, 210);
$black = ImageColorAllocate($img, 0, 100, 222);
```

وضعنا المتغير img كرمز للتعبير عن الصورة، وحددنا أبعاد الصورة باستخدام الدالة image create بالطبع يمكن تغيير الأبعاد الخاصة بها، واستخدمنا لونين مختلفين ورمزنا لهما بالأبيض والأسود إلا أنه يمكن تغيير الألوان بتغيير قيم الأرقام في الكود .

ملاحظة: الأبيض: 255,255,255 الأسود: 0,0,0

- ننتقل الآن إلى مرحلة هامة جداً وأساسية وهي عمل الكود العشوائي للأرقام والحروف التي ستظهر على الصورة باستخدام نظام md5 للتشفير والدالة srand المسؤولة عن توليد الأرقام والأحرف العشوائية.

```
srand((double)microtime()*1000000);
```

- تشفير الكود وتحويله لأرقام وحروف.

```
$string = md5(rand(0,9999));
```

نحدد ثلاث خيارات: المتغير . عدد الخانات التي يتم جلبها من المتغير . عدد الحروف والأرقام . التي ستظهر داخل المتغير الجديد وهي هنا رقم 8.

```
$new_string = substr($string, 17, 8);
```

- ننظم الألوان والأكود داخل الصورة:

```
ImageFill($im, 0, 0, $black);
```

- نحدد مكان ظهور الأرقام على الصورة. إن أبعاد الصورة هي 200×40، نحدد ونجرب ما نراه مناسباً مثلاً:

```
ImageString($img, 5, 65, 10, $new_string, $white);
```

المتغير الأول هو الصورة img.

المتغير الثاني هو حجم الخط يمكن استخدام أي رقم من 1 إلى 5.

80 و 10 احداثي نقطة بداية الكتابه على الصورة ،

$new_string الكود الذي قمنا بتشفيره وجلبة من دالة التشفير.

$white لون خط الكتابة وهو هنا اللون الأبيض.

لاحظ أننا حددنا الألوان أكثر من مرة...؟؟

- **التجهيز النهائي للصورة:**

بقي أن نعطي الصورة اسم يمكن استخدامه في أي مكان في الصفحة مثلها مثل أي صورة عادية:

```
ImagePNG($im, "lolo.png");
ImageDestroy($img);
```

أعطينا الصورة اسم verify.png ، فقط قمنا بتحويل كافة المعلومات التي قمنا بإدخالها للمتغير img إلى صورة وسميناها اسم جديد، ثم استخدمنا دالة image destroy، وهذه الخطوة مهمتها تحرير ذاكرة السيرفر من البيانات المدخلة.

- نقوم الآن بترتيب الأكواد السابقة حسب التسلسل على الشكل التالي:

```
<?php
Header("Content-Type: image/png");
session_start();
$new_string;
session_register('new_string');
echo "<html><head><title>lolo</title></head>";
echo "<body>";
ECHO "<div align='center'>";
$img = ImageCreate(200, 40);
$white = ImageColorAllocate($img, 255, 255, 255);
$black = ImageColorAllocate($img, 0, 0, 0);
srand((double)microtime()*1000000);
$string = md5(rand(0,9999));
$new_string = substr($string, 20, 8);
ImageFill($img, 0, 0, $black);
ImageString($img, 5, 65, 10, $new_string, $white);
ImagePNG($img, "lolo.png");
ImageDestroy($img);
echo "<img src=\"lolo.png\">";
echo "<br><br>";
echo "<b>"." ادخل الأرقام والأحرف التي تشاهدها في الصورة وانتبه إلى وضعية الأحرف
الكبيرة او الصغيرة "."</b>";
echo " <form action=\"handler.php\"
method=post>";
echo "<input name=\"random\"
type=\"text\" value=\"\">";
echo "<input type=\"submit\">";
echo "</form>";
echo "</body>";
echo "</html>";
?>
```

الكود باللون الأخضر تمت إضافة لأننا سننشئ ملف جديد لنستطيع من خلاله التأكد من أن ما كتبه المستخدم مطابق للصورة أم لا..

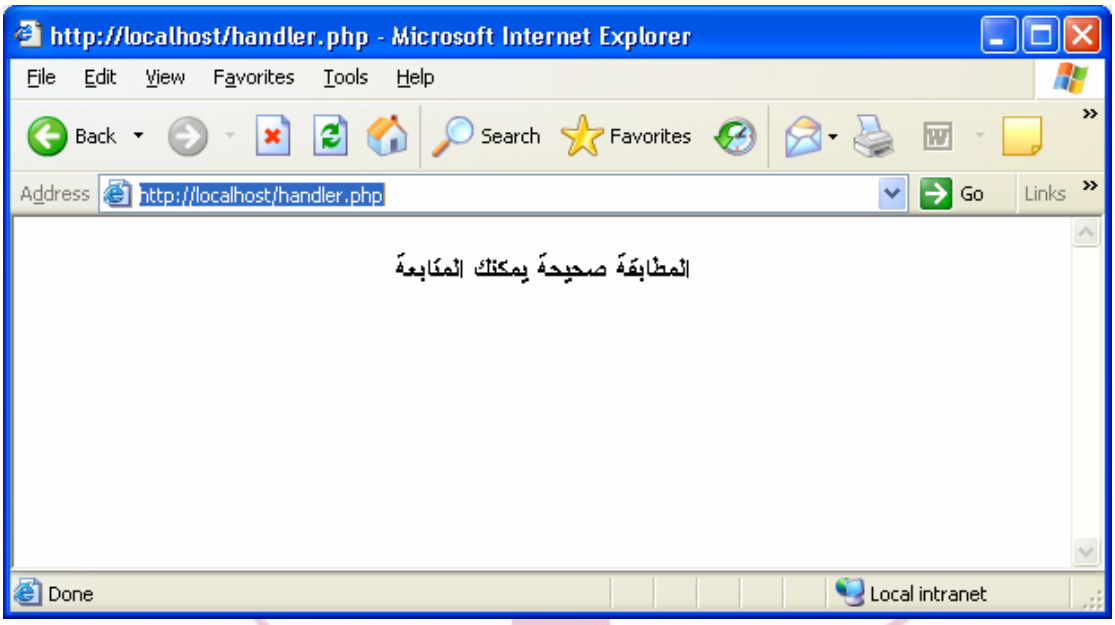
الملف الجديد اسمه handler.php ونضمنه الكود التالي:

وهو كود لجملة شريطة بسيطة سبق وأن تعاملنا معها.

```
<?
ECHO "<div align='center'>";
session_start();
$random = trim($random);
if ($new_string == $random)
{
echo "<b>"." المطابقة صحيحة يمكنك المتابعة ";
}
else{
echo "<b>"."لا يوجد مطابقة يرجى الرجوع واعادة المحاولة";
}
?>
```

جرب الآن الملف الأول:

في حال المطابقة ستظهر الرسالة التالية:



أما في حال عدم وجود مطابقة فستظهر الرسالة التالية:

معرفة رقم الـ IP:

```
<?
$IP = gethostbyname("www.yahoo.com");
echo $IP;
?>
```

تحويل التاريخ من ميلادي إلى هجري:

```
<HTML DIR=RTL>
<?
function hijri($GetDateFormat,$DFormat)
{
   //start function
   $Days=@date("D");   //print day name+Saturday-->Friday
   //start hijri function date
   $TDays=round(strtotime($GetDateFormat)/(3600*24));
   $HYear=round($TDays/354.3667);
   $Remain=$TDays-($HYear*354.3667);
   $HMonths=round($Remain/29.5305);
   $HDays=$Remain-($HMonths*29.5305);
   $HYear=$HYear+1389;
   $HMonths=$HMonths+10;
   $HDays=$HDays+23;
   //hijri function days between [29:30]
   if ($HDays>29.5305 and round($HDays)!=30)
   {
      $HMonths=$HMonths+1;
      $HDays=Round($HDays-29.5305);
   }
   else
   {
      $HDays=Round($HDays);
   }
   //hijri function months
   if ($HMonths>12)
   {
      $HMonths=$HMonths-12;
      $HYear=$HYear+1;
   }
   //hijri month names [print month name]
   if ($HMonths=="1")  $hmname="محرم";
```

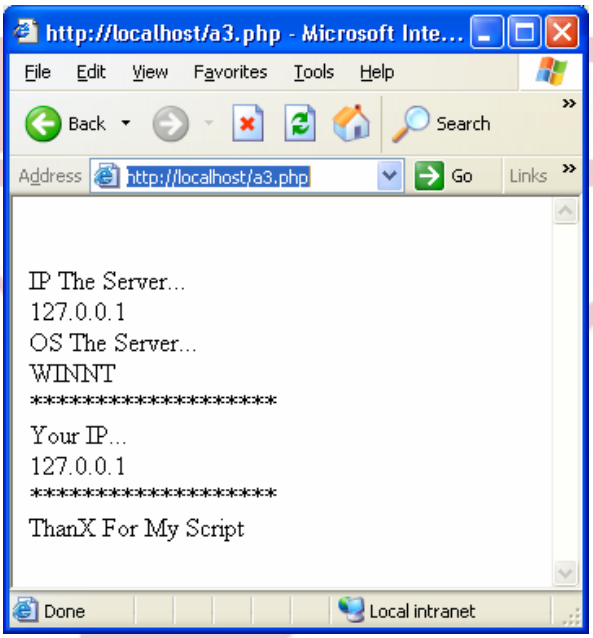
```
if ($HMonths=="2")  $hmname="صفر";
if ($HMonths=="3")  $hmname="ربيع الأول";
if ($HMonths=="4")  $hmname="ربيع الثاني";
if ($HMonths=="5")  $hmname="جمادى الأولى";
if ($HMonths=="6")  $hmname="جمادى الثانية";
if ($HMonths=="7")  $hmname="رجب";
if ($HMonths=="8")  $hmname="شعبان";
if ($HMonths=="9")  $hmname="رمضان";
if ($HMonths=="10") $hmname="شوال";
if ($HMonths=="11") $hmname="ذو القعدة";
if ($HMonths=="12") $hmname="ذو الحجة";
//day function [print day name]
if ($Days=="Sat")   $dd="السبت";
if ($Days=="Sun")   $dd="الأحد";
if ($Days=="Mon")   $dd="الاثنين";
if ($Days=="Tue")   $dd="الثلاثاء";
if ($Days=="Wed")   $dd="الأربعاء";
if ($Days=="Thu")   $dd="الخميس";
if ($Days=="Fri")   $dd="الجمعة";
$les = strlen($DFormat);
for($i=0; $i<=$les; $i++)
{
   $df[$i]= substr ($DFormat,$i,1);
   if($df[$i]=="A" || $df[$i]=="a")
   {
      $ddf=@date("a",$GetDateFormat);
      if(substr($ddf,0,1)=="a")
      {
         $Result.="صباحاً";
      }
      else
      {
         $Result>="مساءً";
      }
   }
   elseif($df[$i]=="D")  {$Result.="$dd";}
   elseif($df[$i]=="d")  {$Result.="$HDays";}
   elseif($df[$i]=="m")  {$Result.="$HMonths";}
   elseif($df[$i]=="M")  {$Result.="$hmname";}
   elseif($df[$i]=="y")  {$Result.="$HYear";}
   elseif($df[$i]=="Y")  {$Result.="$HYear"."هجري";}
   elseif($df[$i]=="g")  {$Result.=@date("g",$GetDateFormat);}
```

```
        elseif($df[$i]=="G")  {$Result.=@date("G",$GetDateFormat);}
        elseif($df[$i]=="i")  {$Result.=@date("i",$GetDateFormat);}
        elseif($df[$i]=="H")  {$Result.=@date("H",$GetDateFormat);}
        elseif($df[$i]=="h")  {$Result.=@date("i",$GetDateFormat);}
        elseif($df[$i]=="s")  {$Result.=@date("s",$GetDateFormat);}
        else
        {
           $Result.=$df[$i];
        }
    }
    return $Result;
    //end hijri function date
}
echo hijri(2004-04-28,"d M Y")."<BR>";
?>
```

معرفة معلومات السيرفر:

```
<?
$s=getenv ("SERVER_ADDR");
$os=PHP_OS;
echo"<br>";
echo "IP The Server...";
echo"<br>";
echo $s;
echo"<br>";
echo "OS The Server...";
echo"<br>";
echo $os;
echo"<br>";
echo "******************";
echo"<br>";
//Clint Info.
$ip = getenv ("REMOTE_ADDR");
```

```
echo "Your IP...";
echo"<br>";
echo $ip;
echo"<br>";
echo "*****************";
echo"<br>";
echo "ThanX For My Script";
?>
```



التعامل مع التاريخ:

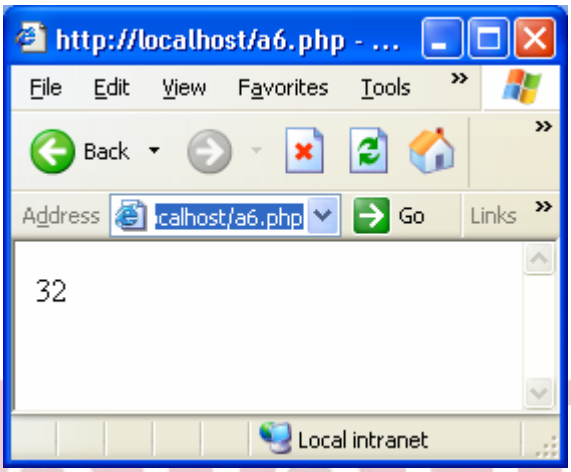
```
<html dir="rtl"> <?
// تاريخ اليوم
$CurrentDate=date ("Y-m-d");
echo 'تاريخ اليوم'." ". $CurrentDate;
// التاريخ منذ أسبوع
$FromOneWeekDate=date ("Y-m-d",mktime
(date("G"),date("i"),date("s"),date("m"),date("d")-7,date("Y")));
echo "<br>";
echo 'التاريخ منذ أسبوع'." ". $FromOneWeekDate;
// التاريخ منذ شهر
$FromOneMonthDate=date ("Y-m-d",mktime
(date("G"),date("i"),date("s"),date("m")-1,date("d"),date("Y")));
echo "<br>";
echo 'التاريخ منذ شهر' . " " . $FromOneMonthDate;
```

```
// التاريخ منذ سنة
$FromYearDate=date ("Y-m-d",mktime (date("G"),date("i"),date("s"),date("m"),date("d"),date("Y")-1));
echo "<br>";
echo 'التاريخ منذ سنة'." " . $FromYearDate;
// التاريخ غدا
$TomrrowDate=date ("Y-m-d",mktime (date("G"),date("i"),date("s"),date("m"),date("d")+1,date("Y")));
echo "<br>";
echo 'التاريخ غدا' . " " .$TomrrowDate; ?>
```

ستظهر النتيجة على الشكل التالي:

حساب الفرق بين تاريخين بالأيام:

```
<?
$FromDate="2005-12-30";
$ToDate="2006-01-31";
$Difference = strtotime("$ToDate") - strtotime($FromDate);
$Days=ceil ($Difference / (60*60*24));
Echo $Days;
?>
```



التحقق من العنوان البريدي:

```
<?
$Email="azzam@mail.sy";
$RightEmail=eregi("^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[_a-z0-9-]+(\.[_a-z0-9-]+)",$Email);
if ($RightEmail==1)
{
Echo "البريد الإلكتروني صحيح";
}
Else
{
Echo "البريد الإلكتروني غير صحيح";
}
?>
```

حاول وضع عنوان خاطئ وشاهد النتيجة.

تمارين...

صمم برنامجاً اختيارياً يحتوي على قاعدة بيانات على ألا يقل عن 4 صفحات مترابطة.. في أي موضوع تختاره على أن يتضمن معظم ما سبق وتحدثنا عنه.. ثم أنشئ ملفاً منفصلاً تشرح فيه الكود لكامل البرنامج بشكل دقيق وكامل.. وأرسل كافة الملفات في مجلد واحد.

مع تمنياتنا لكم بالتوفيق والنجاح

الدائمين