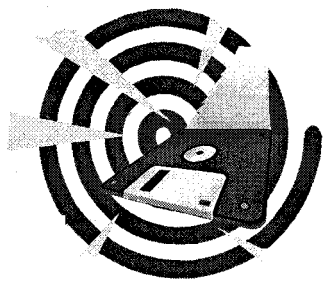
تطوير صفحات ويب
وبرمجة الإنترنت



لغة PHP

مهندس
عبد الحميد بسيوني

رقـم الإيداع بدار الكتب: 2004/15711
الترقيـــم الدولـــــى : 977-287-459-8

©حقوق النشر والطبع والتوزيع محفوظة لدار الكتب العلمية للنشر والتوزيع - 2004

لا يجوز نشر جزء من هذا الكتاب أو إعادة طبعه أو اختصاره بقصد الطباعة أو اختزان مادته العلمية أو نقله بأى طريقة سواء كانت إلكترونية أو ميكانيكية أو بالتصوير أو خلاف ذلك دون موافقة خطيه من الناشر مقدماً .

دار الكتب العلمية للنشر والتوزيع
50 شارع الشيخ ريحان - عابدين - القاهرة
☎ 7954229

لمزيد من المعلومات يرجى زيارة موقعنا على الأنترنت
wwwsbheg.com
e-mail: sbh@link.net

بسم الله الرحمن الرحيم

مُقَدِّمَة

بســـم الله الرحمن الرحيم والحمد لله رب العالمين ، والصلاة والسلام على سيد الخلق وخاتم المرسلين المبعوث رحمة للعالمين ، وبعد

لغة PHP هى لغة برمجة أدوات تطوير صفحات رئيسية شخصية Personal Home Page Tools مجانـــية مفـــتوحة لها برمجيات مفتوحة للاستخدام المجانى على شبكة الإنترنـــت ، وتوفـــر اللغـــة دعما لكافة قواعد البيانات مع دعم قوى لقواعد البيانات MySQL المجانـــية المـــتوفرة فـــى غالبية مواقع استضافة مواقع ويب ، وهى أدوات مستقرة .

لغة بى اتش بى لغة نصوص برمجية جهة الخادم server side scripting language تشبه لغة سى وجافا وبيرل لكنها أبسط تستخدم لتضمين شفرات برمجية يمكنها تنفيذ العديـــد مـــن المهـــام فى صفحات لغة النص المتشعب HTML ، وقد نمت هذه اللغة بمجهودات فردية لتصبح من أقوى لغات برمجة صفحات ويب .

تتمـــيز PHP عـــن لغات أخرى مثل JavaScript التى يتم تفسيرها من جهة الزبون client-side أن الشـــفرة يتم تنفيذها على المزود ، ويتلقى الزبون نتائج تشغيل النص البرمجى (سكريبت) بدون تحديد طبيعة الشفرة المتضمنة ، حتى أنه يمكن تهيئة مزود ويب لمعالجة ملفات HTML تزامنا مع PHP دون أن يعرف المستخدم ما تم إدراجه .

أفضـــل حســـنات استعمال PHP توافر عدة ميزات متقدمة ، وبهذه اللغة PHP يمكن ترميز كل شىء لبرمجة صفحات ويب يتم تنفيذها من جهة المزود بحيث يمكن القيام بأى شىء يقوم به برنامج CGI آخر مثل تجميع بيانات نموذج وتوليد مضمون صفحة تفاعلية أو إرسال وتلقى كعك Cookies .

تمـــتاز اللغة بسهولة البرمجة ، كما يمكن ربط اللغة مع عدد من المكتبات الخارجية

لإضـــافة ممـــيزات مـــثل مكتبات التشفير والرسوم ، ويمكن العمل على معظم نظم التشغيل وبناء وتشغيل البرامج على منصات عمل ويندوز ويونكس وماكنتوش بنفس شفرة المصدر ، ولا يختلف تشغيل برنامج php على أى نظام .

الـــيوم أصبحت لغة PHP لغة برمجة ويب شعبية مع موارد أكثر وتضم مقاطع شفرة يمكن إعادة استعمالها .

- يحتوى الفصل الأول على تمهيد اللغة وقدراتها واحتياجات تركيبها .
- يتناول الفصل الثانى أساسيات ومكونات اللغة .
- يستعرض الفصل الثالث قواعد البيانات واستخدم نموذج قاعدة بيانات MySQL فيها .
- يشرح الفصل الرابع الدوال فى اللغة واستخداماتها .

تمهيد وتركيب اللغة

تشير حروف PHP إلى لغة معالج النصوص الفائقة Hypertext Preprocessor وهى لغـــة برمجة خطية عامة المقصد تمتاز بمرونتها واسعة الاستعمال ومفتوحة المصدر موجهـــة لتطوير صفحات ويب مما ييسر تضمينها فى لغة النص المتشعب HTML ، وقواعدهـــا مستعارة من لغات سي وجافا وبيرل وهى سهلة التعلم ، الهدف الأساسى لهذه اللغة هو إتاحة الفرصة لمطور ويب بمتابعة سريعة لصفحات متشعبة يتم توليدها تفاعليا ، ويمكن تحقيق أكثر من ذلك بهذه اللغة .

ظهرت لغة PHP كاختصار لكلمات صفحة البداية الشخصية Personal Home Page عام ١٩٩٤ على يد راسموس ليردورف Rasmus Lerdorf كمجموعة برامج بيرل ثم قام بتحويلها إلى أداة نصوص برمجية Script Engine مع دعم النماذج Forms لتنشأ فـــى عـــام ١٩٩٥ نسخه مفسر PHP/FI ، وفى عام ٩٧ Version 2 PHP استفاد من ٥٠٠٠٠ موقـــع دعم لهذه اللغة ، ثم تطورت وتوسعت اللغة وزادت شهرتها بإنشاء تطبـــيقات واجهة تطبيقات API لها فى الإصدار Php3 ، مما أدى إلى تطويرها بعد ذلـــك مـــرات ، ثـــم أصـــبحت تعـــرف باسم لغة معالج النص الفائق Hypertext Preprocessor .

تمـــتاز هذه اللغة بالعديد من الامتيازات منها السرعة ، والثبات ، والأمان ، وسهولة

الاستخدام .

تعنى السرعة سرعة تنفيذ البرامج ، وأيضا أنها لا تشكل حملا على الخادم ولا تبطئ عملـــه ولا تســـتنزف موارد الجهاز ، وتتأقلم مع البرامج الاخرى وتتوافق مع أنظمة التشغيل .

يفـيد ثـبات الـــنظام وعـــدم الانهيار عند استدعاء آلاف الصفحات ووجود عدد من المطوريـــن لحـــل المشاكل والعيوب ، مع نظام فعال للتعامل مع المتغيرات ، ونظام إدارة موارد خاص ، مع الأمان وحماية الموقع والتحكم فى ملف الإعدادات php.ini.

تشـــبه هذه اللغة لغة سى فى كثير من الأوامر وتركيبها ، كما تمتاز بالتوافق مع عدد مـــن واجهـــات خدم بروتوكول نقل النص المتشعب HTTP Server interfaces مثل التوافق مع خدم Apache, IIS, AOLServer, Roxen, THTTPD كما يمكن العمل مع الوحدات البرمجية CGI module ، وتعمل كواجهة عمل كثير من قواعد البيانات مثل MySQL, MS SQL, Oracle, Informix, PostgreSQL وغـيرها مـــع دعم ODBC ، ويمكـــن إنشـــاء عدد من المكتبات التى تتعامل مع برامج أخرى بسهولة منها ملفات أكروبات وفلاش وملفات Cybercash, calendars, XML, IMAP, POP وغيرها .

تحـــتوى اللغة على مترجم مكتوب بلغة سى صغير الحجم سريع العمل ، مع تطوير مفتوح ومجانية تضيف إلى المميزات .

مع التطوير تم عمل توسع جديد لحزمة SQLite وتم تحسين سرعة الوظائف الداخلية ، مـــع فحص دقيق لدعم XML ، ووجود امتداد DOM, Simple XML, XSL, PECL ، وإضافة خيارات جديدة إلى ملف التهيئة php.ini مع وظائف ودوال جديدة .

فى التطورات التالية جرى تحسين دعم تدفق البيانات Streams ، وإضافة عنوان نقل الملفـــات ftp:// ودعـــم فتح الدليل وإضافة خيارات لنقل النص المتشعب http:// ، وتعديل وتحسين الوظائف والدوال وقراءة البايتات والبحث والتعامل مع المصفوفات

والأدلة والمدى والعلامة العشرية .

هناك ثلاثة مجالات أساسية يمكن استعمال نصوص برمجيات (سكريبت) PHP فيها : برمجة ويب ناحية الخادم (المزود) Server-side scripting وهو أهم مجال وللقيام بهذا العمل تحتاج ثلاثة أشياء هى :

- مفسر PHP (CGI) أو وحدة مزود Server module .
- مزود ويب .
- مستعرض ويب لاستعراض صفحة PHP انطلاقا من المزود .

برمجة ويب من خلال سطر الأوامر Command line scripting بإمكانه إنشاء سكريبت PHP وتشغيله بدون مزود أو مستعرض ، تحتاج فقط إلى مفسر PHP لاستعماله ، هذا النوع من الاستعمال يعتبر مثاليا بالنسبة للسكريبتات التى يتم تشغيلها بشكل منتظم من خلال استعمال cron على أنظمة يونكس nix* أو Linux أو مجدول المهام فى ويندوز Windows ، وهذه السكريبتات يمكن أيضا استعمالها لإتمام مهام معالجة نص بسيط .

إنشاء تطبيقات ذات واجهات استخدام رسومية من جهة الزبون client-side GUI applications ، وربما لا تكون PHP أفضل لغة لإنشاء تطبيقات رسومية لكن لها بعض ميزات متقدمة فى تطبيقات جهة الزبون ويمكن استعمال PHP-GTK لإنشاء مثل هذه البرامج ، وتتوفر أيضا إمكانية إنشاء تطبيقات متوفرة عبر المنصات cross-platform applications. PHP-GTK ، وهى توسعة للغة .

يمكن استعمال PHP على كل أنظمة التشغيل الأكثر أهمية مثل Linux وأنظمة مستنسخة عن يونكس Unix مثل HP-UX Solaris ومثل OpenBSD وأنظمة Microsoft Windows ونظام Mac OS X وغيرهم ، وتدعم لغة PHP أغلب مزودات ويب الحالية مثل Apache , Microsoft Internet Information Server (IIS),

Personal Web Server (PWS) ، مزودات Netscape , iPlanet, Oreilly Website Pro, Caudium, Xitami, OmniHTTPd وعدة مزودات أخرى .

بالنسبة لمعظم المزودات تشتمل لغة PHP على وحدة Module وبالنسبة لتلك التى تدعم مقياس CGI فإن اللغة PHP يمكنها أن تعمل كمعالج CGI .

فى لغة PHP تتوفر حرية اختيار نظام التشغيل ومزود ويب كما يصبح بالإمكان أيضا اختيار لغة منهجية أو برمجة خليط منهما ، ومع PHP تصبح مجبرا على التعامل فقط مع مخرجات HTML .

النص البرمجى التالى يختلف عن أى سكريبت يتم إنشاؤه بلغات أخرى مثل بيرل Perl أو سى C ، فبدلا من إنشاء برنامج يتضمن كثيرا من أوامر التحكم فى مخرجات HTML تقوم بإنشاء سكريبت HTML يضم شفرة للقيام بشىء ما (إظهار نص مثلا كما فى المثال) ، ويتم تضمين شفرة PHP داخل وسوم بداية ونهاية start and end tags خاصة تسمح بالانتقال من وإلى نمط PHP .

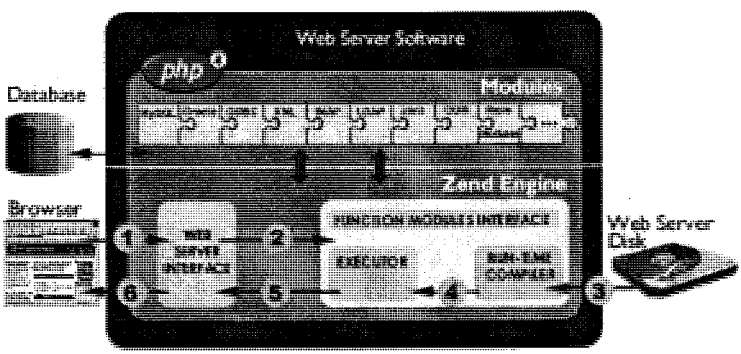
```
<html lang=ar dir=rtl>
  <head>      <title>مثال</title>    </head>
  <body>

    <?php
    echo "هذا سكريبت أهلا PHP!";
    ?>

  </body> </html>
```

تتوسع إمكانات PHP إلى الصور وملفات محمولة PDF وأفلام فلاش Flash ، وأيضا توليد أى نوع من ملفات XML بتوليد آلى لهذه الملفات ثم حفظها فى نظام الملفات file system بدلا من طباعتها .

من أقوى وأهم ميزات لغة PHP دعم قدر واسع من قواعد البيانات بإنشاء صفحة

ويب تتعامل مع قاعدة البيانات مثل :

Oracle	Ingres	Adabas D
Ovrimos	InterBase	dBase
PostgreSQL	FrontBase	Empress
Solid	mSQL	FilePro (read-only)
Sybase	MS-SQL	Hyperwave
Velocis	MySQL	IBM DB2
Unix dbm	ODBC	Informix



تـــتوفر أيضا توسعة تجويد لقاعدة بيانات DBX مما يسمح باستعمال أى قاعدة بيانات مدعومة من قبل هذه التوسعة بصورة شفافة .

كمـــا تدعم معيار الاتصال المفتوح بقواعد البيانات Open Database Connection (ODBC) بحيث يمكن الاتصال بأى قاعدة بيانات أخرى تدعم هذا المعيار العالمى .

تدعم لغة PHP أيضا التفاعل مع خدمات أخرى تتعامل ببروتوكولات مثل : LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM والكثـــير غيرها ، ويمكنها أيضا فتح وصـــلات شـــبكية أولـــية Raw Network Sockets وإحداث تعاملات باستخدام أى بروتوكول آخر .

تدعـــم PHP نسق WDDX المعقد الذى يسمح افتراضيا بتبادل البيانات بين كل لغات

برمجة ويب ، وفى الاتصال المتبادل interconnection تدعم اللغة PHP تمثيل كائنات جافا Instantiation of Java objects واستعمالها بصورة شفافة وكائنات PHP objects وتوسعة CORBA من أجل ولوج كائنات بعيدة Remote Objects .

توفر PHP ميزات جديدة مفيدة لمعالجة النصوص انطلاقا من POSIX Extended أو عبارات بيرل الاعتيادية Perl regular expressions وصولا إلى تفسير مستندات إكس ام ال parsing XML documents لتفسير وولوج مستندات XML ويمكن استعمال توسعة XSLT خاصة من أجل تحويل مستندات XML .

عند استعمال PHP فى حقل التجارة الإلكترونية تتوفر دوال Cybercash payment, CyberMUT, VeriSign Payflow Pro, CCVS مفيدة لبرامج الدفع الإلكترونى .

هناك أيضا عدة توسعات أخرى مفيدة مثل دوال البحث mnoGoSearch ودوال بوابة الدردشة IRC وعدة آليات ضغط gzip bz2 وتحويل التقويم Calendar conversion والترجمة Translation .

الصفحات المطعمة بلغة PHP يتم التعامل معها مثل صفحات HTML ويمكن تحريرها بنفس طريقة صفحات HTML الاعتيادية .

بفرض تشغيل خادم يدعم اللغة PHP وبأن كل الملفات المنتهية بامتداد PHP يتم تداولها من قبل PHP على معظم الخدم (الامتداد الافتراضى لملفات PHP) ، إذا كان الخادم يدعم اللغة PHP ، فلن يكون لزاما عليك القيام بأى شىء وإنما فقط إنشاء ملفات PHP ووضعها فى مجلد شرعى وسيقوم الخادم بتفسيرها دون حاجة لتجميع Compile أو تثبيت أدوات إضافية (معظم الاستضافة الشرعية تقدم دعم اللغة PHP) .

لحفظ سعة الموجة والتطوير تقوم بتثبيت خادم مثل Apache وتثبيت قاعدة بيانات مثل MySQL ، ويمكن تثبيتها بطريقة أحادية أو بطريقة أسهل تتمثل فى اختيار تشغيل حزمة Package جاهزة مسبقا ، ويمكن التثبيت الآلى لهذه المكونات أو تجهيز خادم

يدعم لغة PHP على أى نظام تشغيل بما فى ذلك Linux, Windows .

تحتاج للعمل فى هذه اللغة :

محرر اللغة ، ويمكن استخدام أى معالج نصوص عادى مثل المفكرة NotePad أو الدفتر أو استخدام برامج تحرير لغة php المتوفرة مجانا على صفحات الإنترنت .

بيئة تجربة البرامج داخليا على جهاز الكمبيوتر بتحويله إلى خادم ، ثم العمل على بيئة خارجية بالبحث عن مستضيف يدعم اللغة من موقع http://hosts.php.net/search.php3 ، أو حجز موقع يدعم اللغة وتحميل البرنامج إلى الخادم لتجربته .

تركيب نسخة النظام .

يمكن الحصول على ملف PHP من خلال موقع الرابط http://www.php.net/downloads.php ، وهناك أكثر من نسخه PHP يجب أن تختار نسخه تخص نظام التشغيل .

الطرق التى يمكن تنفيذ صفحة بى اتش بى على الإنترنت هى :

طلب المستخدم المباشر عبر رابط إلى صفحة فى متصفح .

طلب المستخدم صفحة تحتوى على تضمين include لصفحة .

طلب المستخدم صفحة تقرأ صفحة عبر البرمجة fread .

مصادر PHP على الإنترنت :

وثائق اللغة PHPDocumentation بالموقع http://www.php.net/docs.php

برامج البناء PHPBuilder.com بالموقع http://www.phpbuilder.com/ .

معلومات اللغة About Perl/PHP بالموقع http://perl.about.com/ .

مجموعة كتب PHP فى العنوان http://www.php.net/books.php .

تركيب اللغة واحتياجاتها

يتم تركيب اللغة (لغة بى اتش بى PHP) واحتياجاتها من :

تركيب خادم ويب (مثل الأباتشى) .

تركيب قاعدة البيانات المستخدمة (مثل MySQL) .

على نظام التشغيل المستخدم (مثل ويندوز أو يونكس) بعدة طرق ، منها طريقة تركيب كل مكون على حدة ، ومنها استخدام أدوات برمجية يمكن بها تركيب المجموعة المستخدمة مرة واحدة مثل استخدام برنامج PHPtriad أو غيره من البرامج ، وسوف نتناول الطريقتين للفائدة .

برنامج أباتشى Apache

برنامج أباتشى هو برنامج خادم ويب يخدم الوثائق والصور وأى مادة أخرى فى موقع للعالم الخارجى ، أو هو برنامج إرسال الملفات يقبل الطلبات وينجزها .

يستعمل Apache من قبل شركات الاستضافة لتقديم خدمة استضافة مواقع ويب ، وعند تطوير صفحات ويب جانب الخادم بواسطة PHP تحتاج خادم ويب على الجهاز الشخصى ليمكن تنفيذ شفرة PHP ، ويمكن كتابة تطبيقات وتنفيذها من خلال خادم مستضيف مواقع لكن ستعانى من : وقت نقل الملفات وبطء مزود خدمة الإنترنت أو بطء الخادم المستضيف إضافة إلى تكلفة الاتصال (الأفضل تجربة التطبيقات على جهاز شخصى ثم نسخ الموقع بنقل الملفات FTP إلى موقع الاستضافة بعد التأكد من عمل التطبيقات بشكل صحيح) .

يمكن تركيب خادم أباتشى Apache على أى نظام تشغيل مثل Windows أو Linux ، ويمكن تحميله من موقع : httpd.apache.org أو من العنوان : http://www.mysql.com/Downloads/ ، أو من مواقع ويب المختلفة ، ويمكن

تحميل نسخة ثنائية قابلة للتنفيذ ، أو الحصول على ملف مضغوط .

بعد الحصول على الملف يجب فك ضغطه لكن قبل هذا يفضل نقل الملف إلى المسار التالى : /usr/local ، ويتم فك ضغطه بالأمر :

```
gunzip -c apache_1.3.xx.tar.gz | tar -xvf –
```

يقوم الأمر بإنشاء دليل apache_1.3.xx به الملفات ويمكن تغيير اسمه إلى apache ، ثم ننتقل إلى هذا الدليل من خلال أمر مع نظام Linux :

```
cd apache
```

بعد الانتقال داخل الدليل قم بتنفيذ أمر إعداد apache :

```
./configure --enable-module=most --enable-shared=max
```

جـزء enable-module=most-- اختـيارى يقـوم بتركيب كل الوحدات الديناميكية ماعدا بعضها ، ويقوم الجزء enable-shared=max-- لتحميل الوحدات إلى الذاكرة فقط عندما يحتاج .

يتم تجميع apache بالأمر :

```
make
```

بعد التجميع يتم تنصيبه بالأمر :

```
make install
```

يتم تشغيل الخادم بتنفيذ أمر تشغيل Apache :

```
./bin/apachectl start
```

من داخل الدليل أو تكتب المسار بالكامل .

بعد تشغيل الخادم سوف نفتح المتصفح ونكتب العنوان التالى :

```
Localhost
```

أو العنوان :

```
127.0.0.1
```

ملـف إعـداد Apache هو ملف باسم httpd.conf يمكن الوصول إليه فى المسار : /usr/local/apache/conf .

ويستخدم محرر نصوص مثل VI فى نظام Linux .

بعد فتح هذا الملف ابحث عن اسم الخادم ServerName وقم بإزالة علامة # من أمامه ، وعند وجود خادم استضافة يمكن وضعه مثل :

ServerName www.domain.xxx

عندما لا يتوفر خادم استضافة يمكن وضع الجهاز الشخصى بالاسم الافتراضى :

ServerName localhost

قم بالبحث عن هذا جذر المستندات DocumentRoot بعد اسم الخادم ServerName مباشرة ، وهو جزء يشير إلى دليل مكان وضع ملفات النص المتشعب HTML وهو دليل سيكون الدليل الرئيسي لخادم ويب مثال :

DocumentRoot "/home/php/site"

الشرطة أمامية مثل / ، ولو وضعنا ملف html فى المسار /home/php/site باسم php01.htm يتم الوصول إليه فى المتصفح بالعنوان localhost/php01.htm .

بعد هذه التعديلات تقوم بإعادة تشغيل Apache ثانية بالأمر :

./bin/apachectl restart

مصادر Apache على الإنترنت

Apache Documentation http://httpd.apache.org/docs/
Apache Today http://apachetoday.com/
Apache Week http://www.apacheweek.com/

تثبيت اللغة فى نظام لينوكس

يجب تثبيت نسخة أباتشى Apache على جهاز الكمبيوتر لأن لغة PHP تحتاج إلى Apache لتشغيلها ، ويمكن الحصول على نسخة اللغة PHP من العنوان : http://www.php.net/downloads.php ، أو من خلال مواقع شبكة الإنترنت المختلفة .

بعد الحصول على ملف PHP يجب فك ضغطه بعد وضع الملف فى المسار /usr/local ثم كتابة أمر الفك بالصورة :

gunzip -c php-4.x.x.tar.gz | tar -xvf –

بعد فك الضغط يتم إنشاء مجلد باسم php-4.x.x يمكن تغييره إلى php .

يتم إعداد اللغة PHP من خلال الأمر :

./configure --with-mysql --with-apxs=/usr/local/apache/bin/apxs

يقوم خيار --with-mysql بتنشيط دعم قاعدة البيانات MySQL ، ويقوم خيار --with-apxs بتعريف موقع أداة امتداد apxs لتنصيب نفسه كوحدة Apache بالمسار /usr/local/apache/bin/apxs/ ، وعند تغيير مسار تركيب Apache تحتاج تغيير هذا المسار .

لتجميع اللغة PHP نكتب الأمر :

Make

ثم نكتب أمر التنصيب :

Make Install

عندما يتم التركيب بنجاح يبقى نسخ ملف php.ini-dist للمسار /usr/local/lib وتغيير اسمه إلى php.ini ليكون ملف إعداد PHP ، ويتم ذلك باستخدام الأمر التالى الذى يقوم بالنسخ :

cp php.ini-dist /usr/local/lib/php.ini

إعداد PHP

فى نظام Linux عليك إعادة تشغيل Apache كالتالى :

/usr/local/apache/bin/apachectl restart

قم بإنشاء ملف به الشفرة التالية :

```
<?php
phpinfo()
 ?>
```

قم بحفظ الملف باسم phpinfo.php فى المجلد الرئيسى للموقع ، ثم قم بتشغيله من المتصفح بكتابة العنوان كالتالى فى سطر عنوان المتصفح :

localhost/phpinfo.php

رفع أداء PHP

تحتاج تحميل برنامج Optimizer من موقع zend وبعد الحصول على الملف تقوم بفك ضغطه بالأمر التالي :

```
gunzip -c ZendOptimizer-1\[1\].0.0-PHP_4.0.4-Linux_glibc21-i386.tar.gz
-| tar -xvf
```

انتقل داخل هذا الدليل بالأمر التالى :

```
cd ZendOptimizer-1.0.0-PHP_4.0.4-Linux_glibc21-i386
```

قم بإنشاء مجلد جديد فى المسار /usr/local/lib وقم بكتابة أمر إنشاء مجلد جديد كما يلى :

```
mkdir /usr/local/lib/ZendOptimizer
```

انسخ ملف ZendOptimer.so إلى المجلد الجديد الذى قمت بإنشائه بالأمر التالى :

```
cp ZendOptimizer.so /usr/local/lib/ZendOptimizer
```

قم بإضافة السطرين التاليين فى ملف php.ini :

```
zend.optimizer.optimization_level=15
zend_extension="/usr/local/lib/ZendOptimizer/ZendOptimizer.so"
```

تركيب قاعدة البيانات MySQL

خادم قواعد بيانات MySQL يمكن الحصول عليه من موقع : http://www.mysql.com/Downloads/ ، ويمكن تحميل نسخة ثنائية binaries قابله للتجميع والتنفيذ والتركيب السريع ، ويمكن تحميل نسخة مضغوطة ثم فك الضغط بالأمر :

```
Gunzip -c mysql-3.23.xx.tar.gz | tar -xvf -
```

سيتم إنشاء مجلد باسم mysql-3.23.xx نغير اسمه إلى mysql .

نتحول إلى المجلد بالأمر :

```
cd mysql
```

إنشاء مجموعة جديدة باسم mysql عن طريق الأمر :

groupadd mysql

إنشاء اسم مستخدم mysql للمجموعة mysql بالأمر :

useradd -g mysql mysql

تنفيذ أمر :

./configure --prefix=/usr/local/mysql –with-low-memory

الجزء --prefix=/usr/local/mysql يحدد منطقة التركيب .

الجـــزء --with-low-memory يحـــافظ على عدم فقدان البيانات عند عدم استخدامها مهما كانت الذاكر المتوقع استخدامها .

نقوم بالتجميع باستخدام أمر :

Make

التركيب بأمر :

make install

استخدام أمر تركيب الجداول :

scripts/mysql_install_db

تغيير ملكية التركيب إلى المستعمل mysql السابق الإنشاء بأمر :

chown -R mysql /usr/local/mysql
chgrp -R mysql /usr/local/mysql

يمكن تشغيل قاعدة البيانات بتنفيذ الأمر :

/usr/local/mysql/bin/safe_mysqld --user=mysql &

مصادر MySQL على شبكة الإنترنت كثيرة منها :

http://www.mysql.com/doc/T/u/Tutorial.html
http://www.mysql.com/articles/ddws/
http://www.mysql.com/doc/
http://www.sqlcourse.com/

تركيب لغة البرمجة PHP بأدوات برنامج

يمكـــن كتابة برامج لغة PHP باستخدام محرر نصوص عادى مع حفظ الملف بامتداد

php ، ويمكن تشغيل برنامج مكتوب باللغة عن طريق :

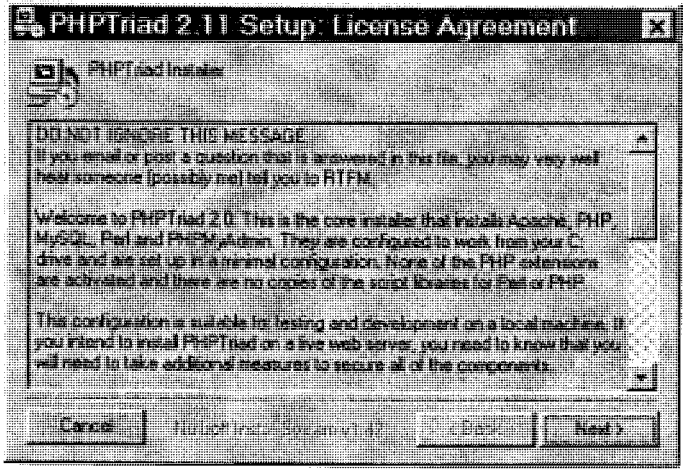
بناء خادم Server وتجربة البرامج على الكمبيوتر الشخصى .

أو إرسال الملفات إلى موقع الإنترنت وتجربتها فى الموقع .

يجـــب تنفيذ وتجربة برامج لغة PHP على الجهاز الشخصى لمراجعتها وذلك بتحويل الكمبيوتر الشخصى إلى خادم استضافة محلى ، وهى عملية سهلة التنفيذ والإعداد تتم بالخطوات التالية :

تحميل برنامج خادم مثل برنامج PHP Triad المجانى أو غيره ووضع ملف البرنامج فـــى القرص الصلب (تقسيم D مثلا) فى دليل فرعى تحت اسم الخادم Server أو أى دلـــيل تخـــتاره بعـــد نقله من شبكة الإنترنت ، وبالنقر المزدوج على رمز (أيقونة) البرنامج .

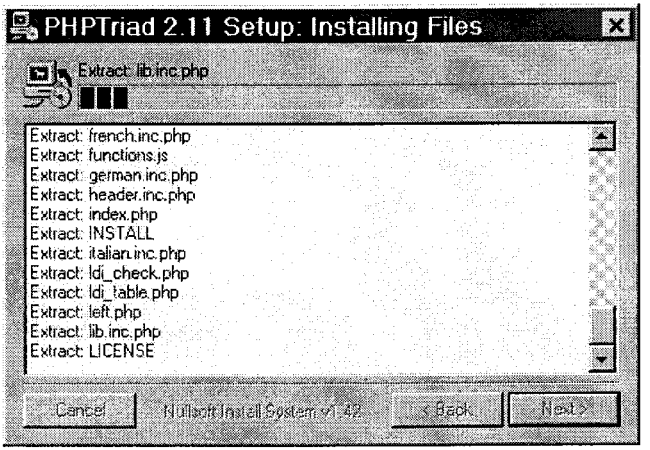
يظهر الشكل التالى الذى يعرض شروط اتفاقية استخدام البرنامج :



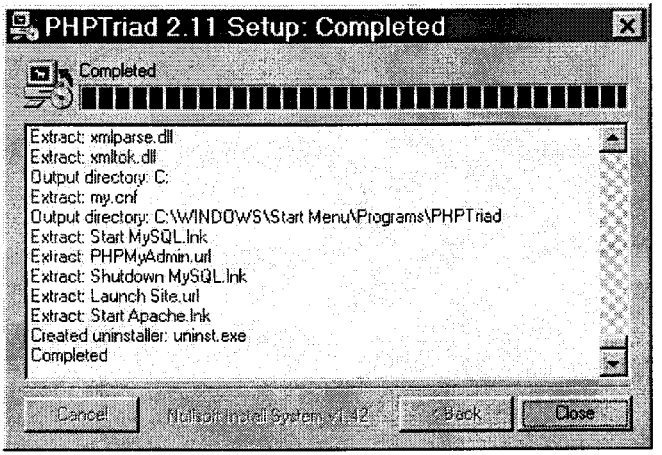
بـــنقر زر الـــتالى تنتقل للخطوة التالية التى تبين تركيب وتثبيت البرنامج على جهاز الكمبيوتر .

يمكـــن معـــرفة تفاصيل عملية التثبيت التى تتم عن طريق نقر زر إظهار التفاصيل

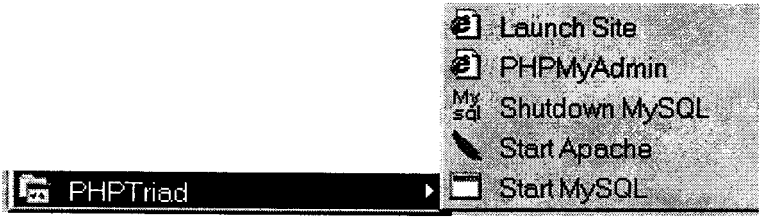
Show details لمعرفة مكان وملفات تثبيت البرنامج .



بـنهاية عملـية التثبيـت يظهـر مربع حوار يبين مكان وضع البرنامج ضمن قائمة الـبرامج Programs الموجـودة فى قائمة ابدأ Start فى ويندوز ، كما يبين أن دليل الخرج سيكون هو القرص الصلب C: .

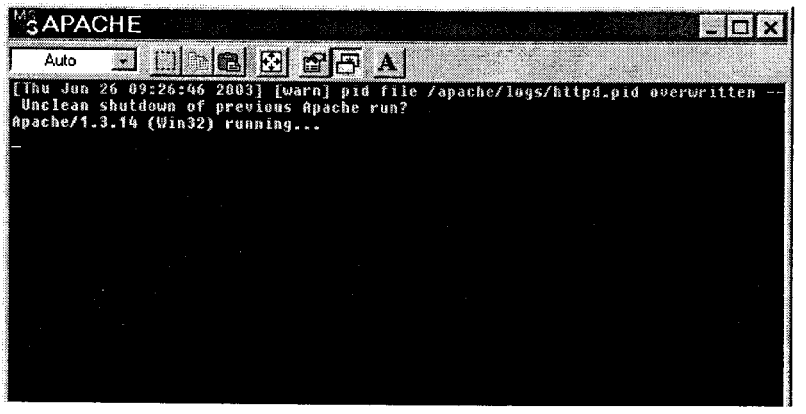


وبنقر زر إغلاق Close يكتمل تثبيت البرنامج ، ويصبح جهاز الكمبيوتر خادم تشغيل برمجيات اللغة بعد تشغيل هذا البرنامج من قائمة البرامج :



لتحويل الكمبيوتر الشخصى إلى خادم ويب تقوم بالوصول إلى برنامج PHPTriad من قائمة البرامج Pograms المتفرعة من قائمة ابدأ Start فى ويندوز : ومن محتويات البرنامج المعروضة على الشاشة تكون هناك حاجة لتشغيل خدمتين ليكون الجهاز خادم ويب .

الخدمة الأولى هى تشغيل محاكى الخادم الشخصى الذى يحول الكمبيوتر إلى خادم شخصى تستطيع من خلاله تشغيل صفحات ويب وبرامج لغة PHP ، ويتم ذلك عن طريق تشغيل خدمة أباتشى بنقر أيقونة بداية أباتشى Start Apache لتظهر شاشة دوس تبين عمل خدمة أباتشى :

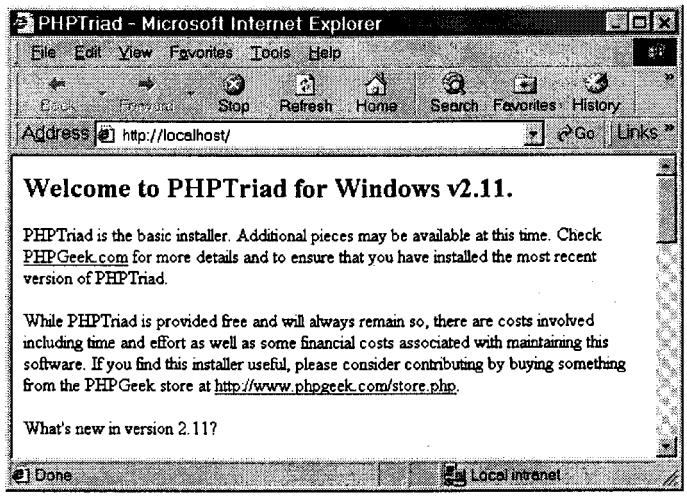


مما يعنى أن جهاز الخادم يعمل بصورة طبيعية .

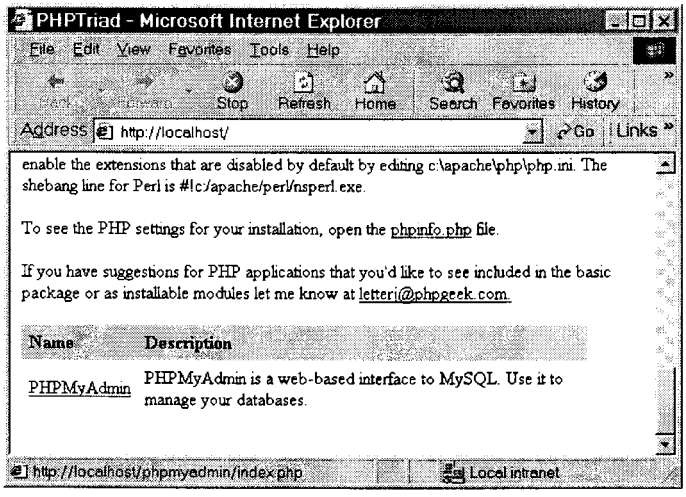
الخدمة الثانية المطلوب تشغيلها هى محرك قاعدة البيانات MySQL ليمكن إنشاء قواعد البيانات واستخدام الجداول وإدخال واستخراج البيانات ، ولتشغيل قاعدة

البـــيانات انقر أيقونة البداية Start MySQL ، لتظهر شاشة موجه أوامر دوس بسرعة وتختفى ، لتبين عمل البرنامج .

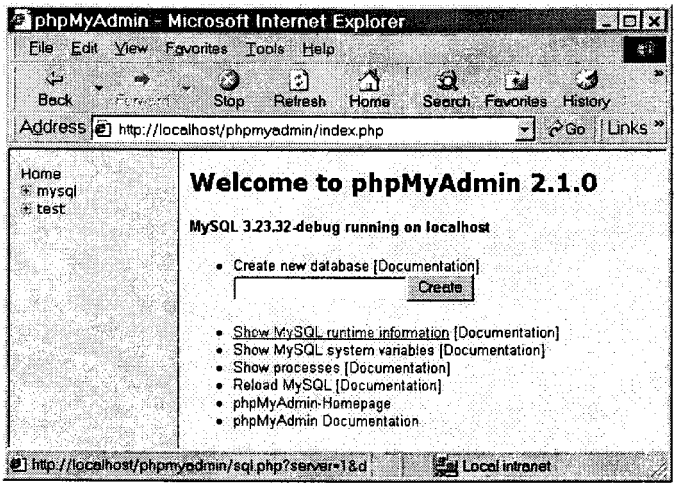
بعد انتهاء تشغيل الخدمتين يصبح جهاز الكمبيوتر الشخصى جاهزا للعمل كمستضيف صـــفحات ويـــب ، وعـــند تشـــغيل بـــرنامج الاســـتعراض وكتابة العنوان المحلى http://localhost فى مربع العنوان سوف تصل إلى الصفحة الرئيسية لهذا البرنامج فى الخادم المحلى :



هذا يعنى أن خطوات التنفيذ سليمة ويعمل الكمبيوتر كخادم بطريقة طبيعية ، لمعرفة أن محرك قاعدة البيانات يقوم بعمله على النحو الطبيعى تجد فى أسفل الصفحة رابطة باسم PHPMyAdmin .



بنقرها تظهر إدارة قواعد البيانات المتوفرة في الجهاز :



عن طريق هذا البرنامج تستطيع تصفح وإنشاء وحذف قواعد البيانات والتعامل معها.

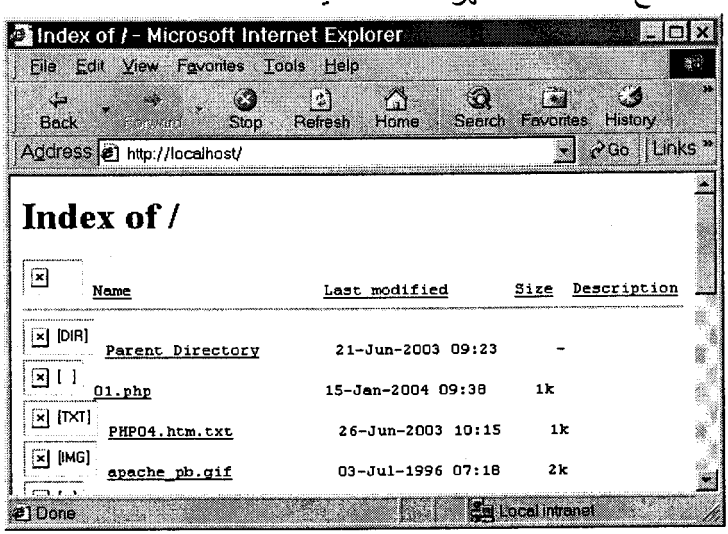
لا يخدم الخادم الشخصي الجهاز بصورة كاملة فالصفحات والبرامج الموجودة خارج هذا الخادم لا يمكن التعامل معها كصفحات نشطة أو متفاعلة .

لكـــى يمكـــن وضـــع الصفحات والنصوص البرمجية بحيث يمكن تصفحها وتنفيذها كصفحات تفاعلية توضع فى المجلد C:\apache\htdocs أو فى أى مجلد آخر أو فى مجلد متفرع منه ، ويعتبر البرنامج هذا المجلد الذى يقوم بإنشائه هو العنوان الرئيسى للموقـــع على الكمبيوتر الشخصى ، فعند كتابة localhost فى شريط عنوان المتصفح يـــنقلك مباشـــرة إلـــى هـــذا المجلـــد فمثلا الملف one.htm الذى يوضع فى المجلد C:\apache\htdocs يمكن تصفحه بكتابة العنوان http://localhost/one.htm .

الصـــفحة الرئيســـية للمجلد هى ملف باسم Index بامتداد HTM أو امتداد HTML أو امـــتداد PHP ، وعند كتابة عنوان المجلد فقط يتم تحميل هذا الملف مباشرة فمثلا عند إدخال عنوان فى شريط عنوان المتصفح مثل www.mysite.com فسوف يتم الدخول مباشرة إلى الصفحة الرئيسية التى ستكون هى www.mysite.com/index .

وإذا لم يكن هناك ملف باسم Index فسوف يحدث التالى :

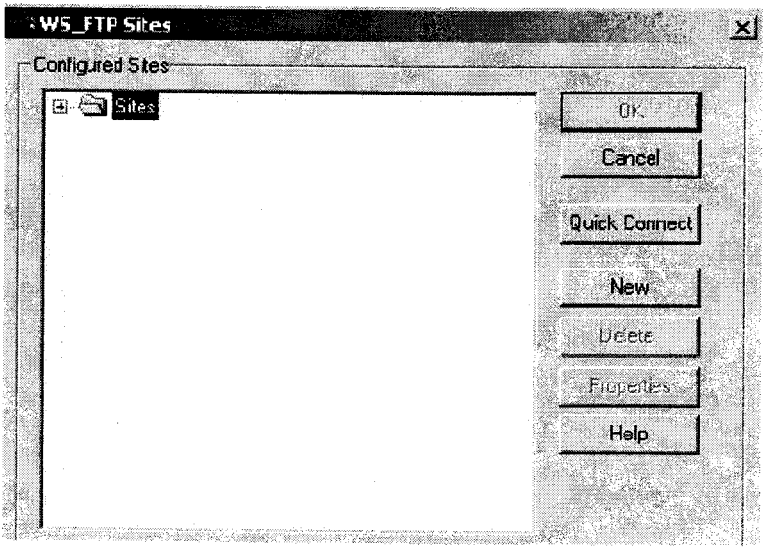
قـــم بتغيير الملف index.php إلى اسم آخر مثل index12.php ، واكتب فى شريط عنوان المتصفح Localhost تظهر الشاشة التالية :



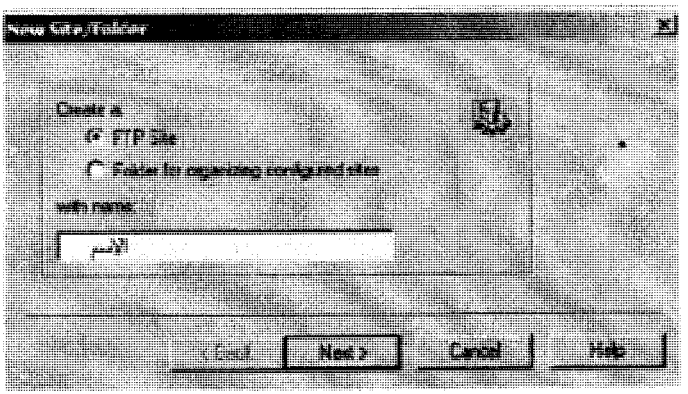
أى أنـــه فـــى حالة عدم وجود الملف فسوف يتم عرض محتويات المجلد لهذا يجب وضـــع هذا الملف فى جميع المجلدات حتى لا يتم عرض محتويات المجلد ، وبالتالى تفتقد سرية المحتويات .

إرسال البرامج إلى شبكة الإنترنت

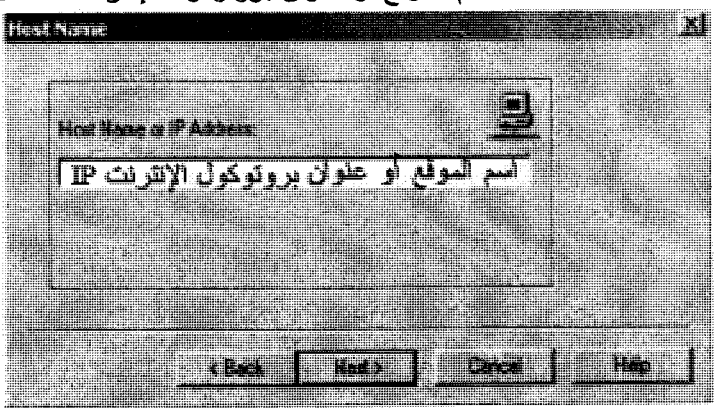
لـــتجربة البرنامج بعد الانتهاء منه فى موقع على شبكة الإنترنت سوف تقوم بتركيب بـــرنامج نقل ملفات FTP مثل البرنامج WS_FTP ، وعند تشغيل هذا البرنامج تظهر الشاشة :



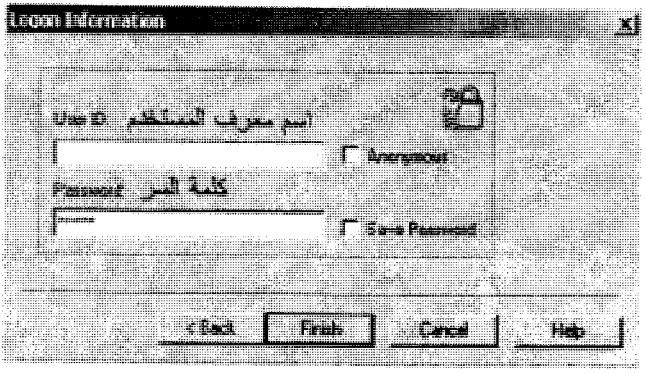
ولمـــا قد تكون هذه هى المرة الأولى لتشغيل هذا البرنامج فسوف تحتاج إلى نقر زر الجديد New فتظهر الشاشة التالية :



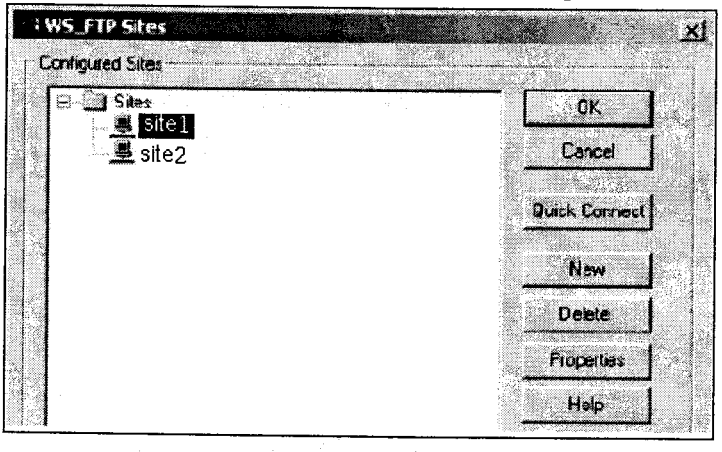
فى الشاشة تكتب اسم الاتصال وهو أى اسم ترغب فيه ثم تنقر على زر التالى Next لتظهر شاشة تالية تكتب فيها اسم الموقع أو عنوان بروتوكول الإنترنت IP مثل :



بعد ذلك تنقر زر التالى Next لتظهر الشاشة التالية :



فـــى هـــذه الشاشة تكتب اسم معرف المستخدم User ID وكلمة السر Password فى مربعات شاشة البرنامج ثم تنقر زر الانتهاء Finish فتظهر الشاشة التالية :



فـــى الشاشـــة السابقة تختار الموقع الذى تريد الانتقال إليه والاتصال به ثم تنقر زر الموافقـــة OK فـــتظهر شاشة تبين نجاح الاتصال ، وتجد فى اليمين محتويات موقع الاستضـــافة علـــى شبكة الإنترنت ، وتجد فى اليسار الملفات الموجودة على الجهاز المحلى .

الصيغة الأساسية للغة PHP

سوف تقوم بإنشاء ملف مثل hello.php ووضعه فى مجلد الجذر للخادم DOCUMENT_ROOT (أو فى مجلد htdocs فى الخادم المحلى) ويتضمن التالى :

```
<html lang=ar dir=rtl>
<head>
<title> اختبار PHP</title>
</head>
<body>
<?php echo "<p>يا عالم أهلا</p>"; ?>
</body> </html>
```

استخدم المستعرض لعرض الملف بمحدد المورد الشامل الخاص الذى ينتهى بالإشارة المرجعية للملف "/hello.php" ، وإذا كنت بصدد التطوير محليا فإن محدد المورد يظهر مثل http://localhost/hello.php أو http://127.0.0.1/hello.php بناء على طريقة تهيئة الخادم ، وإذا تم تثبيت كل شيء بشكل صحيح فإن الملف سيتم تفسيره من قبل PHP وتظهر النتيجة فى المستعرض كما لو كنت قد كتبت النص التالى :

```
<html lang=ar dir=rtl>
<head>
<title>اختبار PHP</title>
</head>
<body>
<p>يا عالم أهلا</p>
</body>
</html>
```

الملف لا يحتاج أن يكون قابلا للتنفيذ أو خاصا فهو ملف HTML عادى يتضمن

مجموعة وسوم خاصة تسمح بالقيام بالكثير من المهام .

البرنامج بسيط بحاجة إلى استعمال PHP لإنشاء صفحة مثل هذه فكل ما تقوم به هو عرض : أهلا يا عالم باستعمال بلاغ **()echo** فى لغة PHP .

إذا جربت المثال ولم يصدر عنه شيء أو لم يعرض كاملا فقد تكون هناك مشكلة على الخادم تحتاج التأكد ومراجعة التثبيت .

فى المثال إظهار تنسيق الخاص بوسم PHP فى هذا المثال ، تستعمل php?> للإشارة إلى بداية وسم PHP ، بعدها يدرج بلاغ PHP ثم خروج من نمط PHP من خلال إضافة وسم الإغلاق <? ، ويمكن الانتقال من وإلى نمط PHP فى ملف HTML كما تشاء .

يمكن استعمال محرر نص أو بيئات تطوير متكاملة Integrated Development Environments (IDE) التى تستعمل لإنشاء وتحرير ملفات PHP ، وهناك الكثير من قائمة محررات PHP ، ويوفر المحرر إبراز الصيغ syntax highlighting وأحيانا ميزة التصحيح Debug .

تستخدم أيضا معالجات الكلمات لكن يجب حفظ الملف كنص خام PLAIN TEXT بامتداد php .

يجب التأكد نوع المستعرض الذى يستعمله الشخص للاطلاع على الصفحة ، من أجل القيام بهذا تقوم بمراقبة السلسلة النصية لعميل المستخدم user agent string التى يرسلها المستعرض كجزء من استعلام HTTP ، هذه المعلومة يتم حفظها فى متغير ، المتغيرات تبدأ دائما برمز الدولار ، فى لغة PHP المتغير الذى يهم فى هذا هو المتغير $_SERVER["HTTP_USER_AGENT"] .

المتغير $_SERVER هو أحد متغيرات PHP الخاصة ويضم كل المعلومات المتعلقة بالخادم وهو معروف كشامل آلى Auto global أو فائق الشمول Super global قبل

هـذا الوقـت تعمـل بـدلا مـنه مصـفوفة $HTTP_*_VARS الأقـدم مـثل $HTTP_SERVER_VARS ، وبالرغم من أنها غير ذات أهمية إلا أن هذه المتغيرات القديمة لا زالت موجودة .

يمكن إظهار هذا المتغير فقط من خلال :

```
<?php echo $_SERVER["HTTP_USER_AGENT"]; ?>
```

نموذج عن نتيجة هذا يمكن أو يكون كالتالى :

```
Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 5.0)
```

توفـر PHP العديـد مـن أنواع المتغيرات ، فى المثال السابق قمنا بطباعة عنصر مصفوفة .يمكن للمصفوفات أن تكون مفيدة .

$_SERVER هو فقط أحد متغيرات لغة PHP ، ويمكن الحصول على قائمة كاملة لها من خلال إنشاء ملف يشبه ما يلى :

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

إذا قمـت بتحميل الملف فى مستعرض تجد صفحة مليئة بالمعلومات بخصوص PHP وبها أيضا لائحة بكل المتغيرات المتوفرة .

يمكـن وضع عدة بلاغات PHP ضمن وسم PHP وإنشاء مقاطع صغيرة من الشفرة تقـوم بأكثر من مجود عرض echo مثلا إذا كنت تريد التأكد من برنامج Internet Explorer يمكن القيام بشيء مثل :

```
<?php
if (strstr($_SERVER["HTTP_USER_AGENT"], "MSIE")) {
        echo "أنت بصدد استعمال Internet Explorer<br />";
}
?>
```

نموذج لنتيجة هذا يمكن أن تكون :

```
أنت بصدد استعمال Internet Explorer<br />
```

مفهـــوم مناداة دالة **strstr()** أن الدالة **strstr()** دالة ضمن PHP تقوم ببحث سلسلة نصـــية من أجل سلسلة نصية أخرى ، فى هذه الحالة فإن البحث عن "MSIE" داخل $_SERVER["HTTP_USER_AGENT"] ، إذا تـــم إيجـــاد سلســـلة نصية تقوم الدالة بإرجـــاع **TRUE** وغير ذلك تقوم بإرجاع **FALSE** ، إذا أرجعت **TRUE** فإن بلاغ if يقـــوم بالتقيـــيم **TRUE** مـــع الشفرة المتضمنة داخل علامتى الحصر {braces} يتم تنفـــيذها ، فـــى غـــير ذلك لن يتم الأمر ، ويمكن إنشاء أمثلة مماثلة بواسطة if else ودوال أخرى مثل **strtoupper()** أو **strlen()** .

لإظهار الانتقال من وإلى نمط PHP حتى فى وسط مقطع PHP :

```
<?php
if (strstr($_SERVER["HTTP_USER_AGENT"], "MSIE")) {
?>
<h3>strstr أرجع لا بد أنه true</h3>
<center><b>أنـــــــت بصـــــــدد اســـــــتعمال Internet
Explorer</b></center>
<?php
} else {
?>
<h3>strstr أرجع لا بد أنه false</h3>
<center><b>لســـــــت بصـــــــدد اســـــــتعمال Internet
Explorer</b></center>
<?php
}
?>
```

ونموذج نتيجة هذا يمكن أن يكون كالتالى :

```
<h3>strstr أرجع لا بد أنه true</h3>
<center><b>أنت بصدد استعمال Internet Explorer</b></center>
```

أقوى ميزات PHP هى طريقة التعامل مع نماذج HTML ، المفهوم الرئيسى هو أن أى عنصـــر فى النموذج يكون متوفرا لنصوص PHP ، وهذه أمثلة بخصوص استعمال النماذج مع PHP :

```
<form action="action.php" method="POST">
اسم: <input type="text" name="name" />
عام : <input type="text" name="age" />
<input type="submit">
</form>
```

هذا عبارة عن نموذج HTML خام بدون وسوم خاصة ، عندما يملأ المستخدم النموذج ويضغط زر الإرسال سيتم مناداة صفحة action.php فى هذا الملف مثل :

```
<?php echo $_POST["name"]; ?> يا أهلا
<?php echo $_POST["age"]; ?> سنة . عام
```

نموذج نتيجة هذا يمكن أن يكون كالتالى :

```
أهلا يا محمد
عام ٢٢ سنة.
```

مـــا يحققـــه هذا واضح ، ليس هناك أكثر من متغيرات $_POST["name"] ومتغير $_POST["age"] تم تحديدها آليا من قبل PHP ، وسيعمل الشامل الآلى $_SERVER ، وقـــد مهـــدوا للشامل الآلى $_POST الذى يضم كل البيانات من نوع POST انتبه كـــيفـــطـــريقة *method* نموذج هى POST إذا استعملت طريقة *GET* فإن معلومات

نمـــوذج ســـتطل بدلا من ذلك فى الشامل الآلى $_GET ، ويمكن أيضا الشامل الآلى $_REQUEST .

الدوال فى لغة PHP

بخلـــف الدوال المعرفة تحتوى اللغة على مجموعة كبيرة من الدوال منها مجموعات الدوال التالية ، وعندما تعرف أن كل مجموعة من هذه الدوال تحتوى على عدد كبير من الدوال تعرف حجم واتساع اللغة :

Apache-specific Functions
Array Functions
Aspell functions
BCMath Arbitrary Precision Mathematics Functions
Calendar functions
CCVS API Functions
COM support functions for Windows
Class/Object Functions
ClibPDF functions
CURL, Client URL Library Functions
Cybercash payment functions
Database (dbm-style) abstraction layer functions
Date and Time functions
dBase functions
DBM Functions
Directory functions
DOM XML functions
Error Handling and Logging Functions
filePro functions
Filesystem functions
Forms Data Format functions
FTP functions
Function Handling functions
GNU Gettext
GMP functions
HTTP functions

Hyperwave functions
ICAP Functions
Image functions
IMAP, POP3 and NNTP functions
Informix functions
InterBase functions
Ingres II functions
LDAP functions
Mail functions
Mathematical Functions
MCAL functions
Mcrypt Encryption Functions
Mhash Functions
Microsoft SQL Server functions
Miscellaneous functions
mSQL functions
MySQL functions
Network Functions
Unified ODBC functions
Oracle 8 functions
Oracle functions
Ovrimos SQL functions
Output Control Functions
PDF functions
Verisign Payflow Pro functions
PHP options & information
POSIX functions
PostgreSQL functions
Program Execution functions
Pspell Functions
GNU Readline
GNU Recode functions
Regular Expression Functions (Perl-Compatible)
Regular Expression Functions (POSIX Extended)
Satellite CORBA client extension
Semaphore and Shared Memory Functions
SESAM database functions

Session handling functions
Shared Memory Functions
Shockwave Flash functions
SNMP functions
Socket functions
String functions
Sybase functions
URL Functions
Variable Functions
WDDX functions
XML parser functions
XSLT functions
YAZ functions
YP/NIS Functions
Zlib Compression Functions

الفصل الثاني

أساسيات اللغة

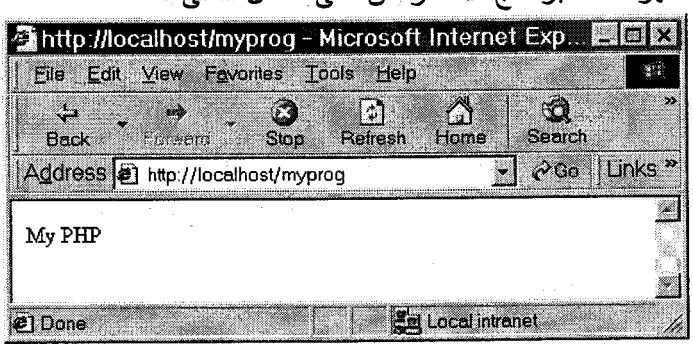
بعد تشغيل الخادم ومحرك قواعد البيانات قم بالانتقال إلى المجلد C:\apache\htdocs (يمكن عمل مجلد جديد واستخدامه) .

قم بتشغيل برنامج المفكرة Notepad واكتب السطور التالية :

```
<?php
Echo "My PHP" ;
?>
```

واحفظ الملف باسم myprog.php فى مجلد C:\apache\htdocs\ .

قم بتشغيل برنامج المتصفح واكتب فى شريط العنوان C:\apache\htdocs\myprog.php ثم اضغط مفتاح الإدخال ، وإذا كانت الخطوات سليمة ستظهر شاشة برنامج الاستعراض على الشكل التالى :



فى هذا المثال البسيط ثم إبلاغ المستعرض أن الجملة التالية للرموز فى السطر الأول جملة لغة PHP لإرسالها إلى مترجم اللغة ليحولها بالتفسير الصحيح إلى جملة مفهومة للمتصفح ، إذا لم يتم إبلاغ المستعرض بأن الجمل التالية سوف تكون جمل لغة PHP فسوف يعتبرها جمل لغة النص المتشعب HTML ، ويتم إهمال الجمل ذات التركيب الخطأ فى لغة النص المتشعب من قبل المستعرض .

فى السطر الثانى إبلاغ المستعرض أن يقوم بكتابة كلمات موجودة بين علامتى تنصيص ، وانتهى السطر بالفاصلة المنقوطة لتخبر المستعرض بانتهاء سطر الأوامر ، فى السطر الثالث الأخير تم إبلاغ الخادم بنهاية محتويات أوامر لغة PHP وانتها جمل سطور الأوامر .

لغة النص المتشعب HTML

ولغة الصفحات الشخصية PHP

يمكن كتابة شفرات HTML مع لغة PHP بطريقتين :

الطريقة الأولى تستعمل عند وجود جملة قصيرة من PHP فى صفحة HTML .

الطريقة الثانية فى مواقع كبيرة الحجم أو تستعمل قواعد البيانات تستخدم علامات PHP فى بداية النص مع وضع علامات HTML فيه .

عند حفظ النص بامتداد لغة PHP مثل php أو php3 يجب وضع علامة الفتح ?> فى أول سطر بالنص بدون مسافة ، وتوضع علامة الإغلاق <? فى نهايته ، ويمكن أن يحتوى النص على وسوم (علامات) لغة HTML ، وقد تتكرر علامات الفتح والإغلاق أكثر من مرة فى نفس النص ، مثال :

```
<?
echo "Hello or Welcome " ;
echo "This is PHP output ";
```

```
?>

<title> PHP Language </title>

<?
echo " PHP again END" ;
?>
```

تلاحظ كتابة أمر الخرج فى لغة PHP وهو أمر echo ، ثم إغلاق الكتابة ، وكتبنا علامات خاصة بلغة النص المتشعب HTML ، ثم قمنا بفتح الكتابة مرة أخرى ، ثم تابعنا كتابة النص .

يجب أن نكتب فى نهاية كل سطر رمز الفاصلة المنقوطة ، وهناك استثناء فالمعاملات الشرطية مثل if ، وحلقات التكرار مثل while لا نضع لها فاصلة منقوطة فى نهاية السطر .

إذا كان سطر النص طويلا يمكن تفرقة السطر بدون وجود دالة فى بداية السطر الثانى ويمكن أن ينتهى السطر بدون فاصلة منقوطة ، ويبدأ السطر الثانى كتكملة للسطر السابق ، وتأكد من الفاصلة المنقوطة فى نهاية السطر .

Basic syntax - HTML-embedded scripting language.

```
<html>
  <head>
    <title>Example</title>
  </head>
  <body>

    <?php
    echo "This is PHP script!";
    ?>

  </body>
</html>
```

فى المثال التالى مزج بين اللغتين كما يلى :

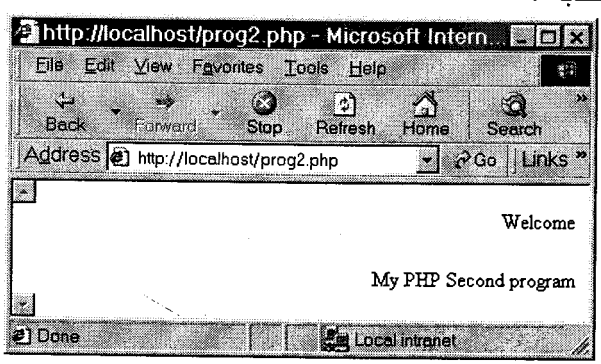
قم بتشغيل برنامج المفكرة Notepad واكتب السطور التالية :

```
<html dir=rtl>
<p> Welcome </p>

<?php
Echo "My PHP Second program" ;
?>
```

واحفظ الملف باسم prog2.php فى مجلد C:\apache\htdocs\ .

قم بتشغيل برنامج المتصفح واكتب فى شريط العنوان C:\apache\htdocs\prog2.php ثم اضغط مفتاح الإدخال ، وإذا كانت الخطوات سليمة تظهر شاشة برنامج الاستعراض على الشكل التالى :

لاحظ اتجاه الكتابة .



من الطرق المستخدمة لتعريف الخادم باستخدام لغة PHP الآتى :

```
<?php
Echo "My PHP" ;
?>
```

أو كتابة نفس الجمل على الصورة :

```
<?
Echo "My PHP" ;
?>
```

أو كتابة نفس الجمل على الصورة :

```
< script language="php"
Echo "My PHP" ;
</script>
```

أو كتابة نفس الجمل على الصورة :

```
<%
Echo "My PHP" ;
%>
```

ملف بى اتش بى ملف نص برمجى Script به شفرة لغة بى اتش بى مضمنة فى شفرات لغة النص المتشعب Html بامتداد ملف php أو php3 أو phtml أو أى امتداد يدعمه الخادم ، فى هذا الملف يمكن تضمين شفرات لغة النص المتشعب html وشفرة بى اتش بى داخلها مثل الشفرة التالية :

```
<?
echo "MY PHP ";
?>
```

عند وضع الشفرة السابقة فى ملف نص وحفظه بامتداد php على الخادم (الكمبيوتر الشخصى الذى يعمل كخادم أو على خادم استضافة) وطلب تنفيذه سيعرض رسالة MY PHP .

Ways of escaping from HTML

```
<? echo ("this is the simplest, an SGML processing instruction\n"); ?>

<?php echo("if you want to serve XHTML or XML documents, do like
this\n"); ?>

<script language="php">
    echo ("some editors (like FrontPage) don't
        like processing instructions");
</script>

<% echo ("You may optionally use ASP-style tags"); %>
<%= $variable; # This is a shortcut for "<%echo .." %>
```

هناك عدة طرق ليعرف برنامج الاستعراض أن شفرة ما هى شفرة لغة php وليست

شفرة لغة النص المتشعب html منها الرموز ?> ، وهى الطريقة الأكثر شيوعا مثال :

```
<?
Code ... ;
...
?>
```

هــذا النص يبين كيفية كتابة النصوص البرمجية فى اللغة ، ففى بداية كل نص نكتب الــرموز ?> ، وفى نهاية النص نكتب الرموز <? ، ولهذه العلامات أكثر من صورة منها :

```
<?
Code ... ;
...
?>
```

وهو الشكل الشائع الاستخدام . أو نكتب :

```
<? Php
Code ... ;
...
?>
```

أو :

```
<%
Code ... ;
...
%>
```

المستخدم مع لغة ASP . أو الشكل :

```
<script language ="php">
..
  ..
</script>
```

كما يمكن كتابة النص على الصورة :

```
<? Php echo "MY Php"; ?>
```

كل جملة برمجية فى بى اتش بى تنتهى بالفاصلة المنقوطة مثال :

```
<? Php
echo "MY Php" ;
?>
```

التعليقات Comments

التعلـــيقات أو الملاحظـــات فى لغات البرمجة عموما عبارة عن نص توضيحى يتم تجاهله عند تنفيذ البرنامج ، وتكتب الملاحظات بعدة طرق فى هذه اللغة منها :

// تعليق فى سطر واحد ..

*/تعليق يمكن كتابته فى عدة سطور ..

منها هذه السطور التى تنتهى عند وجود العلامة

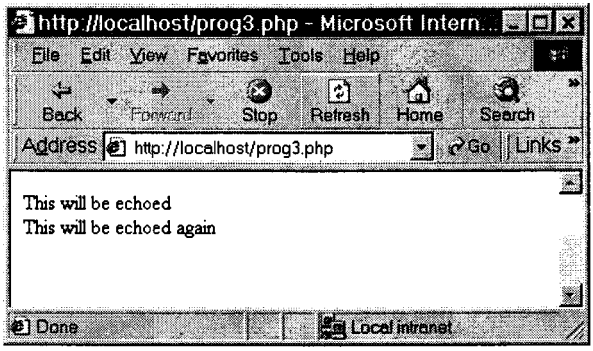
/*بهذا تنتهى سطور التعليقات المحاطة بين العلامتين

```
<?php
   echo "This is a test"; // This is a one-line c++ style comment
   /* This is a multi line comment
      yet another line of comment */
   echo "This is yet another test";
   echo "One Final Test"; # This is shell-style style comment
?>
```

لكتابة تعليق من سطر واحد نستخدم العلامتين // ثم نكتب التعليق مثل :

```
<?php
// notes about
Echo "This will be echoed <br>" ;
// Another note
Echo "This will be echoed again" ; // Another note
?>
```

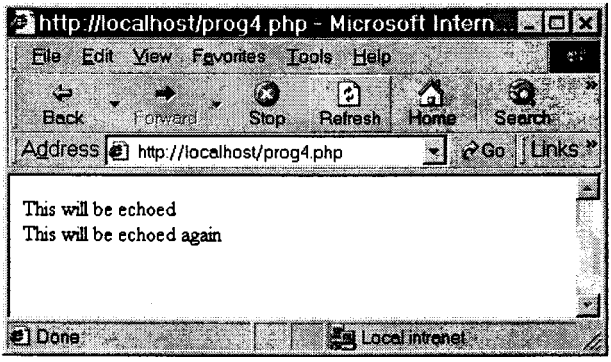
النتيجة :



لكـــتابة تعلـــيق من عدة سطور نستخدم العلامتين */ ثم نكتب التعليق الذى يمتد على عدة سطور ثم ينتهى بالعلامتين /* عند نهاية التعليق مثال :

```
<?php
Echo "This will be echoed <BR>" ;
/*
Line 1 of notes
Line 2 of notes
Line 3 of Comments
*/
Echo "This will be echoed again " ;
?>
```

النتيجة :



ويمكن كتابة التعليقات فى أى جزء من البرنامج بشرط إبلاغ البرنامج بوجود تعليق .

المتغيرات Variables

تتعرف لغة PHP على نوع المتغير تلقائيا ، كما تقوم بالتحويل بين المتغيرات بسهولة فيمكـــن استخدام متغير نص وتحويله إلى متغير رقمى بمجرد كتابة قيمة رقمية فيه ، وتدعم اللغة أنواع المتغيرات التالية :

الأرقام الصحيحة Integers .

أعداد الفاصلة العائمة Floating Point Numbers .

الأرقام العشرية .

سلاسل النصوص Strings .

المصفوفات Arrays .

يـــتم التعبير عن المتغير باسم مسبوق بعلامة الدولار $ فمثلا يعرف المتغير a الذى يحتوى على قيمة قدرها 58 على شكل :

```
$a =58 ;
```

فـــى الســـطر تخصـــيص المتغير a بقيمة ٥٨ ، وينتهى سطر التخصيص بالفاصلة المنقوطة .

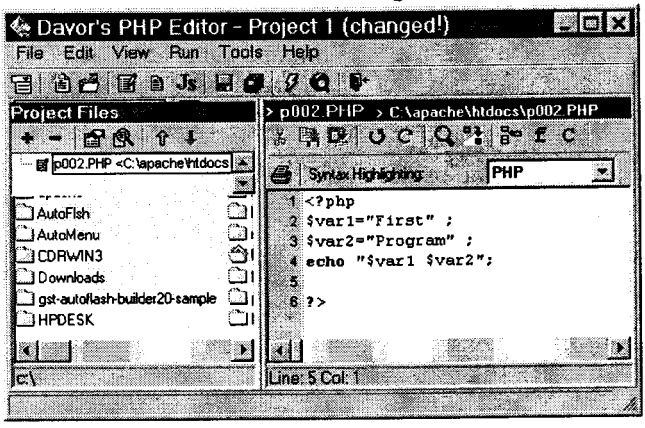
تعتبر لغة php أى كلمة مسبوقة بعلامة الدولار $ على أنها متغير ، مثال :

```
<? Php
$var1="First" ;
$var2="Program" ;
$var3= 12.786 ;
$var4= "My phone number : 111111:" ;
$var5= 22;
echo "$var1 $var2";
?>
```

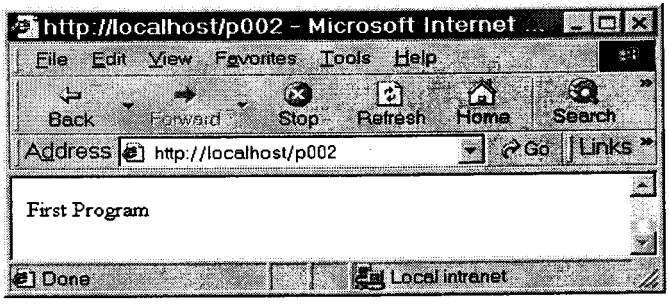
فـــى الســـطور الأول والـــثانى تعريف متغير نص ، وفى الثالث تعريف متغير رقم عشرى ، فى الرابع تعريف نص يحتوى على رقم هاتف ، وفى الخامس رقم صحيح، وفـــى السطر التالى تجميع كتابة المتغيرين الأول والثانى وإخراج الناتج على الشاشة

لتجميع نص .

فى الشكل التالى استخدام أحد برامج تحرير لغة PHP لكتابة مثال وحفظه :



فى الشكل التالى ناتج تنفيذ المثال :



تميز المتغيرات بين حالة الأحرف (كبيرة أو صغيرة) ، ولا تبدأ المتغيرات برقم .

الأعداد الصحيحة Integers :

الأعـداد الصـحيحة نوع من المتغيرات يأخذ قيمة رقمية صحيحة موجبة أو سالبة ، وتعتبر اللغة أى رقم بهذه الخصائص وغير محاط بعلامتى تنصيص هو عدد صحيح .

تدعـم لغـة php الأعـداد الصحيحة بأنواعها الحقيقية والثنائية والثمانية والسداسية

العشرية ، وفى المثال نعين قيمة لعدد صحيح فى متغير a :

```
$a = 1234 ; // Integer
$b= -1235; // negative Integer
$c= 0123 ; // Octal number
$d= 0x12 ; // Hexadecimal
```

الأرقام العشرية Floating point number(Double) :

نـــوع من المتغيرات يأخذ قيمة رقمية تحتوى على علامة عشرية ، وتعتبر اللغة أى نوع من هذه الأرقام غير محاط بعلامتى تنصيص هو من هذه النوع مثل :

```
$a = 1.234 ; // Floating point
$b=1.76;
$c=1.1e2;
```

متغيرات سلسلة النص String :

هـــو متغير يحتوى على حروف أو حروف وأرقام أو أرقام تعامل معاملة النص مثل رقـــم الهاتف ، وتعتبر اللغة أى قيمة موضوعة بين علامتى تنصيص سلسلة حرفية مثل :

```
$s= "String 1458.24" ;
```

يمكـــن تعيين قيمة نصية لأى متغير بوضع النص بين علامتى تنصيص أو علامتى تنصيص مفردة مثال :

```
$a ="Welcome to php" ;
$b = " نص عربى " ;
$c = ' Hello ' ;
```

هناك رموز لا تستخدم مباشرة بين علامتى تنصيص المزدوجة أو المفردة مثل وضع علامة تنصيص فى نص كجملة Hello "world" فوضعها بالشكل التالى $a="Hello "world"";" خطأ فلن يتعرف البرنامج على علامة إغلاق الجملة والعلامة المضمنة ، وطريقة الكتابة الصحيحة هى : $a="Hello "world" "; بوضع مسافة أو علامة قبل

علامـــة التنصـــيص " ، ونفس الأمر ينطبق عند استخدام رموز مثل علامة الدولار داخل جملة نصية .

رموز الهروب Escaped characters

الرموز	الاستخدام
\n	الانتقال لسطر جديد linefeed (LF or 0x0A in ASCII)
\r	يماثل ضغط مفتاح الإدخال carriage return (CR or 0x0D in ASCII)
\t	جدولة أفقية horizontal tab (HT or 0x09 in ASCII)
\\	شرطة عكسية Backslash
\$	علامة الدولار dollar sign
\"	وضع علامة تنصيص مزدوجة double-quote
\[0-7]{1,3}	تتابع الحروف فى نظام ثمانى The sequence of characters matching the regular expression is a character in octal notation
\x[0-9A-Fa-f]{1,2}	تتابع الحروف فى نظام سداسى عشرى The sequence of characters matching the regular expression is a character in hexadecimal notation

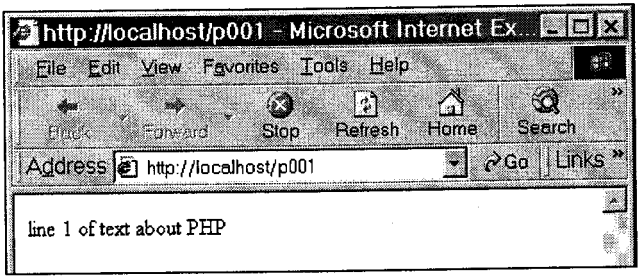
يمكن التعامل مع النصوص كالتالى :

```
$a = "Line1 of text about " ;
echo $a ;
```

بالنسبة للصق النصوص :

```
$a = "line 1 of text about " ;
$a .= "PHP";
echo $a ;
```

بطـبع المتغـيرa نحصـل على جملة Line 1 of text about PHP فعلامة التساوى مسبوقة بنقطة تعنى إضافة فى نهاية المتغير للقيمة التالية .



لدمج متغيرات مثال :

```
$a = 9988 ;
$b = "Your Number $a" ;
```

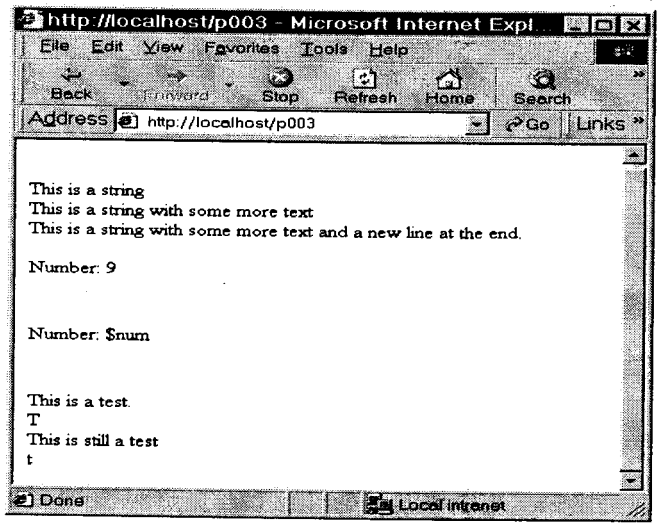
دمجنا قيمة المتغير مع النص فلو طبعنا قيمة $b نحصل على Your Number =9988 .

تربط النصوص مع بعضها البعض باستخدام معامل علامة النقطة (dot) . فاستخدام علامة الجمع (addition) + لا تعمل لهذا الغرض .

```
<?php
/*  تخصيص سلسلة نص لمتغيرAssigning a string. */
$str = "This is a string";
/*  الإضافة للنصAppending to it. */
$str = $str . " with some more text";
/*  طريقة أخرى للإضافة Another append, escaped newline. */
$str .= " and a new line at the end.\n";
/* This string will end up being '<p>Number: 9</p>' */
$num = 9;
$str = "<p>Number: $num</p>";
// لاحظ أن علامة التنصيص المفردة لن تكون صحيحة
/* This one will be '<p>Number: $num</p>' */
$num = 9;
$str = '<p>Number: $num</p>';
/* أول حرف من نصGet First character of a string  */
```

```
$str = 'This is a test.';
$first = $str[0];
/*  آخر حرف من نص Get the last character of a string. */
$str = 'This is still a test';
$last = $str[strlen($str)-1];
?>
```

شـــاهد الـــناتج التالى على شاشة المستعرض بعد أن أضفنا جملة Echo بين كل نص نريد عرضه ، ولاحظ أيضا استخدام علامة /n وعلامة <p> والوصل للحرف الأول من النص والوصول للحرف الأخير فيه :



المصفوفات Arrays

المصفوفة متغير يحتوى عدة قيم ، وهى مجموعة من القيم فى متغير واحد يعبر عنه ككائـــن يحـــتوى على العديد من القيم ، ويمكن التعبير عن قيم المصفوفة بالأرقام أو الأســـماء ، من مهام المصفوفة تيسير البرمجة وتوفير الجهد والوقت ، فيمكن قراءة ملف بجملة واحدة ، مثال :

```
$a[0]="aly";
$a[1]="ahmed";
$c["aaf"]='12';
```

تشـــبه المصـــفوفة الجـــدول بعدد من الخلايا ، مثل متغير المصفوفة $x التالى الذى يحتوى على عدد من الخلايا :

الخلايا	4	3	2	1	0

فالجدول السابق يحتوى خمس خلايا لكل خلية قيمة مثل :

القيم	18	15	14	12	11

لطباعة قيمة الخلية الثالثة (يبدأ عد الخلايا من الصفر فرقم الخلية سيكون ٢) يكون الأمر هو :

```
echo $x[2];
```

أى طباعة محتويات الخلية ليطبع الرقم 14 .

لـــو أردنا تعديل قيمة الخلية الاخيرة رقم ٤ نكتب 30 = [4]$x ، ويمكن أن تحتوى الخلية على قيمة رقمية أو نصية .

عند عدم معرفة عدد خلايا مصفوفة وتريد إضافة قيمة فى نهاية المصفوفة تكتب :

```
$x[] = 22;
```

بدون تحديد رقم الخلية وبالتالى يفترض البرنامج إضافتها فى نهاية المصفوفة .

المصفوفة ذات الأبعاد تحتوى على صفوف مكونة من خلايا مثال :

3	2	1	0
3	2	1	0

للوصول إلى قيمة الخلية المظللة نحتاج تحديدها بمعاملين [2][2]$x أى تحديد الخلية فى الصف رقم ١ (الصف الأول هو رقم صفر) والخلية الثالثة فى الصف .

هناك دوال functions للتعامل مع المصفوفات سوف نستعرضها فيما بعد مع الدوال.

التبديل بين أنواع المتغيرات

تقـــوم لغة PHP بتغيير نوع المتغير عند تغيير القيمة التى يحتويها المتغير فإذا كانت القـــيمة رقمية فسوف تكون عددا صحيحا ، وعند كتابة قيمة نصية فى نفس المتغير يتحول المتغير إلى متغير نص مثال :

```
$a=7; // عدد صحيح
$a="Name" ; // تحويل المتغير إلى نص
$a=9 ; // العودة إلى عدد
```

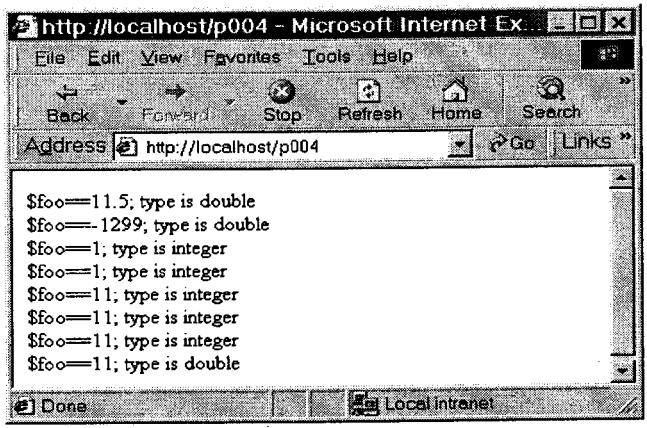
تحويل النصوص String conversion

عندما يكون التعبير نصيا فى البداية فإن نوع المتغير سوف يعتمد على التعبير التالى لجملة التخصيص وبالتالى يمكن تحويل المتغير إلى نوع آخر ، انظر للمثال التالى :

```
$foo = 1 + "10.5";                // $foo is double (11.5)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";

$foo = 1 + "-1.3e3";              // $foo is double (-1299)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";

$foo = 1 + "bob-1.3e3";           // $foo is integer (1)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";

$foo = 1 + "bob3";                // $foo is integer (1)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";

$foo = 1 + "10 Small Pigs";       // $foo is integer (11)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";

$foo = 1 + "10 Little Piggies"; // $foo is integer (11)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";

$foo = "10.0 pigs " + 1;          // $foo is integer (11)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
```

```
$foo = "10.0 pigs " + 1.0;                    // $foo is double (11)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
```

النتيجة سوف تكون فى كل مرة طباعة المتغير مع اختلاف فى القيم ، بناء على طبيعة التغيير فى المتغير ، ويتضح ذلك من النوع الذى نحصل عليه من القيمة gettype للمتغير نفسه بعد تغييره كما فى الناتج الذى يظهر فى المستعرض على الشكل التالى :



العمليات الحسابية والمنطقية على المتغيرات

فى الجدول التالى العمليات الحسابية على المتغيرات :

جمع متغيرين	$a+$b
طرح متغيرين	$a-$b
ضرب متغيرين	$a*$b
قسمة متغيرين	$a/$b
نسبة المتغير الأول للثانى	$a % $b
زيادة واحد لقيمة المتغير	++$a
طرح واحد من قيمة متغير	--$a

المقارنة بين المتغيرات :

التساوى	$a == $b
تساوى المتغيرين من نفس النوع	$a= == $b
متغيران غير متساويين	$a != $b
متغيران غير متساويين أو ليسا من نفس النوع	$a !== $b
المتغير الأول أكبر من المتغير الثانى	$a > $b
المتغير الأول أقل من المتغير الثانى	$a < $b
المتغير الأول أكبر من أو يساوى المتغير الثانى	$a >= $b
المتغير الأول أقل من أو يساوى المتغير الثانى	$a <= $b

مثال :

```
$a=3;
$a +=5;
```

المتغير كان يحمل القيمة ٣ ثم أضفنا إليه القيمة ٥ فتصبح محتويات المتغير بالقيمة ٨ ، ومعنى هذا أن يمكن جمع أى قيمة رقمية على قيمة متغير بكتابة علامة الجمع مع علامة التساوى تليها القيمة المضافة كما فى المثال . مثال آخر :

```
$b="My First";
$b .= "Programming For web developer by PHP";
```

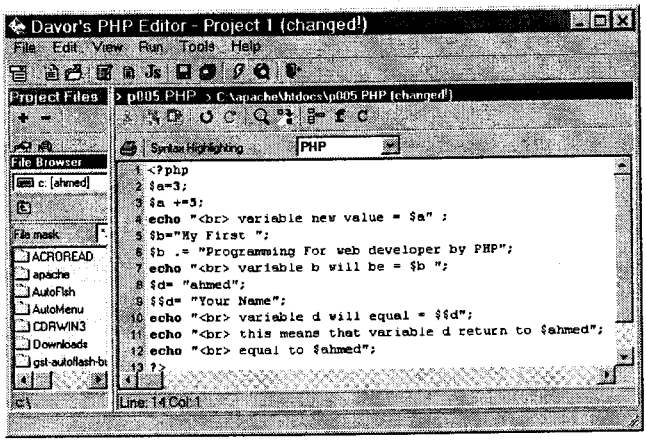
كانـــت قيمة المتغير كلمة My First كسلسلة نص ، وفى السطر التالى أضيفت سلسلة نـــص إلى قيمة سلسلة النص فى المتغير عن طريق كتابة النقطة ثم علامة التساوى يعقبها النص المطلوب إضافته بين علامتى تنصيص . مثال :

```
$d= "Ahmed";
$$d= "Your Name";
```

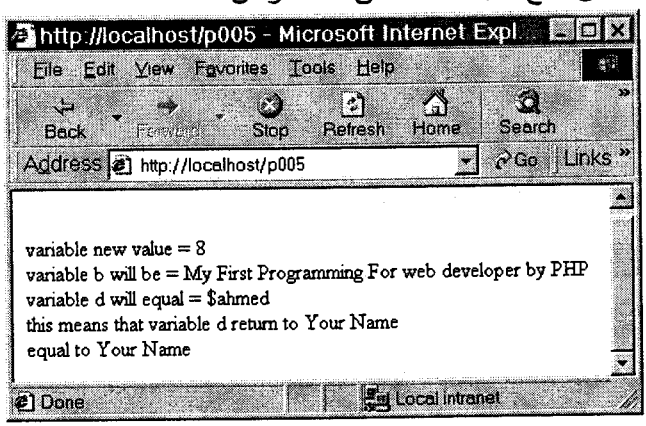
الســـطر الأول سلســـلة نص فى متغير ، بينما يكون السطر الثانى عبارة عن تكوين متغـــير مـــن النص الموجود فى المتغير الأول ووضع قيمة لهذا المتغير الجديد هى محتويات النص Your Name ، أى أن نتيجة تنفيذ هذا هى تكوين المتغير :

```
$a="Ahmed";
$Ahmed= "Your Name" ;
```

ويستخدم هذا الأسلوب عند ربط متغير بمتغير آخر مثل اسم الولوج وكلمة السر . فى الشكل التالى تجميع تلك الأمثلة :



وفى الشكل التالى ناتج تنفيذ المثال على المستعرض :



العوامل Operators والعمليات الرياضية

لتنفـيذ عمليات حسابية مثل الجمع والطرح على متغير $x ومتغير $y ووضع الناتج

فى متغير $z يمكن تنفيذ العمليات التالية :

```
<?php
$x = 12 ;
$y = 6 ;
$z = $x+ $y ;
$z = $x -$y ;
$z = $x* $y ;
$z = $x / $y ;
$z = $x%$y ;
?>
```

عمليات المقارنة Comparison Operators :

$a == $b	المتغيران متساويان
$a ===$b	المتغيران متساويان ومن نفس النوع
$a !=$b	المتغير الأول لا يساوى الثانى
$a > $b	أكبر من
$a < $b	أصغر من
$a >=$b	أكبر من أو يساوى
$a <=$b	أصغر من أو يساوى

عوامل الإضافة والطرح :

إذا كانـــت للمتغـــير $a قيمة ما ;7= $a وأردنا إضافة واحد إليه أو طرح واحد منه يمكن استخدام التعبيرات التالية :

$a++	إرجاع قيمة المتغير مع إضافة واحد لقيمة المتغير $a
++$a	إضافة واحد للمتغير ثم إرجاع القيمة

$a--	إرجاع قيمة المتغير ثم طرح واحد منها
--$a	طرح واحد من قيمة المتغير ثم إرجاع القيمة

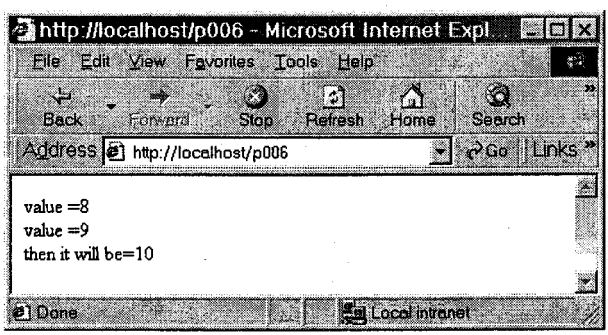
مثال :

```
$a = 8 ;
echo $a++;
أضفنا ١ إلى المتغير لكن التأثير لن يظهر إلا فى مرة تالية .. //
echo ++$a";
أضفنا واحدا للقيمة وظهر تأثير المرة السابقة //
```

فى الشكل التالى استخدام برنامج تحرير لغة PHP لكتابة النص :

```
<?php
$a = 8 ;
echo "value =";
echo $a++;
// return value ..
echo "<BR> value =";
echo $a;
echo "<br> then it will be=" ;
echo ++$a;
?>
```

فى الشكل التالى نتيجة التنفيذ على المستعرض :



العمليات المنطقية Logical Operations :

$a and $b	ضرب منطقى و	تحقق الاثنين
$a or $b	جمع منطقى أو	تحقق كلاهما أو احدهما
$a xor $b	Xor	تحقق أحدهما وليس كلاهما
!$a	نفى لقيمة a	نفى تحقق الشرط
$a || $b	أو	تحقق أحدهما أو كلاهما
$a && $b	و	تحقق كلاهما

العمليات المنطقية للمقارنة بين متغيرات $b , $a مثلا :

$a & $b	And و	إذا تحقق الشرط فى a وفى b معا
$a | $b	Or أو	إذا تحقق الشرط فى $a أو $b
$a^$b	Xor أو مانعة	تحقق شرط فى $a أو $b وليس بكليهما

أولويات العمليات

لكـل عملـية أولوية ، وتنفذ العمليات وفق ترتيب الأولوية فعند حساب قيمة 3*4+2 تكون مساوية 14 وليس ١٨ لأن للضرب أولوية أعلى فينفذ أولا بضرب ٣ فى ٤ ثم جمع القيمة ، ولجمع قيمة ٢ و ٤ وضربها فى ٣ تستخدم الأقواس لتنفيذ عمليات قبل أخرى مثل 3 * (4+2) ، وتتبع الأولويات جدول أولوية المعاملات التالى :

العامل
,
or

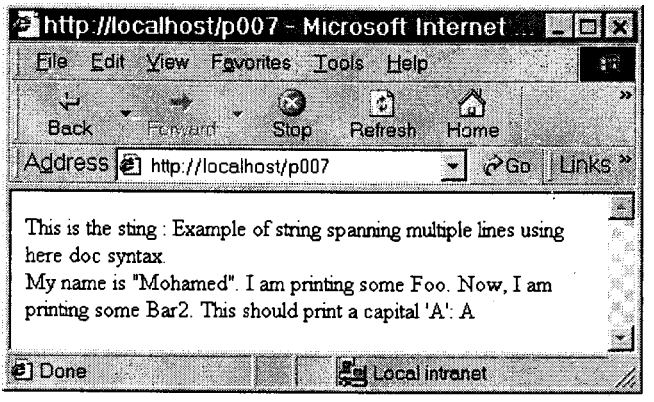
Xor
And
Print
= += -= *= /= .= %= &= |= ^= ~= <<= >>=
? :
||
&&
|
^
&
== != === !==
< <= > >=
<< >>
+ - .
* / %
! ~ ++ -- (int) (double) (string) (array) (object) @
[
New

مثال لاستخدام النصوص الطويلة :

```
<?php
$str = <<<EOD
Example of string
spanning multiple lines
using here doc syntax.<br>
EOD;
Echo "This is the string :";
Echo $str
```

```
/* More complex example, with variables. */

class foo {
    var $foo;
    var $bar;

    function foo() {
        $this->foo = 'Foo';
        $this->bar = array('Bar1', 'Bar2', 'Bar3');
    }
}

$foo = new foo();
$name = 'Mohamed';

echo <<<EOT
My name is "$name". I am printing some $foo->foo.
Now, I am printing some {$foo->bar[1]}.
This should print a capital 'A': \x41
EOT;
?>
```

نتيجة تنفيذ المثال :



الفصل الثالث 3

التحكم [الشروط والتكرار]

تحتوى لغة PHP على أدوات التحكم للتحقق من الشروط كما تحتوى على أدوات لتنفيذ حلقات التكرار ، ودوال تغطى معظم الاستخدامات كما يمكن عمل دول خاصة معرفة بواسطة المبرمج .

تحتوى أدوات التحكم على :

عبارة **if** – عبارة **else** – عبارة **elseif** – عبارة **switch** – حلقة التكرار **while** – حلقة التكرار **for** – حلقة التكرار **do while** .

عبارة الشرط (إذا) if :

تستخدم عبارة الشرط **if** كما فى لغات البرمجة الأخرى للتحقق من شرط معين **if** وإذا كان الشرط صحيحا true يتم تنفيذ سطور محددة ، ويوضع الشرط بين قوسين () ، وتوضع السطور المطلوب تنفيذها بين علامات { } ، ويمكن التخلى عن علامات { } فى حالة وجود سطر واحد فقط .

عبارة else :

تتيح إمكانية وجود إجراء ثان لعدم تحقق الشرط .

عبارة elseif :

يمكن استخدام عبارة **elseif** مع الشرط إذا **if** للتحقق من شرط آخر .

عبارة switch :

تشبه عبـــارة **if** ، لكن يمكن استخدام أكثر من شرطين بإسناد قيمة معينة لحالة case هى بمثابة شرط لتنفيذ الإجراء المحدد لذلك الشرط أو تلك الحالة case .

حلقة التكرار while :

حلقـــة تكـــرار تأخذ شرطا واحدا لتنفيذ ما بين علامات الشرط **{ }** ، والفرق الوحيد بيـــنها وبيـــن الشرط إذا **if** هو تنفيذ الإجراءات طالما كان الشرط صحيحا لأكثر من مرة .

حلقة التكرار do while :

صورة من حلقة التكرار **while** مع فرق التأكد من الشرط وصحته ليتم تنفيذ الإجراء بعده وليس قبله كما فى حلقة التكرار **while** .

اتخاذ القرار (الشروط)

القـــرار أو تنفيذ شرط من العمليات المهمة فى البرمجة وعند تحقق الشرط True يتم تنفيذ عمل ما ، وعند عدم تحقيق شرط False سوف يتم التوجه إلى تنفيذ مهمة أخرى ، ويعد الشرط مقارنة بين قيمتين ، وتستخدم دالة الشرط (إذا If) فى اللغة لتنفيذ هذه المقارنة وصيغة الدالة هى :

```
If (condition) {
Commands }
```

بمعنى : إذا (تحقق الشرط) ، قم بعمل ما بين العلامتين { } ، وتكون السطور التالية فى البرنامج هى تلك التى تنفذ بعد اختبار وتنفيذ دالة الشرط ، مثال :

```
?>php
$b="good";
if ($b=="good") {
echo $b;
}
Print "<BR> Good";
```

<?

هناك دالة الشرط الأخرى التى تنفذ مهمة أخرى عند عدم تحقق الشرط (إذا ، وغير ذلك) If Else ، ويقوم الجزء الأول من الدالة بالتحقق من الشرط ، فإذا تحقق يتم تنفيذ مهمة ما ثم يقفز تنفيذ البرنامج إلى خارج جزء الدالة .

?>php

```
$c="bad";
if ($c=="good")
 {
echo $c;
}
Print "<BR> Not Good";
```

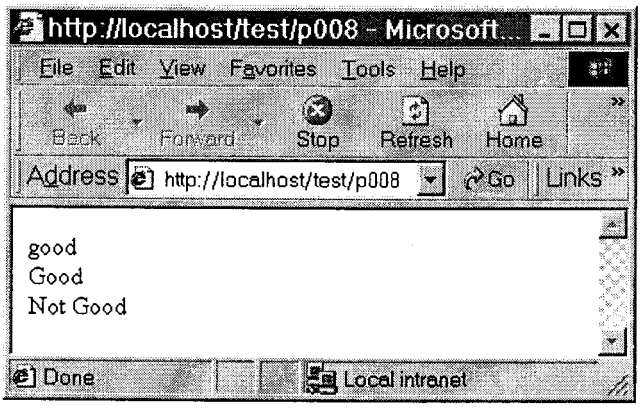
<?

إذا لـــم يـــتحقق الشرط Else يتم تنفيذ عملية أخرى ثم يقفز تنفيذ البرنامج إلى خارج جزء الدالة ، بمعنى تنفيذ أوامر فى حالة تحقق الشرط ، أو تنفيذ أوامر فى حالة عدم تحقق الشرط .

مثال :

```
1  <?php
2  $b="good";
3  if ($b=="good") {
4  echo $b;
5  }
6  Print "<br> Good";
7  $c="bad";
8  if ($c=="good") {
9  echo $c;
10 }
11 Print "<BR> Not Good";
12 ?>
```

النتيجة :



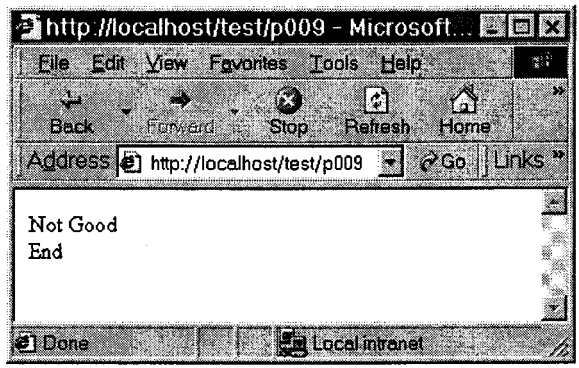
الصيغة العامة لهذه الدالة هى :

```
If (condition) {
 Commands
 }
else {
commands

}
```

مثال :

```
<?php
$b="bad";
if ($b=="good") {
echo $b;
}
else {
echo "Not Good";
}
Print "<BR>End";
?>
```

النتيجة :



لدالـــة الشـــرط أيضا صيغة أخرى للتحقق من الشرط If فإذا تحقق الشرط يتم تنفيذ مجموعـــة مـــن الأوامر ثم الخروج من كتلة الشرط إلى مجموعة الأوامر التالية فى الـــبرنامج ، وإذا لـــم يتحقق الشرط يتم اختبار وفحص الناتج elseif للتحقق من تنفيذ شـــرط فيه ، فإذا تحقق الشرط الثانى يتم تنفيذ الأوامر التالية له ثم الخروج من كتلة الشـــرط إلـــى ســـياق البرنامج ، وإذا لم يتحقق الشرط Else يتم تنفيذ مجموعة من الأوامر ثم الانتقال إلى سياق البرنامج ، والصيغة العامة لهذه الدالة هى :

```
If (condition1) {
Commands

 }
elseif (condition) {
command
}
else {
commands

}
```

مثال :

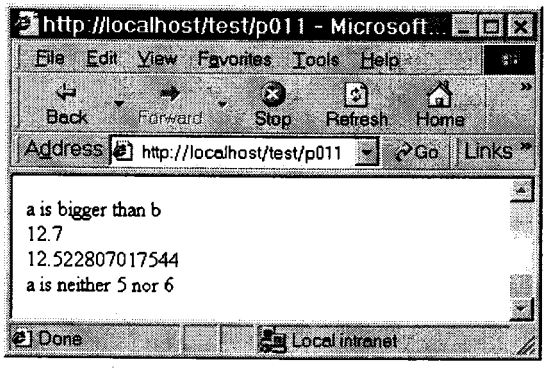
```
<?php
if ($b=="Ahmed") {
echo "Welcome Ahmed";
}
elseif ($b=="Aly") {
```

```
echo "Welcome Aly";
}
else {
echo "Welcome Any one <BR>";
}
Print "End ";
?>
```

النتيجة بالطبع هى طباعة Welcome Any one على سطر ثم تليها فى السطر التالى طباعة كلمة End لعدم وجود قيمة فى المتغير $b .

```
<?php
$a= 12.7;
$b= 142.76/11.4 ;
if ($a > $b) {
    print "a is bigger than b";
    echo "<BR> $a <BR> $b" ;
} elseif ($a == $b) {
    print "a is equal to b";
} else {
    print "a is smaller than b";
}
?>
 <?php if ($a == 5): ?>
A is equal to 5
 <?php endif; ?>
<?
if ($a == 5):
    print "a equals 5";
    print "...";
elseif ($a == 6):
    print "a equals 6";
    print "!!!";
else:
    print "<BR> a is neither 5 nor 6";
endif;
?>
```

النتيجة :



دالة الشرط للتبديل بين القيم Switch :

تستخدم دالة التبديل للاختيار بين عدة حالات لاحتمالات مختلفة ، وهى تقلل من تكرار خطوات البحث عن تحقق شرط ما ، والصيغة العامة لها هى :

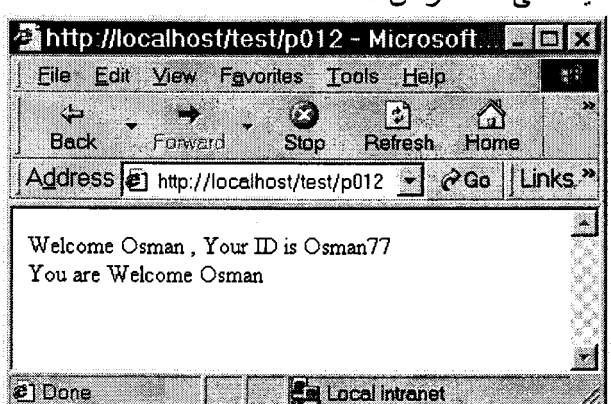
```
Switch (variable) {
Case "string1" :
Commands1
Break;

Case "string2" :
Commands2
Break;

Case "string3" :
Commands3
Break;

Case "string4" :
Commands4
Break;

.....
}
```

مثال :

```
<?php
$a="Osman";
Switch ($a) {

Case "Aly" :
Echo " Welcome Aly , Your ID is Aly123";
Break;

Case "Ahmed" :
Echo " Welcome Ahmed , Your ID is Ahmed321";
Break;
```

```
Case "Omar" :
Echo " Welcome Omar , Your ID is Omar88";
Break;

Case "Khaled" :
Echo " Welcome Khaled, Your ID is Khaled22";
Break;

Case "Osman" :
Echo " Welcome Osman , Your ID is Osman77";
Echo " <BR> You are Welcome Osman ";
Break;
}

?>
```

النتيجة عند تنفيذها فى المستعرض :



التكرار

تقوم حلقة التكرار بتنفيذ عمل أو مجموعة أعمال مهام برمجية لعدد من المرات يعتمد على تحقق شرط ما .

تستخدم عبارة While فى بداية الحلقة ليتم التأكد من تحقق الشرط ، فإذا تحقق الشرط

يتم الدخول فى حلقة لتكرار وتنفيذ مجموعة الأوامر الموجودة فى كتلة حلقة التكرار ، ثم يتم الرجوع مرة أخرى إلى بداية حلقة التكرار وفحص تحقق الشرط مرة أخرى ، وإذا لم يتحقق الشرط يتم الخروج من كتلة التكرار ثم يتم تنفيذ الأوامر التالية للكتلة فى سياق البرنامج الطبيعى ، والصيغة العامة لحلقة التكرار While هى :

```
While (condition) : Commands ..
..
endwhile;
```

أو بصورة :

```
while (expr): statement ... endwhile;
```

وقد تكتب بصورة :

```
while (expr) statement
```

مثال :

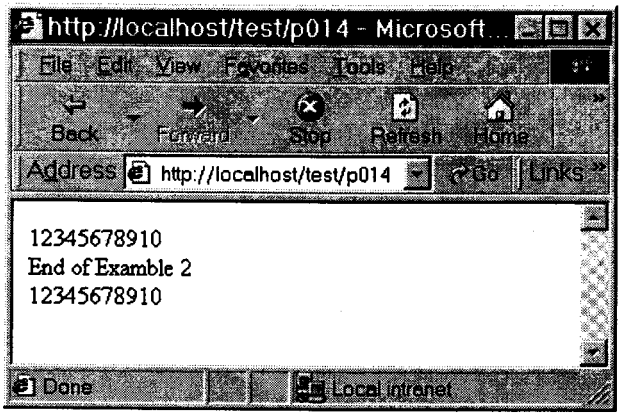
```
<?php
$b=5;
While ($b<15) :
++$b;
Echo "<BR> $b" ;
endwhile;
Print "End ";
?>
```

تقوم الحلقة بطباعة الأرقام من ٦ إلى ١٥ ، وتتحقق الحلقة من الشرط في بداية تنفيذ الحلقة .

```
<?
/* example 2 */
$i = 1;
while ($i <= 10) {
    print $i++;  /* the printed value would be
                    $i before the increment
                    (post-increment) */
}
echo "<BR> End of Example 2 <BR>" ;
/* example 3 */
```

```
$j = 1;
while ($j <= 10):
   print $j;
   $j++;
endwhile;
?>
```

النتيجة فى المستعرض :



حلقة التكرار الأخرى Do while تقوم أيضا بالتكرار باستمرار تنفيذ الحلقة Do طالما يـتحقق شرط While الفحص فى نهاية الحلقة ، فإذا تحقق الشرط يرجع لبداية الحلقة وتنفـيذها حـتى يصـبح الشـرط غير متحقق فيتم الخروج من الحلقة إلى البرنامج الطبيعى، وتنفيذ الشرط فى نهاية الحلقة يسبب تنفيذ الحلقة مرة واحدة قبل التحقق من الشرط ، وتكتب الحلقة بالصيغة التالية :

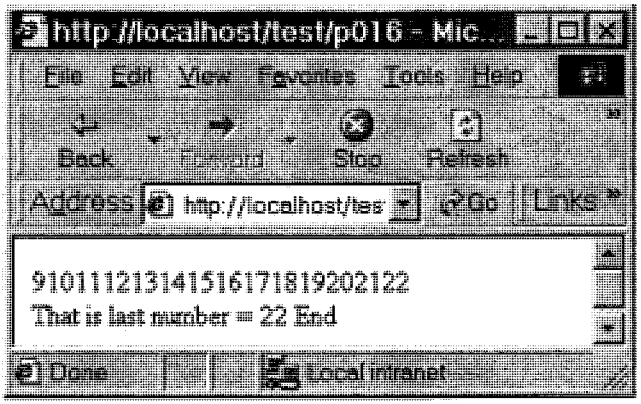
```
Do {
Commands
} while (condition)
```

مثال :

```
<?php
$b=8;
do {
++$b;
```

```
echo $b;
} while ($b<=21);
Print "<BR> That is last number = $b End ";
?>
```

النتيجة فى المستعرض :



حلقة تكرار من For تستمر فى تنفيذ التكرار مادام الشرط متحققا ، ويتم التحقق من الشرط فى بداية الحلقة بتحديد بداية الحلقة وطريقة التصرف والشرط الذى يتم بناء عليه تنفيذ الحلقة ، وتكتب بالصيغة :

```
For (begin; Condition; action)
{Commands }
```

مثال :

```
<?php
For ($I=1 ; $I<20; $I++) {
Print $I;
}
?>
```

تبدأ الحلقة من المتغير $I مساويا للرقم ١ وبشرط أن هذا الرقم لا يزيد عن ٢٠ ثم زيادة قيمة المتغير بعدد واحد ، وتستمر الحلقة لطباعة الرقم بتصاعد حتى ينتهى تحقق الشرط عند رقم ١٩ .

حلقة التكرار كل واحد Each أو لكل واحد Foreach تستمر فى الحلقة للبحث

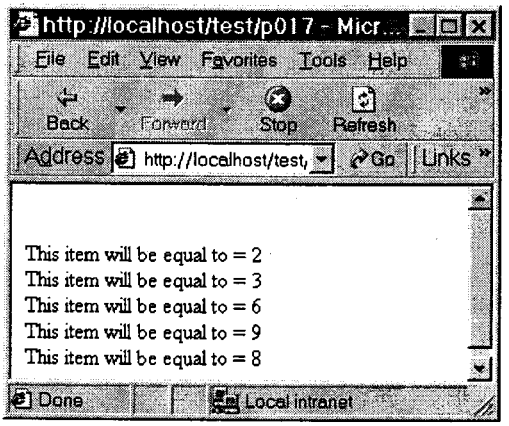
فـــى عناصـــر كائن مثل مصفوفة حتى تنتهى من كل العناصر ، وتفيد فى العمل مع الجداول ، ولها الصيغة :

```
Foreach (object as variable) {
Commands
 }
```

مثال :

```
<?php
$b= array(2, 3 , 6 , 9, 8);
foreach ($b as $c) {
echo "<BR> This item will be equal to " ;
Print $c;
}
?>
```

فـــى المثال يحتوى السطر الثانى على مصفوفة موضوعة فى المتغير $b ، ثم طلب الســـطر الثالـــث وضع كل عنصر فى المصفوفة فى المتغير $c واحدا وراء الآخر بالتكرار ثم طباعة المتغير لطباعة محتويات المصفوفة واحدا بعد الآخر .



كسر Break حلقة التكرار

للخـــروج من حلقة التكرار قبل انتهائها يستخدم أمر كسر الحلقة Break للخروج من

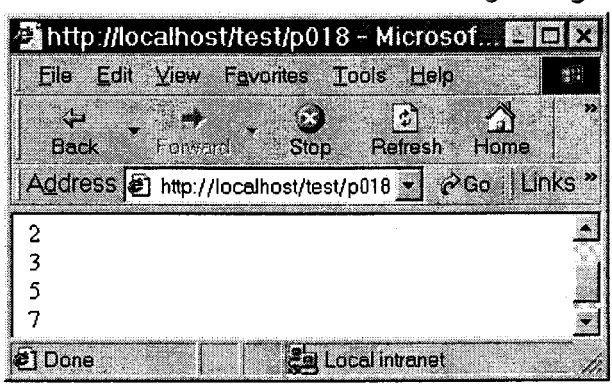
الـتكرار وينـتقل التنفـيذ للخطوة التالية لكتلة الحلقة فى سياق البرنامج الطبيعى ، والصيغة العامة للأمر هى :

```
Break;
```

مثال :

```
<?php
$d= array (2, 3, 5, 7, 8, 9);
foreach ($d as $n) {
If ($n==8) {
Break;
}
Print "<BR> $n";
}
?>
```

بعـد إنشاء مصفوفة وسحب العناصر من المصفوفة يتم مقارنة عنصر المصفوفة مع قـيمة ٨ فـإذا تحقق الشرط يتم كسر حلقة التكرار ، وفى غير ذلك يتم تكرار سحب وعرض عناصر المصفوفة .



أمثلة :

```
/* For example */
for ($i = 1; $i <= 10; $i++) {
    print $i;
```

```
}
Echo "<BR> END <BR>";
/* For example */
 for ($i = 1;;$i++) {
    if ($i > 10) {
        break;
    }
    print $i;
}
 Echo "<BR> END <BR>";
/* For example */
$i = 1;
for (;;) {
    if ($i > 10) {
        break;
    }
    print $i;
    $i++;
}
 Echo "<BR> END <BR>";
/* For example */
for ($i = 1; $i <= 10; print $i, $i++) ;
```

مثال آخر :

```
<?
/* foreach example : value only */
$a = array (1, 2, 3, 17);
foreach ($a as $v) {
   print "Current value of \$a: $v.\n";
echo "<BR>" ;
}
echo "<BR> End <BR>" ;

/* foreach example : value (with key printed for illustration) */
$a = array (1, 2, 3, 17);
$i = 0; /* for illustrative purposes only */
foreach($a as $v) {
```

```
echo "<BR>" ;
   print "\$a[$i] => $k.\n";
}
echo "<BR> End <BR>" ;

/* foreach example : key and value */
$a = array (
   "one" => 1,
   "two" => 2,
   "three" => 3,
   "seventeen" => 17
);
foreach($a as $k => $v) {
echo "<BR>" ;
   print "\$a[$k] => $v.\n";
}
echo "<BR> End <BR>" ;

?>
```

مثال التوقف :

```
<?
$arr = array ('one', 'two', 'three', 'four', 'stop', 'five');
while (list (, $val) = each ($arr)) {
   if ($val == 'stop') {
      break;    /* You could also write 'break 1;' here. */
   }
   echo "$val<br>\n";
}

/* Using the optional argument. */
$i = 0;
while (++$i) {
   switch ($i) {
   case 5:
      echo "At 5<br>\n";
      break 1;  /* Exit only the switch. */
   case 10:
```

```
        echo "At 10; quitting<br>\n";
        break 2;  /* Exit the switch and the while. */
    default:
        break;
    }
}
 ?>
```

مثال الاستمرار Continue :

```
<?
$i = 0;
while ($i++ < 5) {
    echo "Outer<br>\n";
    while (1) {
        echo "  Middle<br>\n";
        while (1) {
            echo "  Inner<br>\n";
            continue 3;
        }
        echo "This never gets output.<br>\n";
    }
    echo "Neither does this.<br>\n";
}
?>
```

الطلب Require والتضمين Include

مـن المهـم تنظيم ملفات الموقع وتقسيم الصفحة ، وتوفر لغة PHP دالتين مختلفتين لـلإدراج يمكـن عـن طريقهما إعادة استخدام ملفات وتنظيم الموقع وتيسير صيانته وتعديله ، الدالة الأولى هى دالة الطلب require والثانية هى دالة التضمين include ، وعـن طريق استخدام أى منها يمكن إدراج شفرة PHP أو نص أو HTML ، ودوال functions أو فئات classes وكائنات .

يسـتخدم الطلب والتضمين لطلب أو تضمين (استدعاء) ملف من ملفات لغة بى اتش

بى PHP فى البرنامج ، فعندما يكون البرنامج كبيرا ويقوم بعدة مهام يمكن تقسيم هذه المهام إلى أجزاء ووضع كل جزء فى ملف ثم طلب الملف عند الحاجة بالصيغة :

```
Require ("Filename") ;
Include ("Filename") ;
```

مثال :

```
Require ("file1.php") ;
Include ("c:\myprog\php\file2.php") ;
```

عند استدعاء ملف يتم تنفيذ جميع الأوامر الموجودة فيه مع التعامل مع المتغيرات المعرفة فيه كما لو كانت بالبرنامج الأصلى .

دالة التضمين include()

يتشابه استخدام دالة الطلب require مع استخدام دالة التضمين Include مع بعض الاختلافات إذ تقلل الحمل على المستضيف ، كما يمكن أن تسترجع return من دالة include قيما تسند إلى متغير آخر .

دالة الطلب require()

عند تنفيذ ملف يحتوى على طلب ملف آخر سيتم استخدام محتويات الملف الآخر كأنها جزء من الملف القائم بالطلب ، مثال :

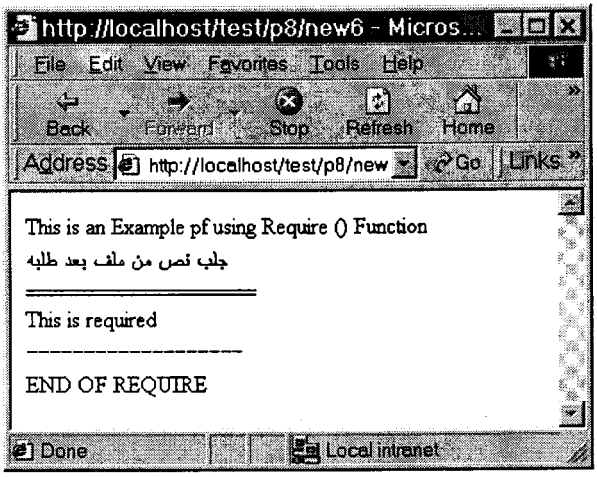
سنحفظ الشفرة التالية فى ملف باسم New6.php :

```
<?php
Echo " This is an Example pf using Require () Function <BR> ";
Echo "  جلب نص من ملف بعد طلبه <BR> " ;
Echo "================== <BR> " ;
Require ("new7.php") ;
?>
```

سنحفظ الشفرة التالية فى الملف new7.php :

```
<?php
Echo " This is required <BR> ";
Echo " ------------------------- <BR>" ;
```

```
Echo "END OF REQUIRE " ;
?>
```



عمليات الإدراج لا تقف على مثال بسيط ، فعند وجود برنامج ضخم تستخدم هذه الطريقة للإدراج وتنظيم العمل والتعديل .

الدوال Functions

توفر الدوال الوقت والجهد ، وتؤمن تنسيق وتطوير وتنظيم البرنامج وإعادة الاستخدام ، وتحتوى لغة PHP مجموعة كبيرة من الدوال منها دوال رياضية Mathematical Functions ، ودوال نظام الملفات File System للتحكم بالملفات و المجلدات ، ودوال الرسوم ، ودوال قواعد البيانات بصفة عامة مثل دوال قاعدة البيانات Mysql ، ودوال المصفوفات Array وغيرها من الدوال الكثيرة .

الدالة المعرفة

الدالة المعرفة هى مجموعة من الأوامر والمتغيرات المستقلة داخل برنامج يتم تنفيذها عند طلبها باستدعائها عن طريق كتابة اسم الدالة وإرسال قيم المتغيرات إليها فتعيد ناتج تنفيذ العملية ، ويمكن إنشاء وتعريف دوال مخصصة للقيام بمهام معينة .

يبدأ تعريف الدالة بكلمة Function تعقبها كلمة اسم الدالة ، يعقبها قوسان توضع بينهما متغيرات الدالة ، ثم يعقب ذلك علامة بداية كتلة الدالة { ، ثم نكتب مجموعة الأوامر التى تنفذها الدالة ، وبعد مجموعة الأوامر تكتب كلمة الرجوع Return لإرجاع القيم الناتجة عن الدالة ، ثم تنتهى كتلة الدالة بكتابة قوس إغلاق كتلة الدالة } بالصيغة التالية :

```
Function Fname (variable1, variable2, ) {
```

```
Commands
...
Return variables
}
```

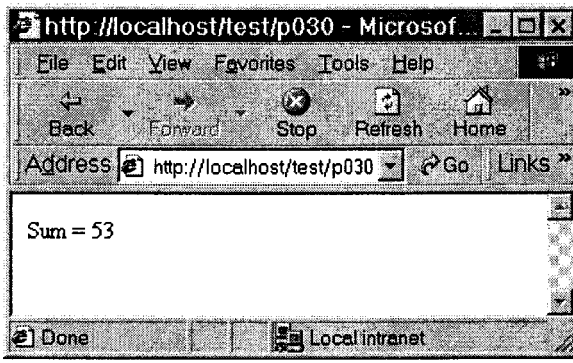
دوال المستخدم المعرفة أيضا يمكن تعريفها بالطريقة التالية :

```
function foo ($arg_1, $arg_2, ..., $arg_n) {
    echo "Example function.\n";
    return $retval;
}
```

مثال :

```
<?
$a = 11; /* global scope */
$b = 42;
Function Test () {
   echo $a; /* reference to local scope variable */
}
Test ();

Function Sum () {
   global $a, $b;
   $b = $a + $b;
}
Sum ();
echo "Sum = $b";
?>
```

النتيجة :

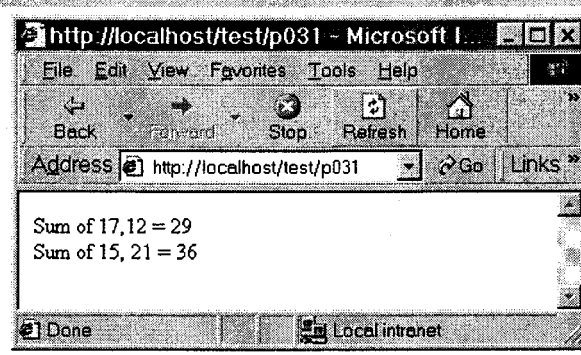


لاحظ أن جملة echo $a; لن تطبع نتيجة لأن المتغير محلى فى الدالة Test() .

مثال :

```
<? php

Function SumNo ($n1, $n2)
 {
$s=$n1+$n2;
Return $s;
}
Print "Sum of 17,12 = ".SumNo (17,12);
Print "<BR> Sum of 15, 21 = ".SumNo(15,21);

?>
```

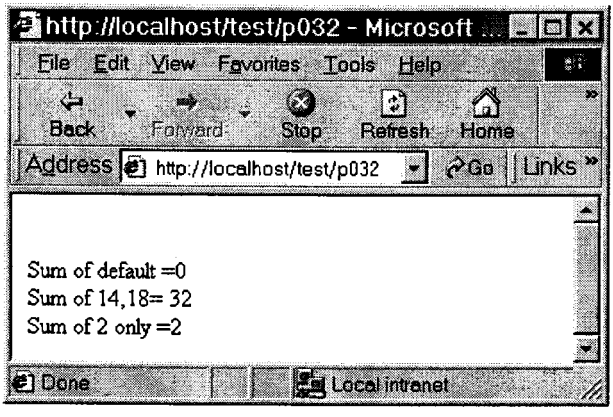
النتيجة :



يحدث الخطأ عند استدعاء الدالة بدون تحديد قيم المتغيرات ، أو عند عدم تحديد كافة قيم المتغيرات فى الدالة ، لذلك يمكن وضع قيم افتراضية مناسبة حتى لا يتسبب ذلك فى حدوث خطأ مثال :

```
<? php
Function SumNo ($n1=0, $n2=0)
{
$s=$n1+$n2;
Return $s;
}
Print "Sum of default =".SumNo ();
Print "<BR> Sum of 14,18= ".SumNo (14,18);
Print "<BR> Sum of 2 only =".SumNo (2);

?>
```

فى المثال تم وضع قيم افتراضية للمتغيرين بقيمة الصفر ، وعند طلب الدالة بدون قيم ترجع الصفر ، وعند طلباها بقيمتين ترجع الناتج الصحيح للجمع ، وعند طلبها بقيمة واحدة ترجع قيمة بدون خطأ .



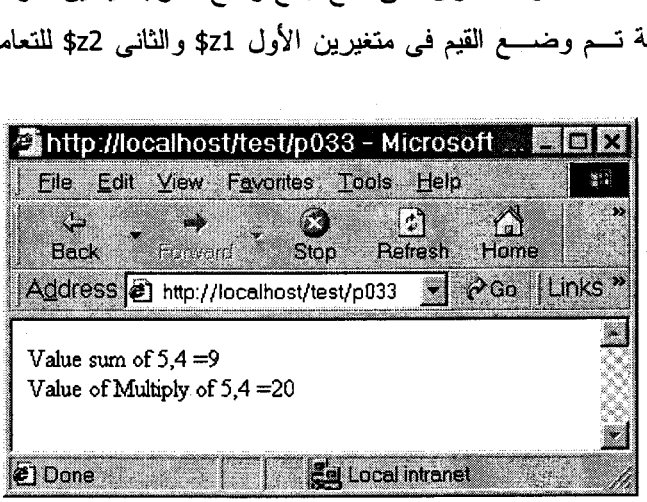
فى المثال التالى تعيد الدالة قيمها فى مصفوفة :

```
<?php
Function SumNo ($n1=0, $n2=0) {
```

```
$s=$n1+$n2;
$m=$n1*$n2;
Return array ($s, $m) ;
}
List ($z1, $z2)= SumNo(5, 4) ;
Print "Value sum of 5,4 =".$z1 ;
Print "<BR> Value of Multiply of 5,4 =".$z2 ;
?>
```

يعـيد المـثال مصفوفة تحتوى على ناتج جمع وناتج ضرب قيمتين ، وعند استدعاء المصـفوفة تـم وضـع القيم فى متغيرين الأول $z1 والثانى $z2 للتعامل معهما .

النتيجة:

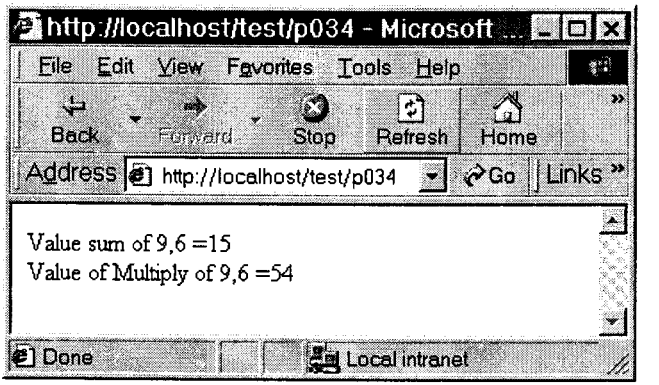


يمكـن استخدام متغيرات عامة فى داخل الدالة لاستخدام هذه المتغيرات فى البرنامج خـارج الدالـة دون حاجـة إلـى تخزينها فى متغيرات أخرى ، بمعنى زيادة مدى المتغيرات إلى خارج الدالة لتكون عامة فى متناول البرنامج ، مثال :

```
<?php
Function SumNo ($n1=0, $n2=0) {
$GLOBALS ["s"]=$n1+$n2; // المتغير العام variable $s
$GLOBALS ["m"]=$n1*$n2; // المتغير العام variable $m
}
SumNo(9, 6) ;
```

```
Print "Value sum of 9,6 =".$s ;
Print "<BR> Value of Multiply of 9,6 =".$m ;
?>
```

النتيجة :



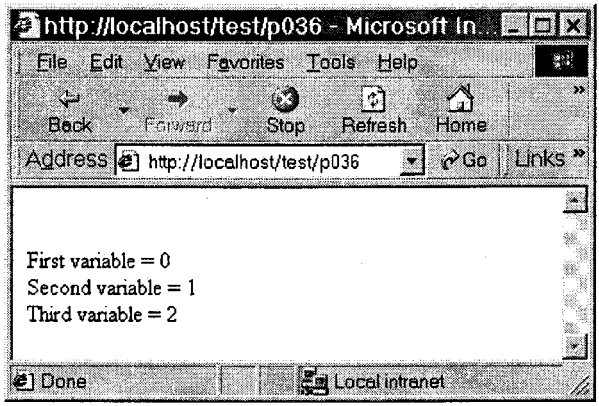
مثال :

```
function square ($num) {
    return $num * $num;
}
echo square (4);   // outputs '16'.
```

لا يمكن إرجاع عدة قيم من الدالة لكن يمكن الحصول على نتائج مشابهة عند إرجاع لائحة list مثال :

```
<?
function small_numbers() {
    return array (0, 1, 2);
}
list ($zero, $one, $two) = small_numbers();
echo "<BR> First variable = $zero" ;
echo "<BR> Second variable = $one" ;
echo "<BR> Third variable = $two" ;
?>
```

النتيجة :



دوال التاريخ والوقت Date and Time Functions

هـناك العديد من الدوال التى تعمل مع الوقت والتاريخ منها دالة التاريخ Date ومنها دوال أخرى مثل :

دالة فحص التاريخ checkdate .

دالة التاريخ date .

دالة الحصول على معلومات التاريخ والوقت getdate .

دالة الحصول على التوقيت الحالى فى اليوم gettimeofday .

دالة تنسيق الوقت gmdate .

دالة توقيت يونكس gmmktime .

دالة تنسيق الوقت على الأوضاع المحلية gmstrftime .

دالة الوقت المحلى localtime .

دالة الوقت لنظام يونكس بالمايكروثانية microtime .

دالة التاريخ لنظام يونكس mktime .

دالة تنسيق الوقت والتاريخ المحلى على الإعدادات المحلية strftime .

دالة الوقت time للحصول على وقت يونكس .

دالة إعراب الوقت strtotime .

فى الغالبية العظمى من الأوضاع تستخدم دالة الوقت التى تكون على الصورة :

string date (List of characters [, int timestamp])

وتعيد سلسلة نص بتنسيق يعتمد على مجموعة الحروف الموضوعة فى الجملة البرمجية ، حيث تستبدل كلمات لائحة الحروف List of characters بمتغير أو مجموعة من متغيرات على شكل حروف .

يمكن باستخدام التاريخ والتوقيت معرفة التاريخ والوقت ، وتأخذ دالة التاريخ Date صيغة بسيطة على شكل :

Date("List of characters")

حيث تستبدل كلمات لائحة الحروف List of characters بمتغير أو مجموعة من المتغيرات التالية على شكل حروف :

بيانها كالتالى :

حرف	الوظيفة
a	قبل الظهر am أو بعد الظهر pm
A	قبل الظهر AM أو بعد الظهر PM
B	عينة توقيت الإنترنت Swatch Internet time
D	اسم اليوم مختصرا مثل Mon
d	رقم اليوم فى الشهر من 01 إلى 31
F	الاسم الكامل للشهر مثل April
G	ساعة Hour بتنسيق 12 ساعة بدون صفر مسبق مثل "1" to "12" .
G	الساعة Hour بتنسيق 24 ساعة بدون صفر مسبق مثل "0"

حرف	الوظيفة
	. to "23"
H	الساعة مع صفر مسبق من 01 إلى 12
h	الساعة من 00 إلى 23
i	الدقائق من 00 إلى 59
I	إضافة 1 للتوقيت الصيفى ، وصفرا للتوقيت الشتوى
j	رقم اليوم من الشهر من 1 إلى 31
l	حرف L صغير لاسم اليوم الكامل مثل Sunday
M	الاسم المختصر للشهر مثل Dec
m	رقم الشهر من 01 إلى 12
n	رقم الشهر من 1 إلى 12
s	عدد الثوانى من 00 إلى 59
S	امتداد رقم اليوم مثل nd ، أو th
t	عدد أيام الشهر الحالى مثل 28, 29, 30 ,31
T	إعدادات توقيت المنطقة مثل MDT
U	الثوانى منذ فترة
w	رقم يوم الأسبوع يبدأ بيوم الأحد 0 والاثنين 1 حتى السبت 6
Y	السنة بأربعة أرقام
y	السنة برقمين
z	رقم اليوم فى السنة
Z	إزاحة توقيت المنطقة بالثوانى

يمكن دمج عدة متغيرات للحصول على نماذج متعددة من التاريخ والوقت ، مثال :

Print date ("j/n/Y");

نتيجة السطر طباعة اليوم والشهر والسنة بالأرقام مثل 15/7/2004 .

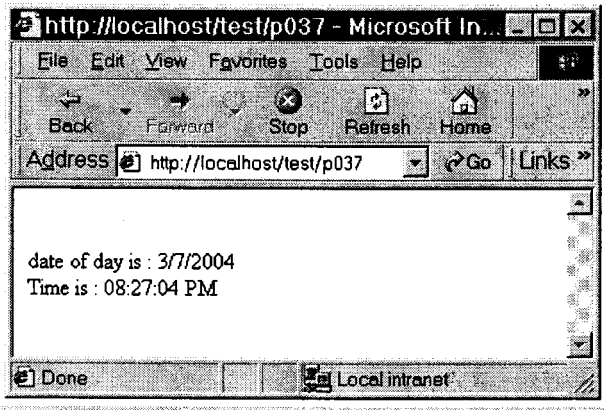
Print date ("h:i:s A");

نتيجة المثال طباعة الوقت بالساعة والدقيقة والثانية قبل الظهر أو بعده مثل 07:10:22 AM .

مثال يمكن أن يكتب على الشكل :

```
<?php
Echo "<BR> date of day is : ";
Print date ("j/n/Y");
Echo "<BR> time is : ";
Print date ("h:i:s A");
?>
```

النتيجة هى :

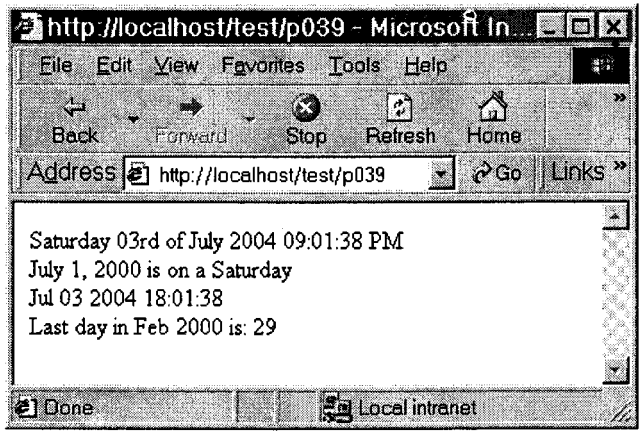


مثال :

```
Print (date ("l dS of F Y h:i:s A"));
print ("<BR> July 1, 2000 is on a " . date ("l", mktime(0,0,0,7,1,2000)));

// Example Gmstrftime() example
echo gmstrftime ("<BR> %b %d %Y %H:%M:%S");

// Example Last day
```

```
$lastday = mktime (0,0,0,4,-31,2000);
echo strftime ("<BR> Last day in Feb 2000 is: %d <BR> ", $lastday);
```

النتيجة :



يمكـــن استخدام أسماء الشهور العربية وأيام الأسبوع العربية عن طريق عدة أساليب منها معرفة قيمة الشهر أو اليوم مع دالة التبديل Switch ، ففى أيام الأسبوع تعيد دالة الـــتاريخ مـــع استخدام المتغير w قيمة تساوى 0 ليوم الأحد ، وقيمة 1 ليوم الاثنين ، وهكذا ، وبالتالى يمكن استخدام المثال التالى :

```
?>php
$d= date("w");
Switch ($d) {

Case 0 :
Echo "الأحد" ;
Break ;

Case 1 :
Echo "الاثنين" ;
Break ;

Case 2 :
```

```
Echo "الثلاثاء" ;
Break ;

Case 3 :
Echo "الأربعاء" ;
Break ;

Case 4 :
Echo "الخميس" ;
Break ;

Case 5 :
Echo "الجمعة" ;
Break ;

Case 6 :
Echo "السبت" ;
Break ;
}
>?
```

دوال التعامل مع الملفات File Functions

دوال نظام الملفات File system functions هى دوال تستخدم للتعامل مع الملفات وتحتوى على مجموعة كبيرة من الدوال التى تتعامل مع نظام الملفات وهى:

دالة الاسم الأساسى basename وتعيد اسم ملف فى مسار .

دالة تغيير مجموعة ملف chgrp .

دالة تغيير حالة ملف chmod (Changes file mode) .

دالة تغيير ملكية ملف chown (Changes file owner) .

دالة مسح مخبأ ملف clearstatcache (Clears file stat cache) .

دالة نسخ ملف copy لنسخ ملف Copies file .

دالة حذف delete .

دالة اسم مجلد أو دليل dirname فى مسار .

دالة مساحة القرص الخالية diskfreespace .

دالة إغلاق ملف fclose مفتوح .

دالة فحص الوصول لنهاية ملف feof .

دالة مبشرة الخرج لملف fflush .

دالة الحصول على رمز Character من مؤشر ملف fgetc .

دالة الحصول على سطر line من مؤشر ملف fgetcsv .

دالة الحصول على سطر من مؤشر الملف fgets .

دالـــة الحصول على سطر من مؤشر ملف fgetss وتجريد علامات لغة النص المتشعب HTML tags .

دالة ملف file تقرأ الملف إلى مصفوفة .

دالة فحص وجود ملف file_exists .

دالة الحصول على آخر وقت وصول للملف fileatime .

دالة معرفة تغيير وقت ملف filectime .

دالة مجموعة ملف filegroup .

دالة معرفة وقت تعديل ملف filemtime .

دالة معرفة ملكية ملف fileowner .

دالة معرفة أذونات ملف fileperms .

دالة معرفة حجم ملف filesize .

دالة معرفة نوع ملف filetype .

دالة قفل ملف flock .

دالة فتح ملف أو عنوان موحد fopen (File or URL) .

دالة إخراج كل البيانات المتبقية إلى مؤشر ملف fpassthru .

دالة الكتابة إلى مؤشر ملف fputs .

دالة قراءة ملف fread بصيغة ثنائية آمنة .

من دوال نظام الملفات أيضا الدوال (fseek, fstat, ftell, ftruncate) .

دالة الكتابة الثنائية الآمنة fwrite .

دالة وضع مخزن مؤقت للملف set_file_buffer .

دالة معرفة ما إذا كان اسم الملف فى الدليل is_dir .

دالة معرفة ما إذا كان الملف تنفيذيا is_executable .

دالة معرفة ما إذا كان الاسم هو لملف is_file .

دالة معرفة ما إذا كان اسم الملف هو رمز رابطة is_link .

دالة معرفة ما إذا كان الملف قابلا للقراءة is_readable .

دالة معرفة ما إذا كان الملف قابلا للكتابة عليه is_writeable .

دالة معرفة ما إذا كان الملف يتم تحميله is_uploaded_file عبر خدمة إرسال بروتوكول النص المتشعب HTTP POST .

دالة الرابط link لإنشاء رابط .

دالة معلومات الارتباط linkinfo .

دالة إنشاء دليل mkdir .

دالة نقل ملفات محملة إلى مكان جديد move_uploaded_file .

دالة إغلاق عمليات مؤشر ملف pclose .

دالة فتح عمليات مؤشر ملف popen .

دالة إخراج ملف readfile .

دالة إعادة رمز رابط readlink .

دالة تغيير اسم ملف rename .

دالة تغيير موضع مؤشر ملف rewind .

دالة حذف دليل rmdir .

دالة معلومات الملف stat .

دالة معلومات ملف أو رمز ارتباط lstat .

دالة المسار المطلق realpath .

دالة إنشاء رمز ارتباط symlink .

دالة إنشاء اسم وحيد tempnam .

دالة إنشاء ملف مؤقت tmpfile .

دالة وضع تعديل وقت ملف touch .

دالة تغيير القناع umask الحالى .

دالة حذف ملف بارتباط unlink .

تشـــمل عمليات الملفات إنشاء أو نسخ أو حذف أو معرفة نوع أو اسم ملف أو (أو مجلد) :

تستخدم دالة اسم القاعدة Basename لمعرفة اسم ملف لغة PHP فى مسار ما مثال :

```
Echo basename ("myprogs/php/index.php") ;
```

تكون نتيجة الأمر طباعة اسم الملف index.php .

دالة اسم المجلد dirname لمعرفة اسم مجلد فى مسار ، مثال :

```
Echo dirname ("myprogs/php/index.php") ;
```

تكون نتيجة الأمر طباعة اسم المسار myprogs/php .

يمكـــن تعديـــل تراخـــيص الملفات والمجلدات باستخدام الدالة chmod ، ويتم تعديل التراخيص عند نقل الملفات إلى الخادم باستخدام بروتوكول نقل الملفات FTP ، وفى هـــذه الحالة تحتاج تغيير التراخيص فى هذا البروتوكول ، ويستخدم الترخيص 755 لتغيـــير خصائص النصوص البرمجية Scripts ، بينما يستخدم الترخيص 777 لتغيير خصائص الملفات والمجلدات ، وهناك تراخيص أخرى أيضا ، ويتم تغيير التراخيص فى لغة PHP للملفات على الشكل :

```
Chmod ("name" , perm) ;
```

حيـــث تســـتبدل كلمة perm برقم الترخيص ، ويستبدل الاسم name باسم الملف أو المجلد المطلوب تغيير خصائصه ، مثال :

```
Chmod ("myfold" , 0777) ;
```

يـــتم تغيـــير ملكـــية Own الملفات والمجلدات بدالة تغيير الملكية Chown ويستطيع المشرف supervisor تغيير ملكية ملف أو مجلد إلى شخص آخر باستخدام هذه الدالة على الصورة :

```
Chown (filename , userID) ;
```

حيـــث تســـتبدل كلمـــة filename باسم ملف أو مجلد لتغيير ملكيته ، وتستبدل كلمة userID بمعـــرف المستخدم الجديد الذى يستطيع امتلاك حق الوصول إلى هذا الملف أو المجلد .

تستخدم دالة النسخ Copy لنسخ ملف ، مثال :

```
Copy ("file1.doc" , "file2.doc");
```

حيث يتم نسخ الملف file1.doc إلى الملف file2.doc .

تستخدم دالة إلغاء الربط Unlink لحذف ملف أو مجلد بالصيغة :

```
Unlink ("filename") ;
```

حيـــث يســـتبدل اســـم الملف Filename باسم الملف أو المجلد شاملا المسار الكامل

والامتداد الكامل لاسم الملف أو المجلد ، مثال :

```
Unlink ("myprog/php/file2.jpg") ;
```

يمكـــن معرفة المساحة الخالية على القرص بالبايت عن طريق استخدام دالة المساحة الحرة بالقرص diskfreespace ، مثال :

```
Echo diskfreespace ("/") ;
```

لقراءة ملف تستخدم دالة الملف File لقراءة الملف ، ويمكن وضع محتويات كل سطر من سطور الملف فى متغير بمصفوفة يمكن عرضها والتعامل معها ، مثال :

```
$array1= file ('http://www.domain1.xxx/filename.ext') ;
```

تحـــتاج الدالة فقط اسم الملف لوضعه فى مصفوفة يمكن التعامل معها ، تستخدم دالة وجود الملف File_exists لمعرفة وجود ملف للتعامل مع الملف للقراءة أو الكتابة أو النسخ أو غيرها من العمليات ، مثال :

```
If (file_exists (file1.jpg")) {
Echo "File found" ;
} else {
Echo "file not found" ;
}
```

فـــى المثال يتم فحص وجود الملف file1.jpg فإذا كان موجودا تظهر رسالة العثور علـــى الملـــف ، وإذا لم يكن موجودا تظهر رسالة الملف غير موجود ، وتعيد الدالة قيمة 1 عند وجود الملف ، وغير ذلك عند عدم العثور على الملف .

تستخدم دالة حجم الملف Filesize لمعرفة حجم الملف بالبايت ، مثال :

```
Echo filesize ("file1.jpg") ;
```

تستخدم دالة نوع الملف Filetype لمعرفة نوع الملف بالقيم : fifo, char, dir, block, link, file, unknown مثال :

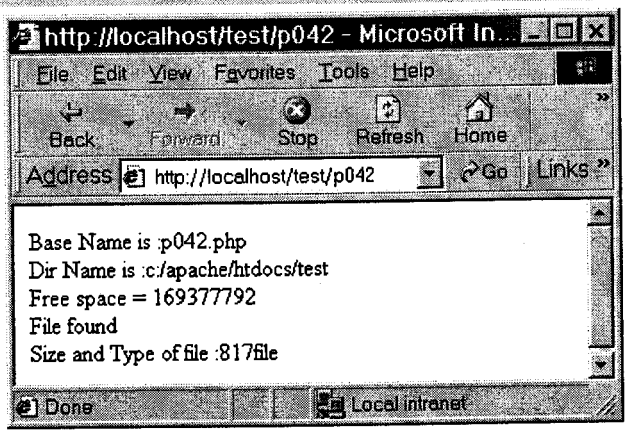
```
Echo filetype ("file1.jpg") ;
```

مثال :

```
<?
// Base and Dir Name
echo "Base Name is :";
```

```
Echo basename("c:/apache/htdocs/test/p042.php") ;
echo "<BR> Dir  Name is :";
Echo dirname("c:/apache/htdocs/test/p042.php") ;

// Change
Chmod ("c:/apache/htdocs/test/p042.php" , 0777) ;
Chown ("c:/apache/htdocs/test/p042.php" , Ahmed) ;

// copy
Copy                    ("c:/apache/htdocs/test/p042.php"
"c:/apache/htdocs/test/p043.php");

// Free space
echo "<BR> Free space = ";
Echo diskfreespace ("/") ;
$array1= file ('c:/apache/htdocs/test/p042.php') ;

// Exist

If (file_exists ("c:/apache/htdocs/test/p043.php")) {
Echo "<BR> File found" ;
} else {
Echo "<BR> file not found" ;
}

// Size and Type
echo "<BR> Size and Type of file  :";
Echo filesize ("c:/apache/htdocs/test/p043.php") ;
Echo filetype ("c:/apache/htdocs/test/p043.php") ;
?>
```

النتيجة :



ملحوظة :

تستخدم كل من الشرطة المائلة / والشرطة المائلة العكسية \ في مسار الملف في نظام تشغيل ويندوز windows لكن في نظم التشغيل الأخرى تستخدم الشرطة المائلة فقط (/) forward slash .

تستخدم دالة فتح الملف Fopen لفتح ملف ، وتحتاج الدالة فى استخدامها إلى ثلاثة متغيرات أحدهم اختيارى ، وهم : متغير اسم الملف ، ومتغير حالة الفتح ، ومتغير الاختيارات المسبقة فى اللغة (اختيارى) .

متغيرات حالة الفتح تظهر فى الجدول التالى :

المتغير	الوظيفة
r	فتح الملف للقراءة فقط ووضع المؤشر فى بداية الملف
r+	فتح ملف للقراءة والكتابة ووضع مؤشر فى بداية الملف
w	فتح ملف للكتابة فقط ووضع المؤشر ببداية الملف ، ومسح كل البيانات ، وإذا لم يتواجد الملف يتم إنشاؤه

المتغير	الوظيفة
w+	فـتح ملف للكتابة والقراءة ووضع المؤشر ببداية الملف ، ومسح كل البيانات ، وإذا لم يتواجد الملف يتم إنشاؤه
a	فتح الملف للكتابة فقط (الإضافة) ووضع المؤشر فى نهاية الملف ، وإذا لم يكن الملف موجودا يتم إنشاؤه
a+	فتح الملف للقراءة والكتابة ووضع المؤشر فى نهاية الملف ، وإذا لم يكن الملف موجودا يتم إنشاؤه

مثال :

```
$farw = fopen ("file1.txt","r") ;
$farw = fopen ("file1.txt","w") ;
```

تســتخدم دالة إغلاق الملف Fclose بعد انتهاء العمليات على فتح ملف باستخدام دالة فتح الملف ، مثال :

```
$farw = fopen ("file1.txt","r") ;
Fclose ($farw) ;
```

لقــراءة محتويات الملف تستخدم دالة قراءة ملف Fread التى يمكنها قراءة الملف أو جــزء مــنه ، وتحــتاج الدالــة متغيرين لتحديد مناول الملف ، وتحديد حجم الجزء المطلوب قراءته من الملف بالبايت ، مثال :

```
$farw = fopen ("file1.txt","r") ;
$qrs = fread ($farw, 35) ;
fclose ($farw) ;
```

لقراءة الملف كله تحتاج معرفة حجم الملف بدالة حجم الملف Filesize مثال :

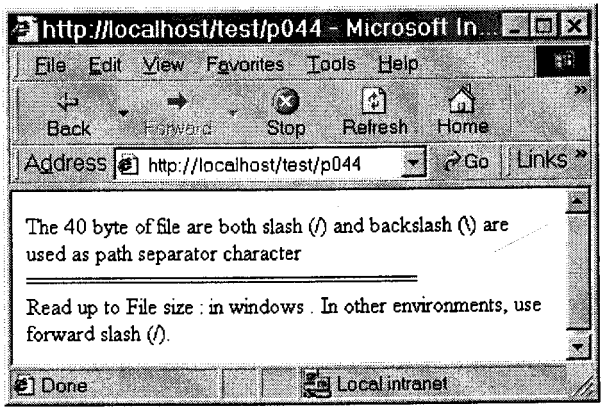
```
$farw = fopen ("file1.txt","r") ;
$qrs = fread ($farw, filesize("file1.txt")) ;
fclose ($farw) ;
```

مثال :

```
<?

$farw = fopen ("file1.txt","r") ;
```

```
$qrs = fread ($farw, 70) ;
echo " The 70 byte of file are $qrs ";
echo "<BR> ============================== " ;

$q = fread ($farw, filesize("file1.txt")) ;
echo "<BR>  Read up to File size : $q" ;

fclose ($farw) ;

?>
```

النتيجة :



ملحوظة :

عند طلب القراءة بكتابة حجم الملف فقد تمت قراءة بقية الملف لأن المؤشر يقف عند آخر جزء تمت قراءته .

تستخدم دالة معرفة الوصول لنهاية الملف feof لمعرفة الوصول إلى نهاية ملف ما مثال :

```
Feof ($farw);
```

وتعيد القيمة true عند الوصول إلى نهاية الملف .

تستخدم دالة الكتابة إلى ملف Fwrite للكتابة فى ملف ، ولهذه الدالة ثلاثة متغيرات هى : متغير (مقبض) الملف ، والنص المطلوب كتابته ، وحجم النص الذى هو أمر

اختـــيارى لا يستخدم ، وتتم الكتابة فى الملف بناء على موضع المؤشر (فى بداية أو نهاية أو منتصف الملف) ، مثال :

```
Fwrite($farw, "Text") ;
```

مثال :

```
<? php
$f2 = fopen("file1.txt", "w");
Fwrite ($f1 , "This is the added text ") ;
Fclose($f2);
?>
```

وســيتم وضـــع هــذا النص فى الملف وإذا كان الملف يحتوى على نص سيتم حذفه ووضع النص الجديد .

تســتخدم دالـــة الملـــف File لقراءة الملف ووضعه فى مصفوفة يمثل كل سطر منه عنصـــرا من عناصر هذه المصفوفة ، وتحتاج الدالة اسم الملف فقط ولا تحتاج دالة فتح أو إغلاق للملف ، ويفضل استخدام هذه الدالة فى الملفات المتوسطة الحجم بسبب استهلاك موارد الذاكرة فى جهاز الخدمة ، مثال :

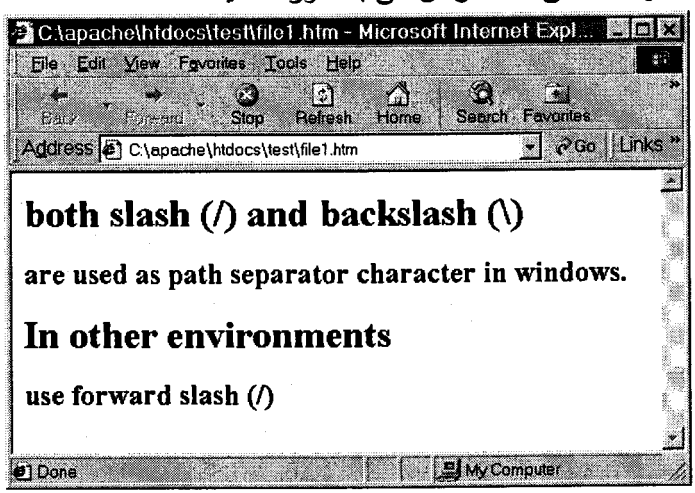
```
<? php
$fcse = file ('http://www.domain.ext') ;
while (list ($line_num,$line)= each ($fcse))
 {
echo "<b> Line $line_num:</b>  " . htmlspecialchars ($line) .
"<br>\n" ;
}
?>
```

مـــثال لفـــتح ملف وقراءة وعرض سطوره وهو الملف file1.htm الذى يحتوى على النص التالى :

```
<HTML>

<BODY>

<h1>both slash (/) and backslash (\) </h1>
```

```
<h2> are used as path separator character in windows.</h2>

<h1> In other environments</h1>

<h2>use forward slash (/)</h2>

</HTML>
</BODY>
```

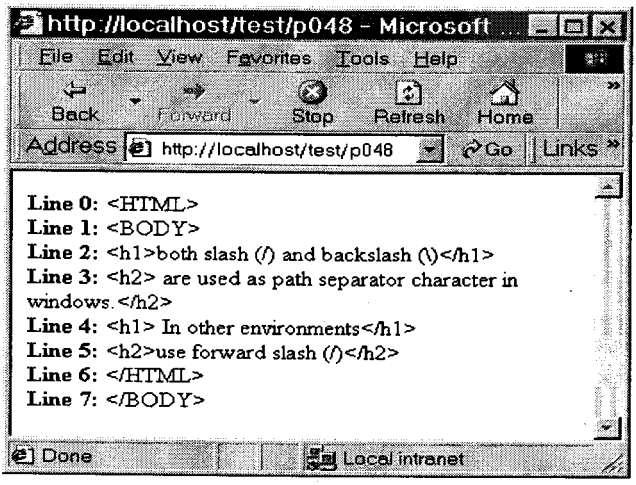
ونتيجة استعراضه على المستعرض هى بالصورة التالية :



والمثال هو :

```
<?
$f1 = file ('file1.htm') ;
while (list ($line_num,$line)= each ($f1))
 {
echo "<b> Line $line_num:</b>  " . htmlspecialchars ($line) .
"<br>\n";
}
?>
```

النتيجة :



تستخدم دالة معرفة وجود مجلد is_dir لفحص وجود مجلد ، وتعود الدالة بالقيمة true=1 عند وجود المجلد ، والقيمة False=0 عند عدم وجود المجلد ، مثال :

```
Echo is_dir ("myprog") ;
```

تستخدم دالة إنشاء مجلد mkdir لإنشاء مجلد ، وفى هذه الدالة تستخدم قيمة باسم المجلد المطلوب إنشاؤه ، وقيمة بترخيص المجلد ، ويجب استخدام دالة فحص وجود المجلد قبل إنشائه حتى لا تحدث رسالة خطأ ، مثال :

```
Echo mkdir ("myprog1",0777);
```

تستخدم دالة حذف مجلد Rmdir لحذف مجلد موجود مثال :

```
Echo rmdir ("myprog1") ;
```

تستخدم دالة تغيير الاسم لتغيير اسم مجلد أو ملف مثال :

```
Echo rename ("mypic1.jpg" , "pic01.jpg") ;
```

تستخدم دالة المسار الحقيقى Realpath لمعرفة المسار الكامل لملف أو مجلد ، مثال :

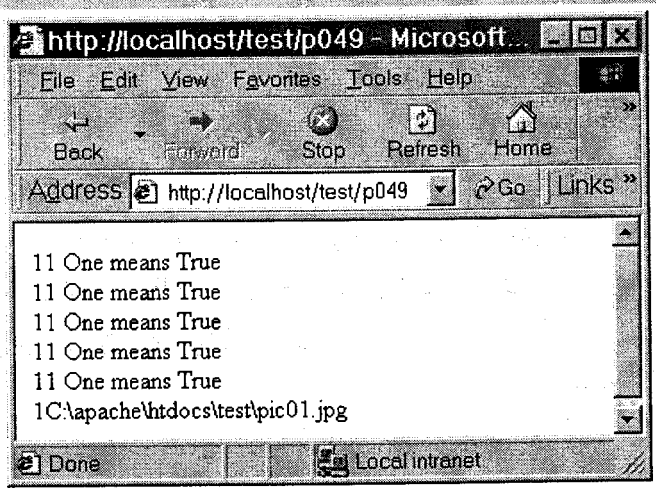
```
Echo realpath ("pic01.jpg") ;
```

مثال :

```
Echo mkdir ("myprog1",0777);
Echo "1 One means True <BR>" ;
```

```
Echo is_dir ("myprog1") ;
Echo "1 One means True <BR>" ;

Echo rmdir ("myprog1") ;
Echo "1 One means True <BR>" ;
Echo mkdir ("myprog",0777);
Echo "1 One means True <BR>" ;

Echo is_dir ("myprog") ;
Echo "1 One means True <BR>";

Copy ("pic.jpg" , "pic1.jpg") ;

Echo rename ("pic1.jpg" , "pic01.jpg") ;
Echo realpath ("pic01.jpg") ;
```

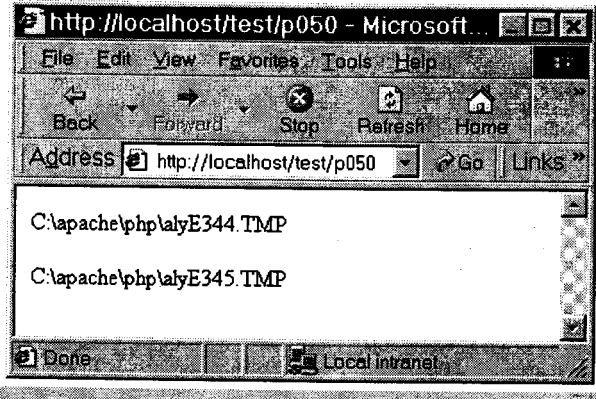
النتيجة :



تستخدم دالة الاسم المؤقت tempnam لإنشاء ملف مؤقت ، ويجب أن يكون اسم الملف غير موجود حتى لا يسبب الخطأ عند التنفيذ ، ويتم إنشاء الملف فى مجلد الملفات المؤقتة Temp ، أو فى المجلد الحالى ، مثال :

```
<?php
$tmpf1 = tempnam("","aly") ;
echo $tmpf1 ;
echo "</p>" ;
$tmpf1 = tempnam("","aly") ;
echo $tmpf1 ;
?>
```

يتم إنشاء ملفين مؤقتين بتكرار استخدام الدالة مرتين ويكون اسم كل من الملفين بادئا بالحروف aly متبوعا برقم ، مع امتداد tmp .



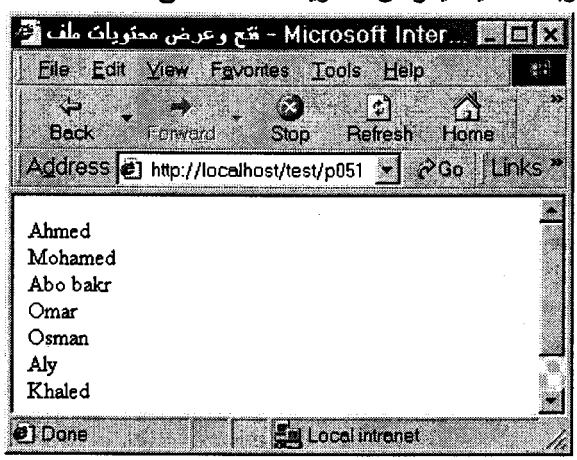
مثال :

فـتح ملـف وقراءته ليعرضه المستخدم فى برنامج المستعرض ، سنقوم بكتابة ملف اختـبار مـثل file01.dat يحـتوى على بعض الكلمات ، ثم نقوم بعمل برنامج لفتح الملف وقراءة محتوياته على الشكل التالى :

```
<HTML dir =rt1>
<HEAD>
<TITLE>  فتح وعرض محتويات ملف  </TITLE>
</HEAD>
<BODY>

<?php
$file1= fopen("file01.txt", "r");
```

```
if (!($file1))
{
print ("Can not open file");
exit :
}
while (!feof($file1))
{
$line1=fgets($file1,255);
print("$line1 <BR>\n");
}
fclose($file1);
?>

</BODY>
</HTML>
```

بعـد حفظ الملف تحت اسم بامتداد php يمكن تحميله بكتابة اسمه فى شريط عنوان المستعرض ، وبعد تشغيله يعرض محتويات الملف على الشاشة .

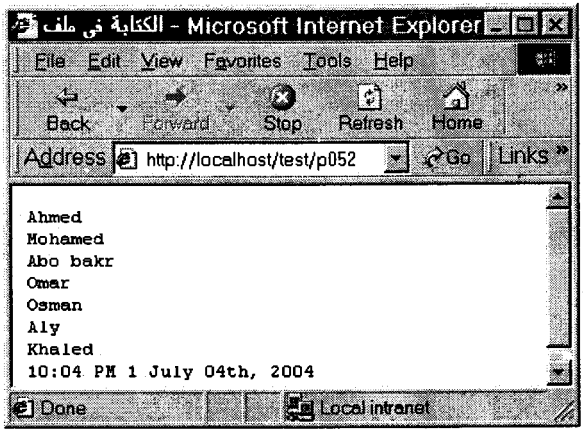


فى المثال التالى كتابة التاريخ والوقت فى ملف نص txt :

```
<HTML>
<HEAD>
<TITLE> الكتابة فى ملف </TITLE>
```

```
</HEAD>
<BODY>
<?php
$f2= fopen("file3.txt", "a");
flock($f2 ,2) ;
fputs ($f2, date ("h:i A I F dS, Y\n")) ;
flock($f2 ,3) ;
fclose ($f2) ;
print ("<PRE>") ;
readfile("file3.txt") ;
print ("</PRE>\n") ;
?>
</BODY>
</HTML>
```

نتيجة التنفيذ لمرات متكررة هو إضافة التاريخ والوقت مرات بعدد مرات التكرار :

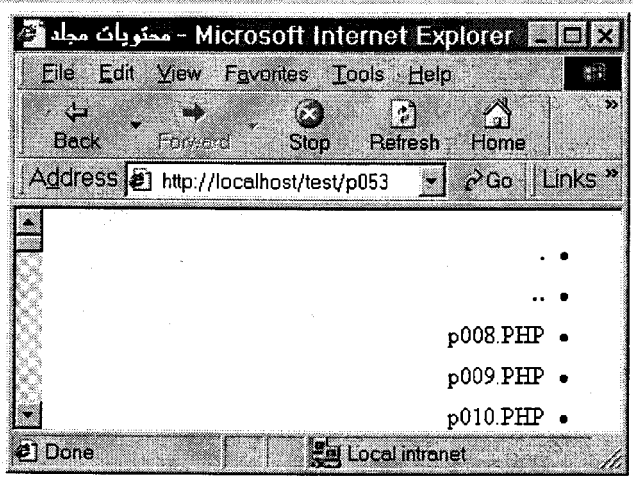


مثال استعراض محتويات مجلد :

```
<HTML dir=rtl>
<HEAD>
<TITLE> محتويات مجلد </TITLE>
</HEAD>
<BODY>
```

```
<?php
print ("<UL>\n") ;
$mydir1= dir(".") ;
while ($eName =$mydir1->read() )
{
print ("<LI> $eName \n") ;
}
$mydir1 -> close () ;
print ("</UL> \n") ;
?>

</BODY>
</HTML>
```

النتيجة :



دوال المصفوفات

تقوم دالة المصفوفة Array بإنشاء مصفوفة ، توفر المصفوفات الوقت والجهد ببرنامج أصغر حجما وأسرع تنفيذا ، ويكون شكل المصفوفة على صورة :

```
$myat = array ("a", "b", "c", "d", "e", "f", "g", "h", "i");
```

أول عناصر هذه المصفوفة هو a وآخر عناصرها هو العنصر I ، ولطباعة العنصر

الثانى نكتب :

```
Print $myart [1];
```

حيث نتعامل مع اسم المصفوفة ورقم العنصر ، فلكل مصفوفة اسم ، ولكل عنصر فى المصفوفة رقم أو اسم يدل عليه ، يمكن كتابة المصفوفة بواحدة من طريقتين كالتالى:

```
$a ["color"]  = "Yellow" ;
$a ["taste"]  = "fruit" ;
$a ["shape"]  = "round";
$a ["name"]  = "orange" ;
$a [3]  = 4 ;
```

الطريقة الثانية :

```
$a = array (
"color"  = > "Yellow" ,
"taste" => "fruit" ,
"shape"  => "round",
"name"  => "orange" ,
3  =>4
) ;
```

المصفوفة السابقة بطريقتى الكتابة هى مصفوفة ذات بعد واحد تبين عناصر مصفوفة عـن الـبرتقال باللون والطعم والشكل والاسم ، ولكتابة مصفوفة متعددة الأبعاد عن أنواع مختلفة من الفاكهة مثل البرتقال والتفاح والموز نكتب مصفوفة متعددة الأبعاد ، مثال :

```
<? Php
$a = array (
"orange" => array (
"color"  = > "Yellow" ,
"taste" => "fruit" ,
"shape"  => "round") ,

"apple" => array (
"color"  = > "red" ,
"taste" => "sweet" ,
"shape"  => "round")
```

```
);
?>
```

لمعرفة لون البرتقال نكتب اسم المصفوفة واسم العنصر مثل :

```
Echo $a ["orange"]["color"] ;
```

تستخدم دالة عد قيم عناصر مصفوفة array_count_values لعرض تكرار قيم معينة فى عناصر المصفوفة ، مثال :

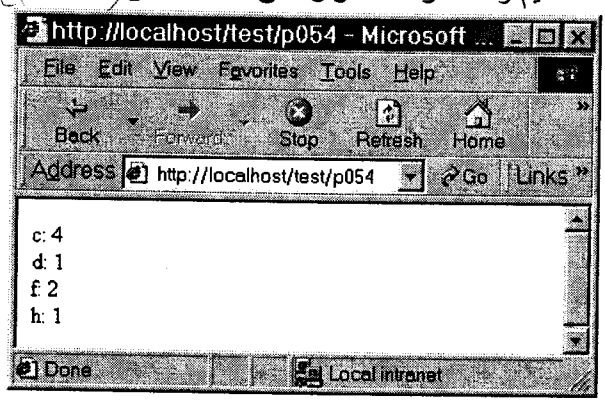
إذا كانت المصفوفة بالصورة التالية :

```
$a1 = array ("c", "d" , "c", "f" , "c" , "f" , "c" , "h") ;
```

فى المصفوفة يتكرر حرف c أربع مرات ، ويتكرر حرف f مرتين ، ولكل من حرف h , d مرة واحدة ، لمعرفة تكرار قيمة ما يمكن كتابة :

```
<?php
$a1 = array ("c", "d" , "c", "f" , "c" , "f" , "c" , "h") ;
$c1 = array_count_values ($a1);
foreach ($c1 as $key=>$value)
{
print ("$key: $value <BR>\n");
}
?>
```

سوف يتم طباعة القيم وعدد مرات التكرار على الشكل التالى :

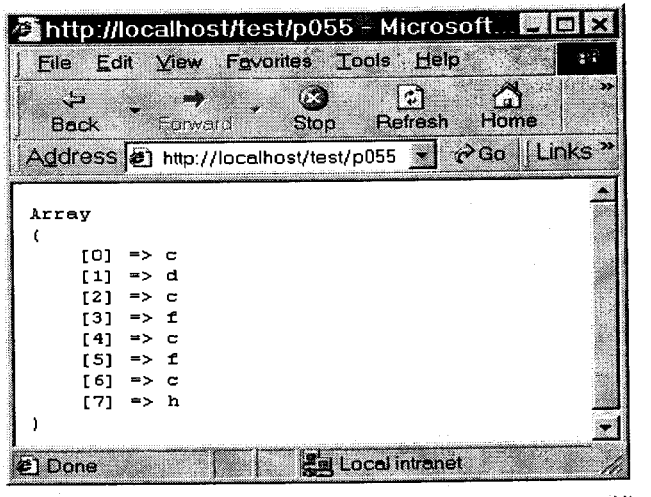


تستخدم دالة طباعة المصفوفة print_r لطباعة قيم عناصر المصفوفة بترتيب أرقامها

فى المصفوفة ، مثال :

```
<?
$a1 = array ("c", "d" , "c", "f" , "c" , "f" , "c" , "h") ;

print ("<PRE>");
print_r($a1);
print ("</PRE>");
?>
```

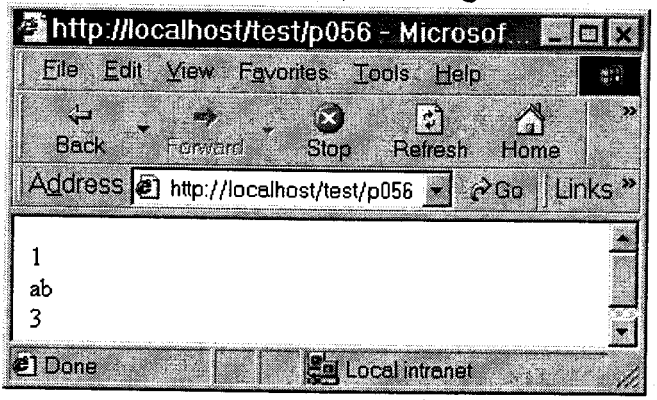
عند تنفيذ المثال تكون النتيجة على الشكل التالى :



تستخدم دالة مفاتيح المصفوفة array_keys لعرض مفاتيح عناصر المصفوفة التى تكون على شكل أسماء أو أرقام لعناصر هذه المصفوفة ، مثال :

```
<?php
$a2=array (1 => "x" , "ab" =>"home" , 3 =>"computer");

foreach (array_keys($a2) as $key)
{
print ("$key<BR>\n");
}
?>
```

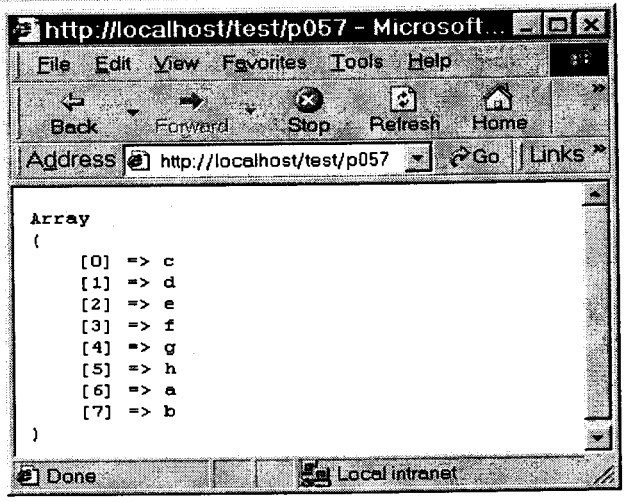
النتيجة ستكون معرفة المفاتيح الثلاثة للقيم فى المصفوفة وهى المفاتيح :



تستخدم دالة جمع أو دمج مصفوفة array_merge لدمج عناصر عدة مصفوفات فى مصفوفة واحدة ، مثال :

```
<?

$a3 = array ("c", "d" , "e", "f") ;
$a4 = array ("g" , "h" , "a" , "b") ;

$a5 = array_merge ($a3, $a4) ;

print ("<PRE>");
print_r($a5);
print ("</PRE>");
?>
```

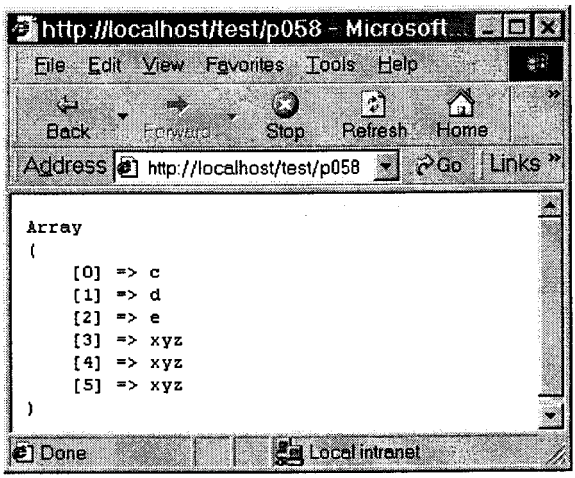
النتيجة هى :



تستخدم دالة استكمال المصفوفة array_pad لجعل عدد عناصر المصفوفة مساويا لعدد معين ، ووضع قيم افتراضية فى العناصر الزيادة ، ويمكن زيادة عدد عناصر المصفوفة من أعلى بوضع العناصر الجديدة فى بداية المصفوفة ، أو زيادة عدد عناصر المصفوفة من أسفل بوضع عناصر جديدة فى نهاية المصفوفة ، مثال :

```
<?
$a6 = array ("c", "d" , "e") ;
$a7 = array_pad ($a6, 6,  xyz) ;

print ("<PRE>");
print_r($a7);
print ("</PRE>");
?>
```

النتيجة هى :



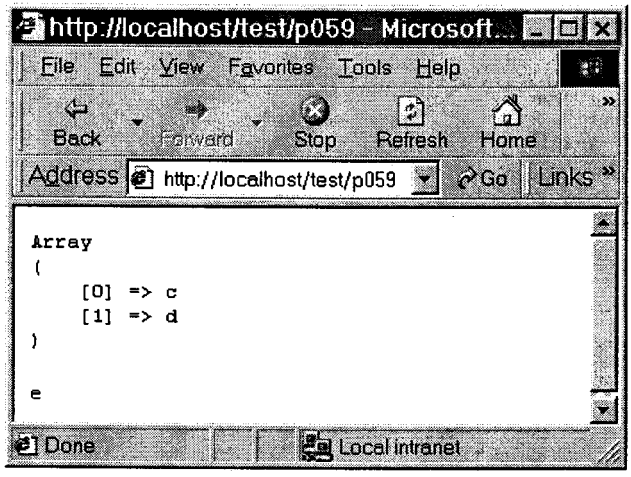
ملحوظة :

لإضافة القيم أعلى المصفوفة يوضع الرقم ٦ سالبا ، فيتم تعديل عند عناصر المصفوفة إلى ستة عناصر أولها ثلاثة عناصر بالقيم الافتراضية المضافة xyz ، ثم تليها العناصر الأخرى .

تستخدم دالة قص آخر عنصر فى المصفوفة array_pop لحذف آخر عنصر من عناصر مصفوفة ، مثال :

```
<?
$a8 = array ("c", "d" , "e") ;
$a9= array_pop($a8) ;

print ("<PRE>");
print_r($a8);
print ("</PRE>");
print($a9);
?>
```

النتيجة هى :



تستخدم إضافة عنصر فى المصفوفة array_push لإضافة عنصر أو أكثر إلى عناصر مصفوفة فى نهاية المصفوفة ، مثال :

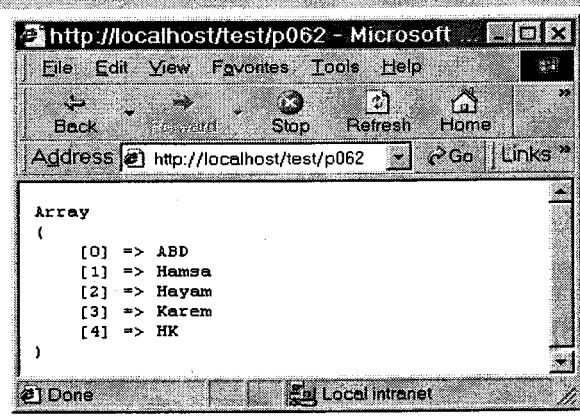
```
<?
$a10 = array ("HK", "KH" , "Aly", "Ahmed") ;
$a11= array_push ($a10, "Kareem" , "Hayam");

print ("<PRE>");
print_r($a11);
print ("</PRE>");
?>
```

النتيجة هى إعادة الرقم الجديد لعدد عناصر المصفوفة ففى المصفوفة الأولى أربعة عناصر ، وفى المصفوفة الثانية عنصران ، وبالتالى تعيد الرقم ٦ كناتج للعملية .

تستخدم دالة عكس المصفوفة array_reverse لعكس ترتيب عناصر المصفوفة ليصبح العنصر الأخير فى المصفوفة هو العنصر الأول ، ويصبح العنصر الأول فى المصفوفة هو العنصر الأخير ، وهكذا بالنسبة للعناصر التى بينهما بعد عكس المصفوفة ، مثال :

```
<?
$a12 = array ("HK", "Karem", "Hayam", "Hamsa", "ABD" );
$a14 =array_reverse($a12) ;
print ("<PRE>");
print_r($a14);
print ("</PRE>");
?>
```

النتيجة هى :

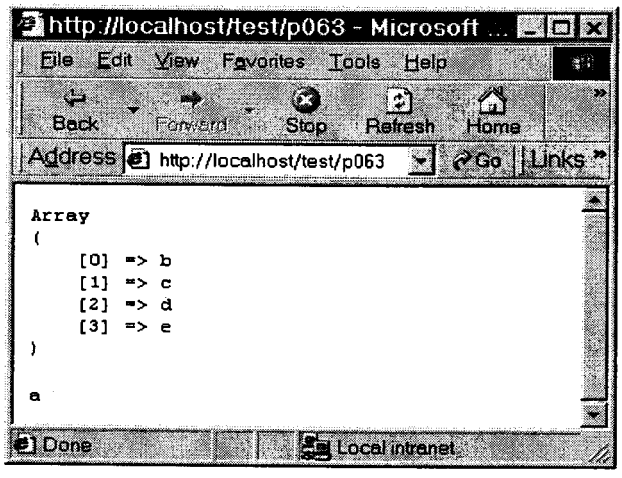


تستخدم دالة إزاحة مصفوفة array_shift لحذف أول عنصر من عناصر مصفوفة .

مثال :

```
<?
$a15 = array ("a" , "b" , "c", "d" , "e" );
$a16 =array_shift($a15) ;
print ("<PRE>");
print_r($a15);
print ("</PRE>");
print ("<PRE>");
print_r($a16);
print ("</PRE>");
?>
```

النتيجة هى :



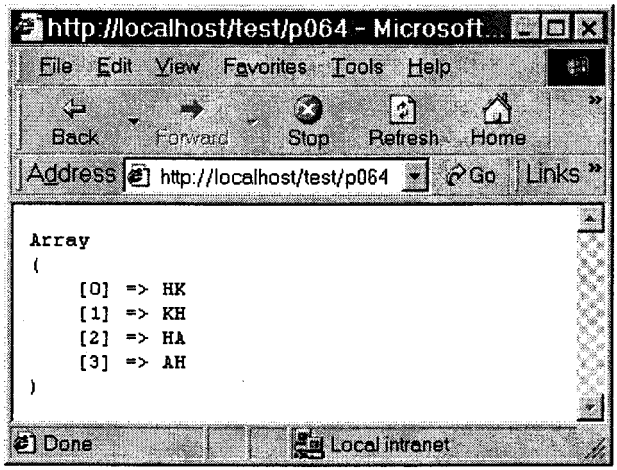
تستخدم دالة شريحة مصفوفة array_slice لنسخ جزء من مصفوفة إلى مصفوفة جديدة ، بحيث يتم تحديد بداية النسخ وعدد القيم المنسوخة إلى المصفوفة الجديدة ،

مثال :

```
<?
$a17 = array ("abd" , "Kareem" , "Ahmed", "HK" , "KH" ,"HA" , "AH");
$a18 =array_slice($a17, 3 ,4 ) ;
```

```
print ("<PRE>");
print_r($a18);
print ("</PRE>");
?>
```

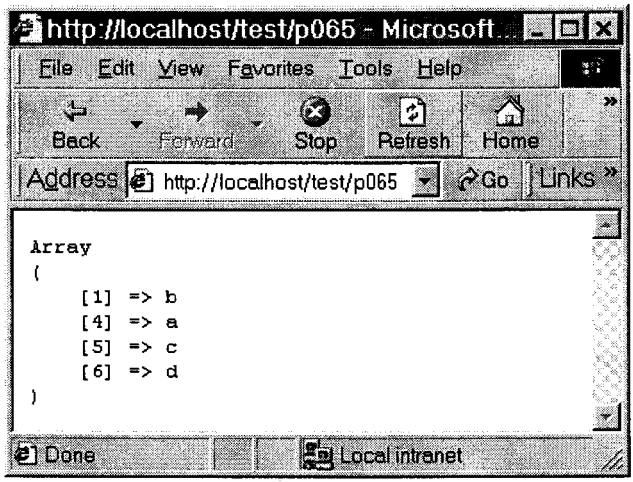
النتيجة هى :



تستخدم دالة القيم الوحيدة array_unique لجلب القيم المنفردة الغير مكررة فى مصفوفة ، مثال :

```
<?
$a19 = array ("a" , "b" , "c", "d" , "a" ,"c" , "d");
$a20 =array_unique($a19) ;

print ("<PRE>");
print_r($a20);
print ("</PRE>");
?>
```

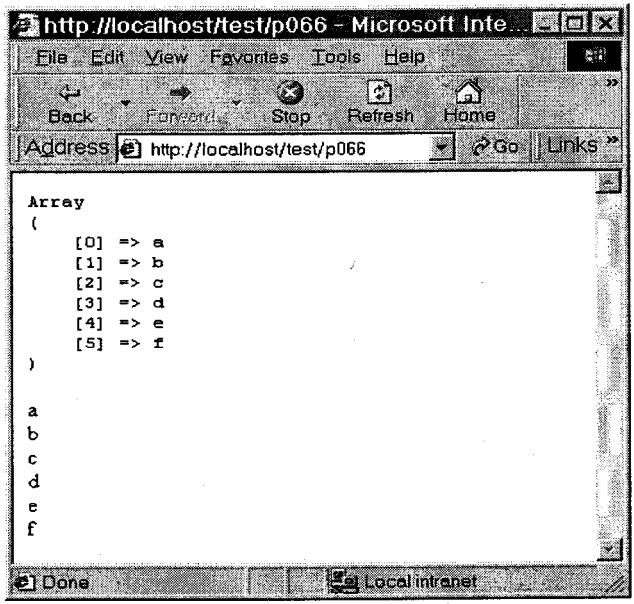
النتيجة هى :



تستخدم دالة قيم عناصر المصفوفة array_values لوضع قيم عناصر مصفوفة فى مصفوفة جديدة ، مثال :

```
<?
$aa1 = array ("a" , "b" , "c", "d" , "e" ,"f");
$aa2 =array_values ($aa1) ;
print ("<PRE>");
print_r($aa2);
print ("</PRE>");
for ($m=0; $m < count($aa2); $m++) {
print ("$aa2[$m] <BR>\n") ;
}
?>
```

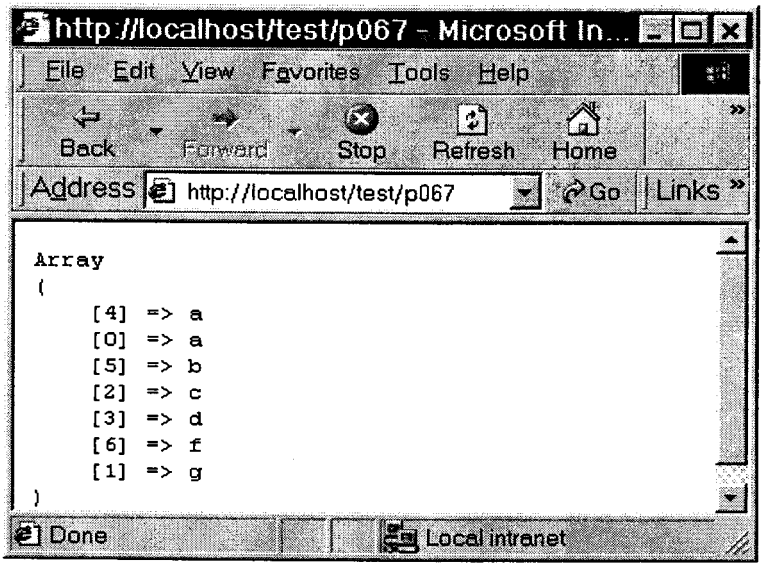
النتيجة هى :



تستخدم دالة ترتيب قيم عناصر المصفوفة تصاعديا asort لترتيب قيم عناصر مصفوفة من الأصغر إلى الأكبر ، مثال :

```
<?
$b1 = array ("a" , "g" , "c", "d" , "a" ,"b" , "f");
asort ($b1) ;
print ("<PRE>");
print_r($b1);
print ("</PRE>");
?>
```

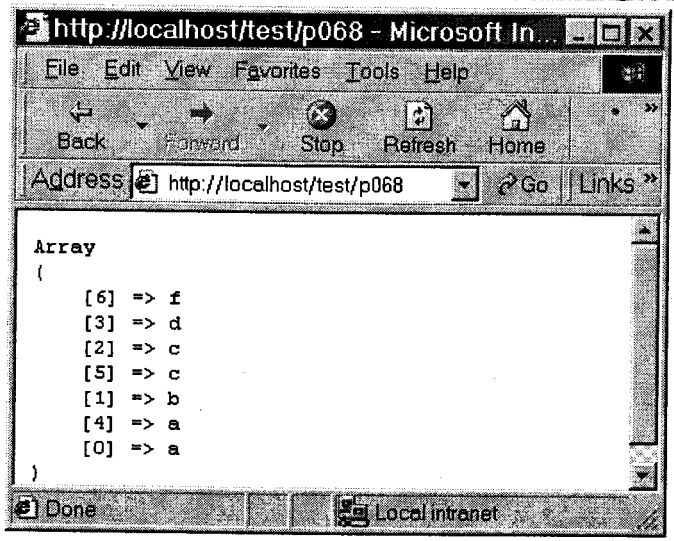
النتيجة هى :



تستخدم دالة ترتيب قيم عناصر المصفوفة تنازليا arsort لترتيب قيم عناصر مصفوفة من الأكبر إلى الأصغر ، مثال :

```
<?
$b1 = array ("a" , "b" , "c", "d" , "a" ,"c" , "f");
arsort ($b1) ;

print ("<PRE>");
print_r($b1);
print ("</PRE>");
?>
```

النتيجة هى :



تستخدم دالة عد count عناصر المصفوفة لإرجاع عدد عناصر المصفوفة ، مثال :

```
<?
$b1 = array ("a" , "b" , "c", "d" , "a" ,"c" , "f");
print count ($b1) ;
?>
```

النتيجة هى : 7 .

دوال حركة مؤشر قراءة مصفوفة

عند قراءة مصفوفة يقوم البرنامج بوضع مؤشر وهمى عند المتغير الذى يقرأه ، وعن طـريق المؤشـر يمكن التحرك بين عناصر المصفوفة للأمام أو الخلف ، وفيما يلى دوال حركة مؤشر المصفوفة :

الدالة	الوظيفة
Reset	وضع المؤشر فى بداية المصفوفة
Next	الانتقال للعنصر التالى فى المصفوفة

الدالة	الوظيفة
Prev	الانتقال للعنصر السابق فى المصفوفة
Current	قيمة المتغير الحالى
End	وضع المؤشر فى نهاية المصفوفة

مثال :

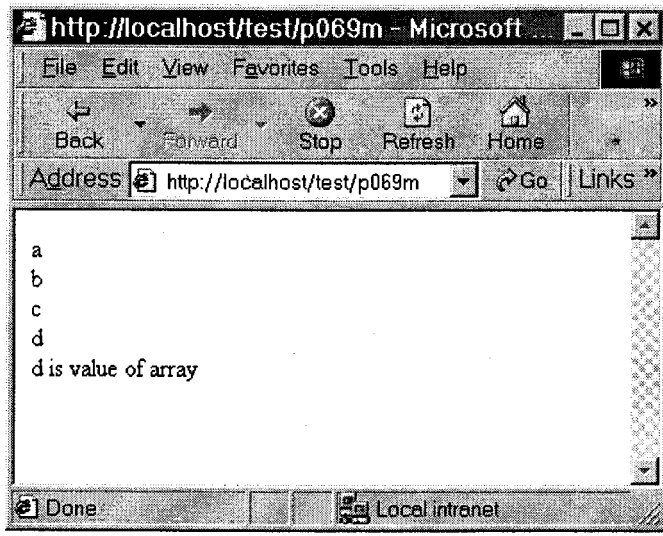
```
<?
$t1 = array ("a" , "b" , "c" , "d") ;
for (reset ($t1) ; $value = current ($t1) ; next ($t1))
{
print ("$value <BR>\n") ;
}
?>
```

النتيجة : طباعة قيم المصفوفة .

تستخدم دالة البحث عن قيم داخل المصفوفة in_array للعودة بقيمة التحقق True عند وجود قيمة بين عناصر الدالة ، مثال :

```
<?
$t1 = array ("a" , "b" , "c" , "d") ;
if (in_array ("d",  $t1))
print ("d is value of array ") ;
?>
```

نتيجة المثالين السابقين معا :



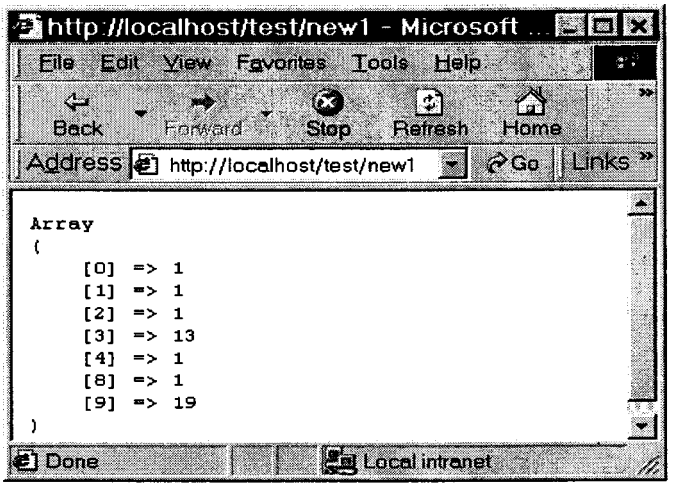
أمثلة متنوعة :

يمكـــن إنشاء مصفوفة باستخدام دالة مصفوفة array() أو دالة لائحة list() أو وضع القيم مباشرة فى مصفوفة .

مثال توليد رقم فهرس تلقائى :

```
//Automatic index with Array()
$array = array( 1, 1, 1, 1,  1, 8=>1,  4=>1, 19, 3=>13);
print "<PRE>";
print_r($array);
print "</PRE>" ;
```

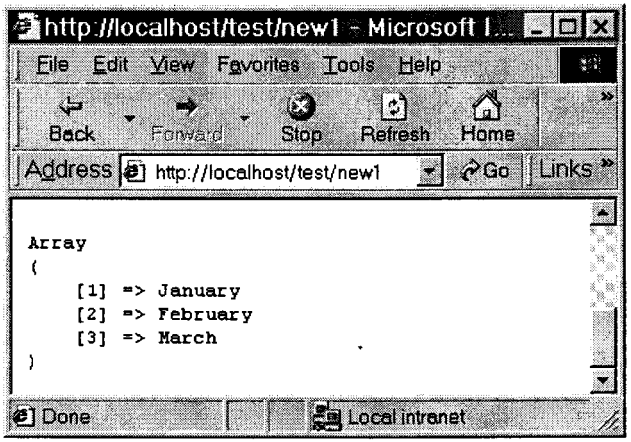
النتيجة :



مثال توليد رقم دليل فهرس أساسى 1-based array فى المصفوفة :

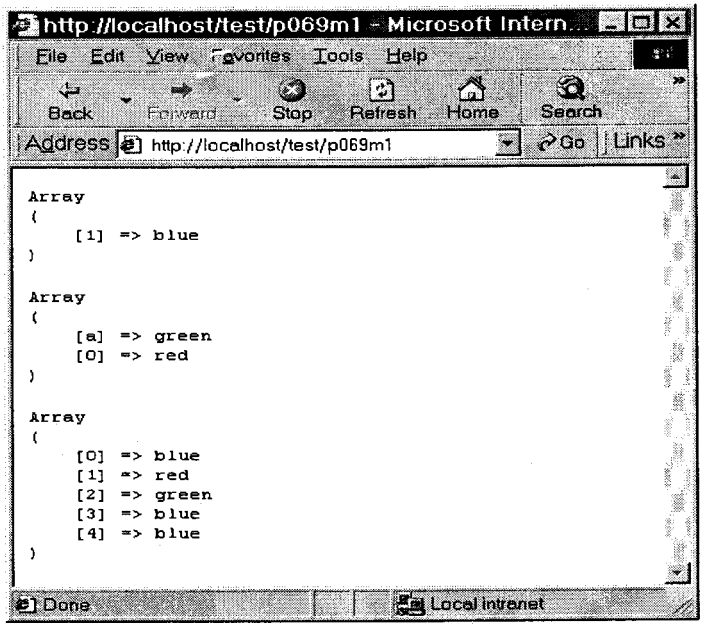
```
// based index with Array()
$firstquarter  = array(1 => 'January', 'February', 'March');
print "<PRE>";
print_r($firstquarter);
print "</PRE>" ;
```

نتيجة المثال ستكون :



أمثلة أخرى لدوال المصفوفات :

```
// Example Array_diff()
$array1 = array ("a" => "green", "red", "blue");
$array2 = array ("b" => "green", "yellow", "red");
$result = array_diff ($array1, $array2);
print ("<PRE>");
print_r($result);
print ("</PRE>");
// Example Array_intersect()
$array1 = array ("a" => "green", "red", "blue");
$array2 = array ("b" => "green", "yellow", "red");
$result = array_intersect ($array1, $array2);
print ("<PRE>");
print_r($result);
print ("</PRE>");
// Example Array_keys()
$array = array (0 => 100, "color" => "red");
array_keys ($array);       // returns array (0, "color")
$array = array ("blue", "red", "green", "blue", "blue");
array_keys ($array, "blue");  //  returns array (0, 3, 4)
print ("<PRE>");
print_r($array);
print ("</PRE>");
```



الدوال الرياضية

تتـــيح دوال الأرقـــام والدوال الرياضية التعامل مع الأرقام وإجراء العمليات المختلفة عليها .

الـــدوال الرياضية بصفة عامة كثيرة ومتعددة منها الدوال التى تنفذ المهام الرياضية ومـــنها الدوال التى تقوم بعمليات رياضية وفيما يلى الدوال المستخدمة فى العمليات الرياضية :

bcadd	Add two numbers	جمع رقمين
bccomp	Compare two numbers	مقارنة رقمين
bcdiv	Divide two numbers	قسمة رقمين
bcmod	Modulus of number	القيمة المطلقة

bcmul	Multiply two numbers	ضرب رقمين
bcpow	Raise number to another	رفع رقم لأس رقم
bcscale	Set default scale	وضع مقياس
bcsqrt	Square root of number	جذر تربيعى
bcsub	Subtract number from another	طرح رقم من آخر

فيما يلى دوال رياضية :

abs	Absolute value	القيمة المطلقة
acos	Arc cosine	
asin	Arc sine	
atan	Arc tangent	
atan2	arc tangent of two variables	
base_convert	Convert number between arbitrary bases	تحويل أرقام بين نظامين
bindec	Binary to decimal	ثنائى إلى عشرى
ceil	Round fractions up	تقريب
cos	Cosine	جيب تمام
decbin	Decimal to binary	عشرى إلى ثنائى
dechex	Decimal to hexadecimal	عشرى إلى سداسى عشرى
decoct	Decimal to octal	عشرى إلى ثمانى
deg2rad	Converts degrees to radian equivalent	درجات إلى دائرية
exp	e to the power of	رفع أسى

floor	Round fractions down	تقريب كسر للأدنى
getrandmax	Show largest possible random value	أكبر قيمة عشوائية
hexdec	Hexadecimal to decimal	سداسي عشرى إلى عشرى
log	Natural logarithm	لوغريتم طبيعى
log10	Base 10 logarithm	لوغاريتم الأساس ١٠
max	Find highest value	أكبر قيمة
min	Find lowest value	أقل قيمة
number_format	Format number with grouped thousands	صيغة العدد بتجميع الألوف
octdec	Octal to decimal	ثمانى إلى عشرى
pi	Get value of pi	قيمة ط
pow	Exponential expression	تعبير أسى
rad2deg	Converts radian number to the equivalent number in degrees	تحويل دائرى إلى درجات
rand	Generate a random value	قيم عشوائية
round	Rounds a float	تقريب عدد مضاعف
sin	Sine	جيب
sqrt	Square root	جذر تربيعى
tan	Tangent	ظل زاوية

تستخدم دالة الجمع bcadd لجمع رقمين ، وتتيح تحديد عدد الأرقام بعد العلامة العشرية ، ويتم إرسال ثلاثة متغيرات لهذه الدالة هى : الرقم الأول ، والرقم الثانى المطلوب جمعهما ، ومتغير عدد الأرقام بعد العلامة العشرية ، مثال :

```
<?
Print (bcadd("21.1", "1.254", 0)) ;
Print "<BR>" ;
Print (bcadd("2.4", "5.551", 2)) ;
Print "<BR>" ;
Print (bcadd("3.21", "7.72", 4)) ;
?>
```

النتيجة :

```
22
7.95
10.9300
```

تستخدم دالة المقارنة bccomp للمقارنة بين رقمين ، إذا تساوى الرقمان تعود بنتيجة صفر ، وإذا كان رقم اليسار أكبر من رقم اليمين تعود بقيمة واحد ، وإذا كان رقم اليسار أقل من رقم اليمين تعود بقيمة سالب واحد ، وتتجاهل الدالة الأرقام بعد الفاصلة العشرية ما لم يتم إرسال متغير ثالث للدالة (مع الرقم الأول ، والرقم الثانى) لتحديد عدد الأرقام التى تتضمنها عملية المقارنة ، مثال :

```
<?
Print (bccomp ("21.1", "21.254")) ;
Print "<BR>" ;
Print (bccomp("12.4", "5.551")) ;
Print "<BR>" ;
Print (bccomp("3.21", "7.72")) ;
?>
```

النتيجة :

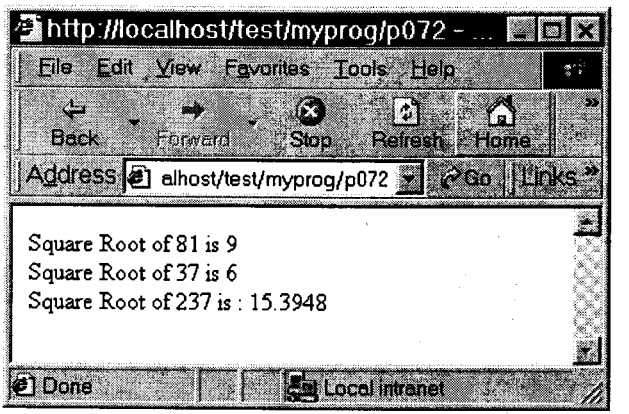
```
0
1
-1
```

تستخدم دالة الجذر التربيعى bcsqrt للحصول على الجذر التربيعى لرقم ، ويمكن

تحديد لأرقام بعد الفاصلة العشرية ، مثال :

```
<?
Print "Square Root of  81 is " ;
Print (bcsqrt ("81")) ;
Print "<BR> Square Root of  37 is " ;
Print (bcsqrt ("37")) ;
Print "<BR>" ;
$a= 237;
Print "Square Root of $a  is : " ;
Print (bcsqrt ($a, 4)) ;
?>
```

النتيجة :



تستخدم دالة القيمة الحقيقية abs لإعادة القيمة الحقيقية لرقم ، مثال :

```
Print (abs(-24));
```

تعود بالرقم 24 .

تستخدم دالة القيمة العظمى max للحصول على أكبر قيمة من بين مجموعة قيم رقمية أو نصية ، مثال :

```
<?
Print (max( 4, 7, 22));
Print ("<BR>") ;
Print (max(7, 22, "ahmed"));
```

```
?>
```

النتيجة :

```
22
22
```

تستخدم دالة القيمة الصغرى min للحصول على أقل قيمة من بين مجموعة قيم رقمية أو نصية .

تستخدم دالة التقريب ceil لتقريب قيمة رقمية إلى أقرب رقم صحيح تال للقيمة مثال :

```
Print (ceil(11.34)) ;
```

تعود بالقيمة 12 .

تستخدم دالة لوغاريتم log للحصول على لوغريتم عدد ، مثال :

```
Print (log(42.24));
```

تعود بالقيمة 3.743367639398 .

تستخدم دالة الجذر التربيعي sqrt للحصول علي جذر عدد ، مثال :

```
Print (sqrt(37));
```

تعود بالقيمة 6.0827625302982 .

دوال المتغيرات

تستخدم هذه الدوال للتعامل مع المتغيرات :

وتتنوع هذه الدوال إلى حد كبير ومنها :

doubleval	double value of variable	قيمة مضاعفة لمتغير
empty	Is variable was set	هل المتغير خالى
gettype	Get type of variable	إيجاد نوع المتغير
intval	Integer value of a variable	قيمة صحيحة لمتغير
is_array	Is variable an array	هل المتغير مصفوفة
is_bool	Is variable boolean	هل المتغير بولى

is_double	Is variable double	هل المتغير مضاعف
is_float	Is variable float	هل المتغير بعلامة عشرية
is_int is_integer is_long	Is variable integer	هل المتغير عدد صحيح
is_numeric	Is variable number or a numeric string	هل المتغير رقم أم نص
is_object	Is variable object	هل المتغير كائن
is_real	Is variable real	هل المتغير حقيقي
is_resource	Is variable resource	هل المتغير مورد
is_string	Is variable string	هل المتغير نص
isset	Is variable is set	هل المتغير معد
print_r	Prints variable readable information	طباعة قيم
serialize	Generates a storable representation of a value	توليد تمثيل لقيمة
settype	Set type of variable	إعداد نوع متغير
strval	Get string value of a variable	قيمة نص المتغير
unset	Unset a given variable	إلغاء إعداد متغير
var_dump	Dumps information about a variable	إلغاء معلومات متغير

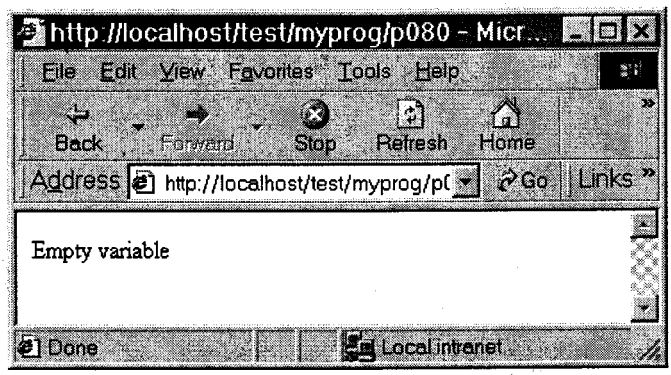
تستخدم الدالة doubleval لتحويل متغير إلى نوع رقمي مضاعف الدقة ، مثال :

```
<?
$v1= '733.641 Computer speed' ;
$dv1= doubleval ($v1) ;
Print $dv1 ;
?>
```

النتيجة : 733.641

دالـــة معرفة احتواء المتغير على قيمة Empty ، يمكن عن طريقها معرفة إذا كان قد تم إنشاء متغير وهل هو فارغ أو له قيمة ، فإذا كان المتغير موجودا وليس فارغا ولا يساوى الصفر تعيد الدالة قيمة False ، مثال :

```
<?
$v2= 0 ;
if (empty ($v2)) {
Print "Empty variable" ;
}
else {
echo "Variable value = $v2" ;
}
?>
```



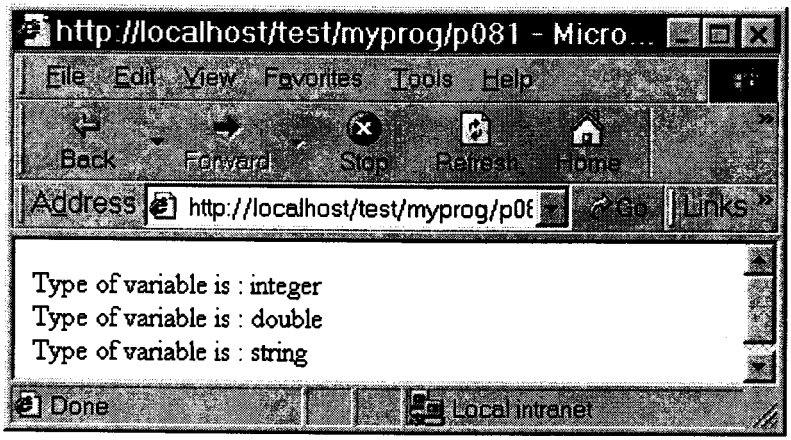
دالة معرفة نوع المتغير gettype تتمكن من معرفة نوع المتغير ، مثال :

ســـوف نســـتخدم الدالـــة sprintf للحصول على صيغة تنسيق النص وهى من دوال النصوص الحرفية Strings :

```
<?
$a= sprintf ("%s <BR>\n" , gettype (24)) ;
Echo "Type of variable is : $a ";

$b=sprintf ("%s <BR>\n" , gettype (4.2)) ;
Echo "Type of variable is : $b ";
```

```
$c = sprintf ("%s <BR>\n" , gettype (Welcome)) ;
Echo "Type of variable is : $c ";
?>
```

النتيجة :

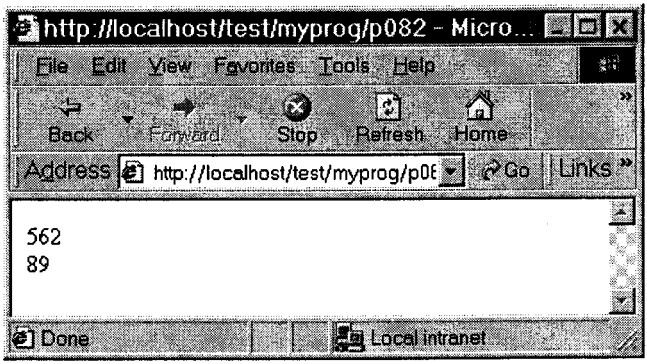


تستخدم دالة تحويل متغير إلى عدد صحيح Intval لتحويل متغير إلى عدد صحيح ،

مثال :

```
<?

$v3= "89.5 KGM" ;

$v4 = intval ("562.53mt") ;
$v5= intval ($v3) ;

Print $v4 ;
Print "<BR>" ;
Print $v5 ;

?>
```

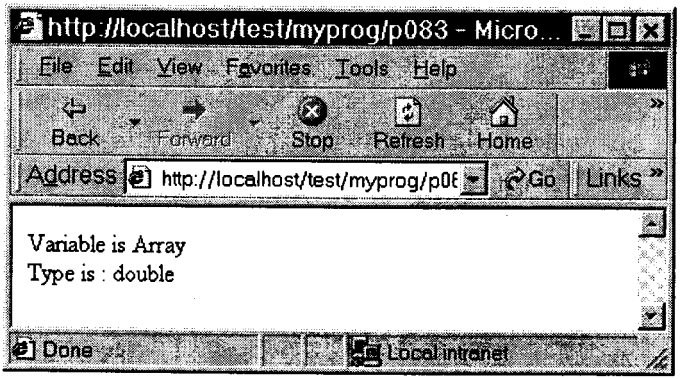
النتيجة :



تستخدم الدالة is_array لمعرفة ما إذا كان المتغير مصفوفة ، وتعود بقيمة True عندما يكون المتغير مصفوفة ، مثال :

```
<?
$fruits = array ("Apple" , " Orange" , "Mango") ;
if (is_array($fruits)) {
Print (" Variable is Array") ;
}

$f = 12.56 ;
if (is_array($f)) {
Print ("Variable is Array") ;
}
else {
$b=gettype ($f) ;
echo "<BR> Type is : $b " ;
}
?>
```

النتيجة :



تستخدم الدالة is_bool لمعرفة ما إذا كان المتغير من نوع الجبر البولى Boolean (حقيقى True أو غير حقيقى False) ، وتعود بقيمة True عندما يكون المتغير من هذا النوع ، مثال :

```
<?
$v5 = true ;
if (is_bool($v5)) {
Print (" Variable is Boolean ") ;
}
<?
```

تستخدم الدالة is_double لمعرفة ما إذا كان المتغير من نوع دقة مضاعفة Double .

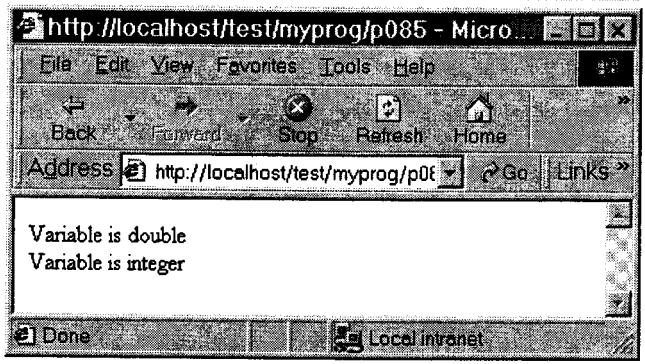
تستخدم الدالة is_int لمعرفة ما إذا كان المتغير من نوع العدد الصحيح Integer .

مثال :

```
<?
$v6 = 41.567 ;
$v7 = 74 ;
if (is_double ($v6)) {
Print (" Variable is double <BR>") ;
}
if (is_int ($v7)) {
Print (" Variable is integer ") ;
}
```

```
?>
```

النتيجة :



دوال الإسناد هى مجموعة من الدوال تستخدم لمعرفة تعريف متغير وإسناد نوع من البيانات إلى متغير ، ومسح المتغير من الذاكرة ، وتحديد نوع المتغير والبيانات الموجودة فيه ، ومنها :

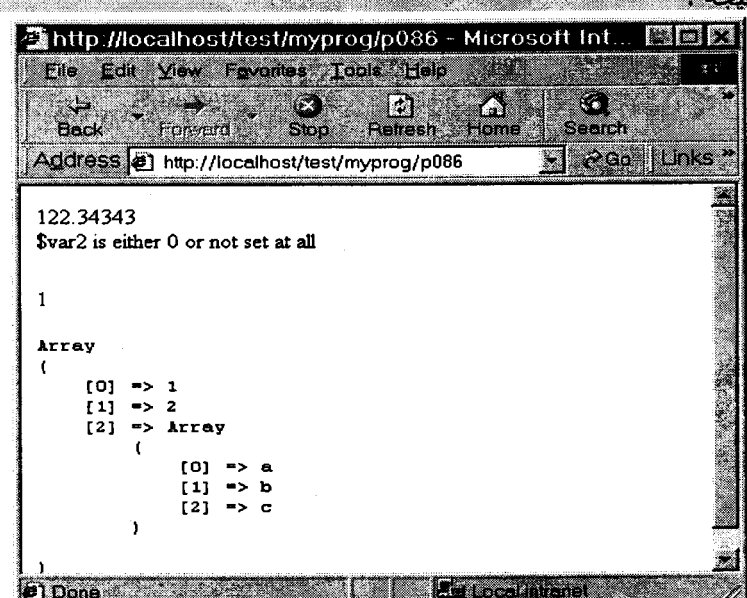
دالة معرفة تعريف متغير isset ، وتبين ما إذا كان قد تم تعريف متغير أم لا ، وتقوم دالة settype بإسناد نوع من البيانات إلى متغير ، وتقوم الدالة unset بمسح متغير من الذاكرة ، وتقوم الدالة var_dump بتحديد نوع المتغير والبيانات الموجودة فيه .

أمثلة :

```
<?php
// doubleval -- Get double value of a variable
$var1 = '122.34343The';
$dvov = doubleval ($var1);
print $dvov; // prints 122.34343
print "<BR>";

// empty -- Determine whether a variable is set
$var2 = 0;
if (empty($var)) {  // evaluates true
   echo '$var2 is either 0 or not set at all';
}
print "<BR>";
```

```
if (!isset($var2)) { // evaluates false
    echo '$var2 is not set at all';
}
print "<BR>";

// isset -- Determine whether a variable is set
$a = "test";
echo isset ($a); // true
unset ($a);
print "<BR>";
echo isset ($a); // false

// print_r --  Prints readable information about a variable
$a = array (1, 2, array ("a", "b", "c"));
Print "<PRE>" ;
print_r ($a);
Print "</PRE>" ;
?>
```

النتيجة :



أمثلة أخرى :

```
function foo() {
   static $a;
   $a++;
     echo "$a\n";
   unset($a);
}
foo();
foo();
foo();
```

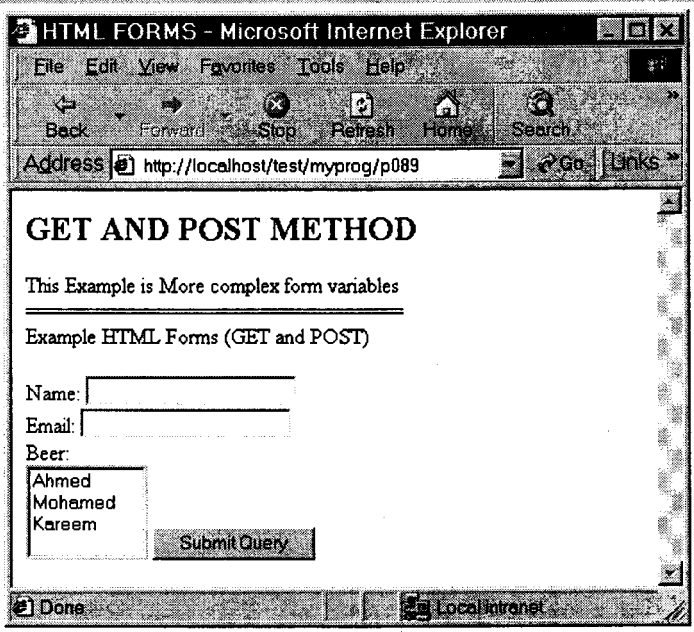
نتيجة المثال السابق ستكون : طباعة 123 على الشاشة .

مثال للمتغيرات الخارجية :

```
<HTML>
<TITLE> HTML FORMS </TITLE>
<BODY>
<H2> GET AND POST METHOD </H2>

<?php
// Variables from outside PHP
// HTML Forms (GET and POST)
// Example  More complex form variables
Echo "This Example is  More complex form variables";
Echo "<BR> ==============================";
Echo "<BR> Example HTML Forms (GET and POST) ";
?>
<form action="array.php" method="post">
   Name: <input type="text" name="personal[name]"><br>
   Email: <input type="text" name="personal[email]"><br>
   Beer: <br>
   <select multiple name="beer[]">
      <option value="ahmed">Ahmed
      <option value="mohamed">Mohamed
      <option value="kareem">Kareem
      </select>
   <input type="submit">
```

```
</form>
</BODY>
</HTML>
```

النتيجة :



دوال الصور Image functions

تعتمد معالجة الصور على مكتبة الرسوم GD المثبتة فى النظام وإنشاء ومعالجة الرسوم يجب تثبيت مكتبة GD وهى مكتبة متاحة فى الموقع http://www.boutell.com/gd/ ، ومن دوال معالجة الرسوم :

GetImageSize	Get size of GIF, JPG, PNG or SWF image	مقاسات رسم
ImageArc	Draw partial ellipse	رسم جزء بيضاوى
ImageChar	Draw character horizontally	رسم حرف أفقيا

ImageCharUp	Draw character vertically	رسم حرف رأسيا
ImageColorAllocate	Allocate color for image	تخصيص لون لرسم
ImageColorDeAllocate	Deallocate color for image	إلغاء تخصيص لون
ImageColorAt	Get index of color of pixel	فهرس لون بكسل
ImageColorClosest	Get index of closest color to specified color	فهرس أقرب لون من لون معين
ImageColorExact	Get index of specified color	فهرس لون فهرس
ImageColorResolve	Get index of specified color or closest alternative	فهرس لون معين أو أقرب بديل
ImageGammaCorrect	Apply gamma correction to GD image	تطبيق تصحيح جاما
ImageColorSet	Set color for specific palette index	إعداد لون لفهرس لوحة تلوين
ImageColorsForIndex	Get colors for index	لون فهرس
ImageColorsTotal	Number of colors in image's palette	عدد ألوان تلوين صورة
ImageColorTransparent	Define color as transparent	تحديد لون شفاف
ImageCopy	Copy part of image	نسخ جزء رسم
ImageCopyResized	Copy and resize part of image	نسخ وتحجيم جزء رسم
ImageCreate	Create new image	إنشاء رسم جديد

ImageCreateFrom GIF	Create new image from file or URL	رسم جديد GIF من عنوان مورد موحد
ImageCreateFrom JPEG	Create new image from file or URL	رسم جديد JPG من عنوان مورد موحد
ImageCreateFrom PNG	Create new image from file or URL	رسم جديد PNG من عنوان مورد موحد
ImageDashedLine	Draw dashed line	رسم خط منقط
ImageDestroy	Destroy an image	إتلاف رسم
ImageFill	Flood fill	تعبئة رسم
ImageFilledPolygon	Draw filled polygon	رسم مضلع ملون
ImageFilledRectangle	Draw filled rectangle	رسم مستطيل ملون
ImageFillToBorder	Flood fill to specific color	تعبئة بلون معين
ImageFontHeight	Get font height	معرفة ارتفاع خط كتابة
ImageFontWidth	Get font width	معرفة عرض خط كتابة
ImageGIF	Out image to browser or file	خرج شكل GIF إلى المستعرض أو ملف
ImagePNG	Out PNG image to browser or file	خرج شكل PNG للمستعرض أو ملف
ImageJPEG	Output image to browser or file	خرج شكل JPG

		للمستعرض أو ملف
ImageInterlace	Enable or disable interlace	تشغيل أو تعطيل واجهة بينية
ImageLine	Draw line	رسم خط
ImageLoadFont	Load new font	تحميل خط كتابة
ImagePolygon	Draw polygon	رسم مضلع
ImagePSBBox	Bound text box with PostScript Type1 fonts	مربع نص
ImagePSEncodeFont	Change character encoding vector of a font	تغيير متجه تشفير حروف خط
ImagePSFreeFont	Free memory used by PostScript Type 1 font	تحرير الذاكرة
ImagePSLoadFont	Load PostScript Type 1 font from file	تحميل خط بوست سكريبت من ملف
ImagePSText	Draw text string over image with PostScript1 fonts	رسم نص فوق صورة باستخدام بوست سكريبت
ImageRectangle	Draw a rectangle	رسم مستطيل
ImageSetPixel	Set a single pixel	وضع بكسل
ImageString	Draw string horizontally	رسم نص أفقيا
ImageStringUp	Draw string vertically	رسم نص رأسيا
ImageSX	Get image width	معرفة عرض رسم

ImageSY	Get image height	معرفة ارتفاع رسم
ImageTTFBBox	Give bound text box by True Type	مربع محيط بنص
ImageTTFText	Write text to image by True Type	كـــتابة نص على رسم بخطوط حقيقة
ImageTypes	Return image type	نوع الرسم

لكل دالة من هذه الدوال معاملاتها كما يتضح من الأمثلة التالية :

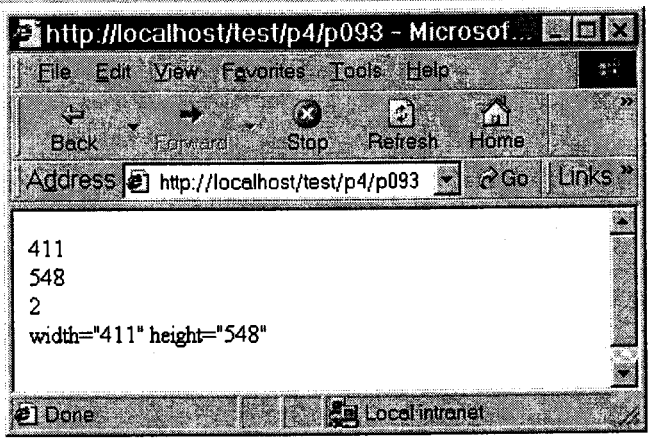
تقوم دالة حجم الصورة بطباعة طول وعرض أى صورة وتكتب الدالة بشكل :

getimageSize("Name of picture")

دالة معرفة حجم صورة getimagesize تعطى معلومات الصورة على هيئة مصفوفة فيها : العنصر الأول عرض الصورة ، والعنصر الثانى ارتفاع الصورة ، والعنصر الثالـــث نوع امتداد تنسيق الصورة أو ناتج متغير نوع الصورة على شكل أرقام تبين نـــوع الصورة على الشكل : (1=GIF ، 2=JPG ، 3=PNG ، 4=SWF ، 5=PSD ، 6=BMP) ، والعنصـــر الـــرابع عرض وارتفاع الصورة مع النص مع وسم عرض الصورة IMG ، مثال :

```
<?
$im1="file1.jpg";
$imd1= getimagesize ($im1) ;
print $imd1[0] . "<BR>" ;
print $imd1[1] . "<BR>" ;
print $imd1[2] . "<BR>" ;
print $imd1[3] . "<BR>" ;
print $imd1[4] . "<BR>" ;
?>
```

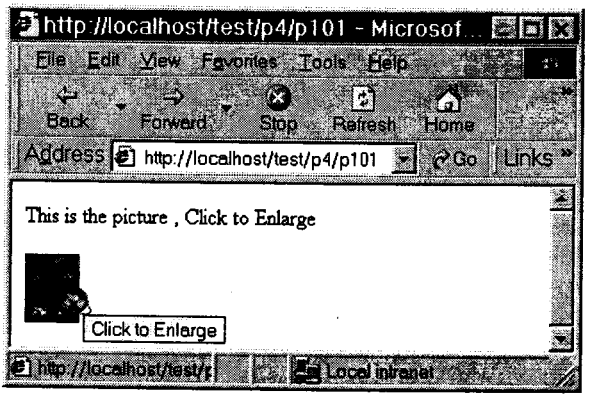
النتيجة :



تستخدم دالة حجم الصورة فى أغراض كثيرة منها معرفة عرض وارتفاع صورة وطباعتها مصغرة مع رابط للصورة المكبرة ، مثال :

```
<?
echo "This is the picture , Click to Enlarge <BR> ";
$a = getimageSize("ab.gif");
if($a[0] > 50 or $a[1] > 50 )
{
echo "<p> <a title=' Click to Enlarge ' href='ab.gif'> <img border='0'
src='ab.gif' width='50' height='50'> </a></p>";
} else {
echo"<p>
<img border='0' src='ab.gif' $a[3]> </p>";
}
?>
```

النتيجة سوف تكون على الشكل :



وعند نقر الصورة سوف تظهر بالمقاس الطبيعي مثل :

تستخدم دالة إنشاء صورة imagecreate لإنشاء صورة جديدة ، وتحتاج الدالة إلى متغير عرض الصورة ، ومتغير ارتفاع الصورة ، وعند إنشاء صورة تعيد الدالة مقبضا للصورة يستخدم مع الدوال الأخرى ، مثال :

```
$pic01 = @ImageCreate (40,80) ;
```

فى المثال إنشاء صورة أو مساحة عمل ، يعود المقبض عن طريق المتغير $pic01 .

تستخدم دالة تخصيص لون الصورة Imagecolorallocate لتعريف لون يستخدم فى رسم الصورة ، مثال :

```
$ColorGreen=ImageColorAllocate ($pic01, 0,0,0) ;
```

وقد استخدم مقبض مساحة العمل مع متغيرات اللون .

تستخدم دالة رسم قوس imagearc فى مكان ما وبنصف قطر ما ، وبزاوية ما لرسم قوس فى منطقة العمل ، مثال :

```
Imagearc ($pic01 , 80,40, 80 , 80 ,0 , 360, $ColorGreen) ;
```

تستخدم الدالة ImageChar لرسم خط فى منطقة العمل ، مثال :

```
Imagechar ($pic01 , 1, 0, 0 , "C" , $ColorGreen) ;
```

تستخدم الدالة ImagefontHeight لتحديد ارتفاع خطوط الكتابة فى منطقة العمل ، كما تستخدم الدالة ImagefontWidth لتحديد عرض خطوط الكتابة فى منطقة العمل مثال :

```
$w2=Imagefontwidth (25) ;
$h2=Imagefontheight (6) ;
```

تستخدم الدالة Imagejpeg لحفظ وإخراج صورة فى منطقة العمل إلى المتصفح أو إلى ملف بتنسيق jpg ، مثال :

```
Imagejpeg ($pic01 , "mypic.jpg") ;
```

تستخدم الدالة ImageGIF لحفظ وإخراج صورة فى منطقة العمل إلى المتصفح أو إلى ملف بتنسيق GIF ، مثال :

```
Imagegif ($pic02 , "mypic.gif") ;
```

تستخدم الدالة Imagepng لحفظ وإخراج صورة فى منطقة العمل إلى المتصفح أو إلى ملف بتنسيق png ، مثال :

```
Imagepng ($pic03, "mypic.png") ;
```

تستخدم الدالة Imageloadfont لتحميل نوع خط كتابة معين ، مثال :

```
$font01=imageloadfont("font01") ;
```

تستخدم الدالة Imagestring لكتابة نص على منطقة العمل مع تحديد مقبض الصورة ولون النص ، مثال :

```
Imagestring ($pic01, 4, 10, 10, "Program sale", $colorGreen) ;
```

أمثلة :

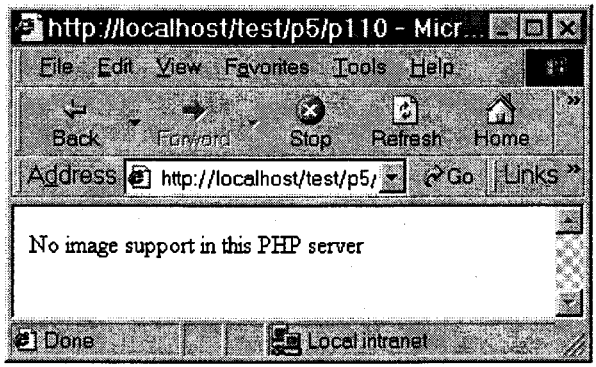
```
$white = ImageColorAllocate ($im, 255, 255, 255);
$black = ImageColorAllocate ($im, 0, 0, 0);
$white = ImageColorAllocate($im, 255, 255, 255);
ImageColorDeAllocate($im, $white);
```

```
// Example Creating GD image stream and output image
header ("Content-type: image/png");
$im = @ImageCreate (50, 100)
or die ("Cannot Initialize new GD image stream");
$background_color = ImageColorAllocate ($im, 255, 255, 255);
$text_color = ImageColorAllocate ($im, 233, 14, 91);
ImageString ($im, 1, 5, 5,  "A Simple Text String", $text_color);
ImagePng ($im);
```

مثال لمعرفة ما إذا الخادم يدعم الصور أم لا :

```
if (function_exists("imagegif")) {
Header("Content-type: image/gif");
   ImageGif($im);
  }
elseif (function_exists("imagejpeg")) {
   Header("Content-type: image/jpeg");
   ImageJpeg($im, "", 0.5);
  }
elseif (function_exists("imagepng")) {
   Header("Content-type: image/png");
   ImagePng($im);
  }
  else
   die("No image support in this PHP server");
```

نتيجة هذا المثال عند تعطيل مكتبة GD هي :



دوال التعامل مع خادم نقل الملفات FTP

تستخدم اللغة عددا من الدوال للتعامل مع خادم نقل الملفات FTP Server ، والتحكم بالملفات والمجلدات فيه مثل :

دالة الاتصال ftp_connect للاتصال بخادم نقل الملفات فإذا تم الاتصال بنجاح تعود بمقبض الاتصال لاستخدامه فى عمليات أخرى ، مثال :

```
$ftp= ftp_connect ("mydomain.xxx") ;
```

دالة الولوج ftp_login تستخدم لتسجيل دخول مستخدم إلى خادم نقل الملفات ، تحتاج الدالة مقبض الاتصال ، واسم المستخدم ، وكلمة السر ، وتعود بقيمة True عند نجاح الدخول ، مثال :

```
$log= ftp_login ($ftp, "user", "Password") ;
```

دالة المجلد الحالى ftp_pwd تستخدم لمعرفة اسم المجلد الحالى وتحتاج مقبض الاتصال وتعود باسم المجلد الحالى ، مثال :

```
$dir01= ftp_pwd ($ftp) ;
```

دالة تغيير المجلد الحالى ftp_chdir تقوم بتغيير المجلد الحالى فى خادم نقل الملفات إلى مجلد آخر ، وتحتاج مقبض اتصال لاستخدامه ، مثال :

```
$cd01 = ftp_chdir ($ftp, "mydir") ;
```

دالة إنشاء مجلد ftp_mkdir تقوم بإنشاء مجلد فى المسار المحدد بخادم نقل الملفات ،

وتحتاج مقبض الاتصال لاستخدامه ، مثال :

```
$md01 = ftp_mkdir ($ftp, "dir03") ;
```

دالـــة حـــذف مجلد ftp_rmdir تقوم بحذف مجلد بخادم نقل الملفات ، وتحتاج مقبض الاتصال لاستخدامه ، مثال :

```
$rd01 = ftp_rmdir ($ftp, "dir01") ;
```

دالـــة عرض محتوى مسار ftp_nlist تعرض محتويات مسار فى خادم نقل الملفات على هيئة مصفوفة ، وتحتاج مقبض الاتصال ، مثال :

```
$listf01 = ftp_nlist ($ftp, ".") ;
```

دالـــة تنزيل ملف ftp_get تقوم بتنزيل ملف من خادم نقل الملفات إلى الكمبيوتر أو الموقـــع ، وتحـــتاج مقبض الاتصال وتحديد نوع النقل (آسكى FTP_ASCII أو ثنائى FTP_BINARY) ، مثال :

```
ftp_get($ftp, "mydir/file01.dat", "dir02/file2.dat", FTP_ASCII) ;
```

دالـــة تحميل ملف ftp_put تقوم بتحميل ملف من الكمبيوتر أو الموقع إلى خادم نقل الملفـــات ، وتحتاج مقبض الاتصال وتحديد نوع النقل (آسكى FTP_ASCII أو ثنائى FTP_BINARY) ، مثال :

```
ftp_put ($ftp, "dir02/file2.dat" , "mydir/file01.dat" , FTP_ASCII) ;
```

دالة حجم الملف ftp_size تقوم بإعادة حجم ملف ، مثال :

```
$size01 =ftp_size ($ftp, "file2.dat") ;
```

دالة تغيير اسم ملف ftp_rename تقوم بتغيير اسم ملف ، مثال :

```
$name01 =ftp_rename ($ftp, "file2.dat", "aaa.dat") ;
```

دالة حذف ملف ftp_delete تقوم بحذف ملف ، مثال :

```
$name02 =ftp_delete ($ftp, "file2.txt") ;
```

دالة إرسال الأوامر إلى الخادم ftp_site لإرسال أوامر للخادم ، مثال :

```
$n03 =ftp_site ($ftp, "cd") ;
```

دالة خروج وإغلاق اتصال الخادم ftp_quit مثال :

```
$n04 =ftp_quit ($ftp) ;
```

قواعد البيانات

مـــن خصـــائص نظـــام إدارة قواعد البيانات العلاقية RDBMS سهولة الوصول إلى البيانات المخزنة وسرعة الاستعلام ، وتتكون قواعد البيانات من جداول ، والجداول **Tables** عبارة عن أعمدة وصفوف تحتوى على خلايا قيم بيانات .

الأعمـــدة **Columns** : لكل عمود فى الجدول اسم خاص ، ولكل عمود نوع خاص يصف نوع البيانات التى تخزن فيه ، مثل النوع الرقمى Integer ، والنصى Text .

الرقم	الاسم	الدرجة
١	أحمد محمد عبد الفتاح	٩٨
٢	أحمد محمد فاضل	٩٧
٣	أحمد محمد مصطفى	٩٩
٤		

الصـــفوف **Rows** : يحتوى كل صف على خلايا قيم مختلفة ويمثل الصف معلومات متكاملة (السجل) .

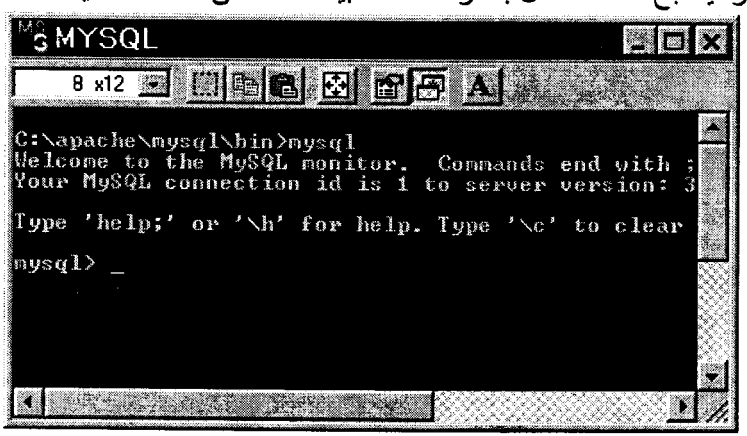
القيم **Values** : محتويات خلايا تقاطع الصفوف والأعمدة .

المفاتـــيح **Keys** : مـــن طـــرق تيسير الوصول لبيانات قواعد البيانات ، مثل أرقام تعريف Id متسلسلة لا تتكرر ، وتعتبر مفتاحا مناسبا لكل صف .

تستطيع لغة PHP التعامل مع نطاق واسع من قواعد البيانات ، وتستخدم معها مجموعة كبيرة من الدوال المتشابهة فى الاستخدام مع تغيير اسم قاعدة البيانات ، كما يجب تركيب قاعدة البيانات لاستخدامها .

التعامل مع قواعد البيانات MySQL

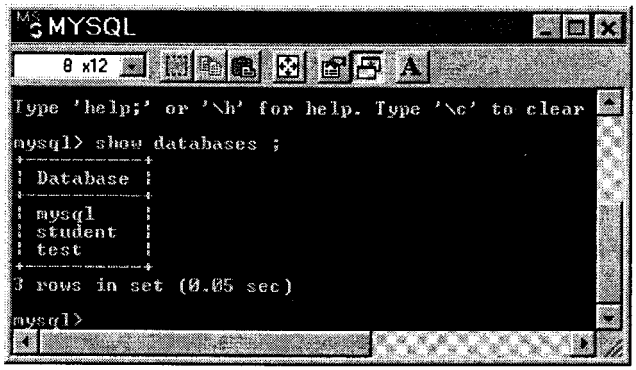
يعتبر محرك قواعد بيانات MySQL من أقوى محركات قواعد البيانات ، وهو مجانى مفتوح المصدر ، يمكن استخدام هذا المحرك عن طريق موجه الأوامر DOS Command كما يلى : الخروج إلى موجه الأوامر ثم تغيير المسار إلى المسار C:\apache\mysql\bin ، وكتابة الأمر mysql وضغط مفتاح الإدخال يظهر موجه الأوامر ليصبح هناك اتصال بمحرك قاعدة البيانات كما فى الشاشة التالية :



يمكن استعراض قواعد البيانات المتوفرة ، والجداول الموجودة فى كل قاعدة بيانات ، والأعمدة والحقول الموجودة فى الجداول ، ولمعاينة قواعد البيانات المتوفرة نكتب الأمر :

```
Show databases ;
```

تكون النتيجة مشابهة للشكل التالى :



للتعامل مع قاعدة بيانات معينة يكتب أمر استخدامها على الصورة :

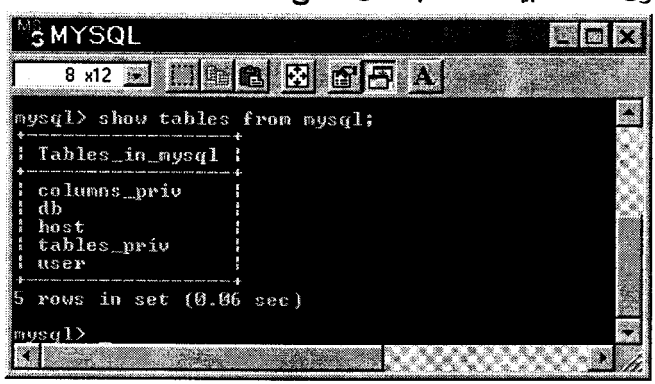
Use database_name حيث نضع بدلا من كلمة database_name اسم قاعدة البيانات مثل استخدام قاعدة البيانات mysql يكون الأمر :

Use mysql ;

وعندما يصدر أى استفسار يتم التعامل مع هذه القاعدة ، ولمعرفة جداول موجودة فى قاعدة البيانات نكتب الأمر :

SHOW TABLES FROM mysql ;

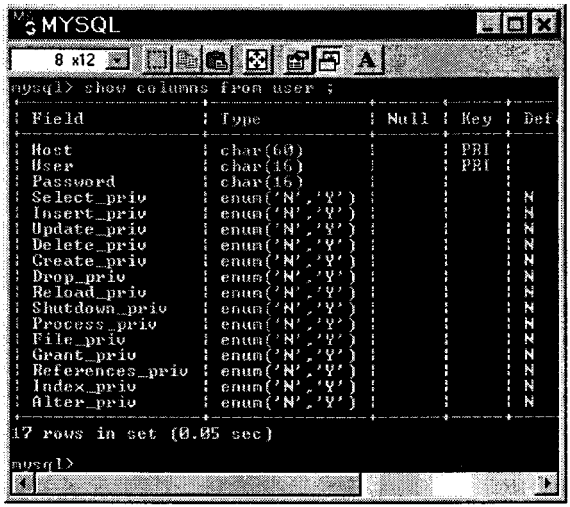
فتظهر جداول قاعدة البيانات كما بالشكل التالى :



ولمعرفة حقول جدول من الجداول مثل الجدول user نكتب الأمر :

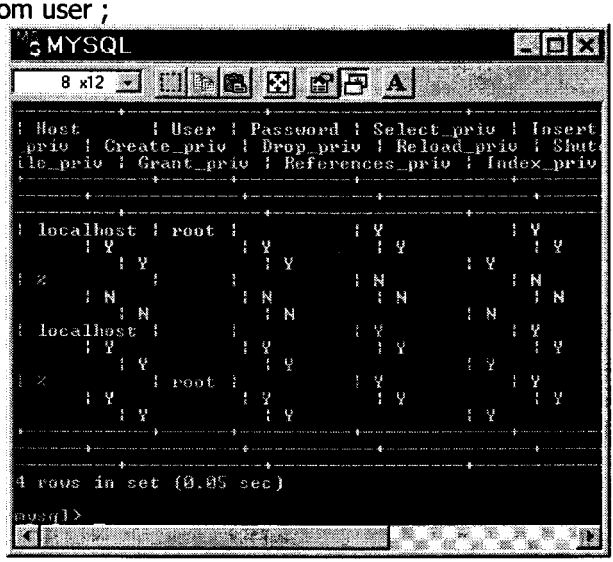
SHOW COLUMNS FROM user ;

فتظهر حقول الجدول كالتالى :



لمعرفة البيانات الموجودة فى جدول user يكتب الأمر التالى :

Select * from user ;

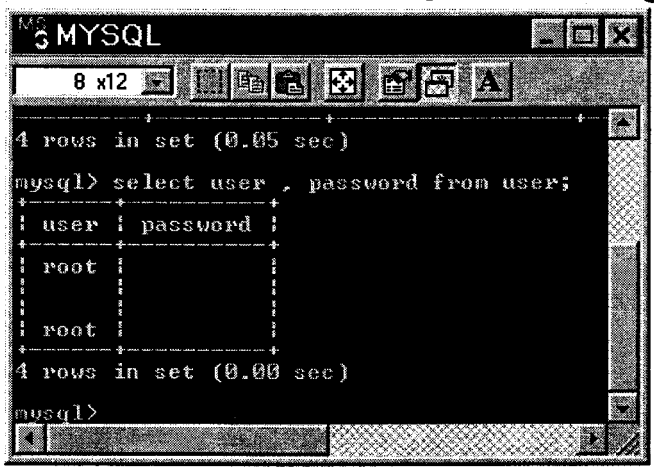


عند تنفيذ الأمر تكون البيانات كثيرة ، وقد نحتاج ترتيبها أو عرض القليل منها .

يمكـــن ترتيب البيانات وتقليل المعروض منها بطلب عرض أجزاء من البيانات مثل اختيار Select عرض جزء من البيانات مثل عرض الأسماء وكلمات السر كالتالى :

Select user , password from user ;

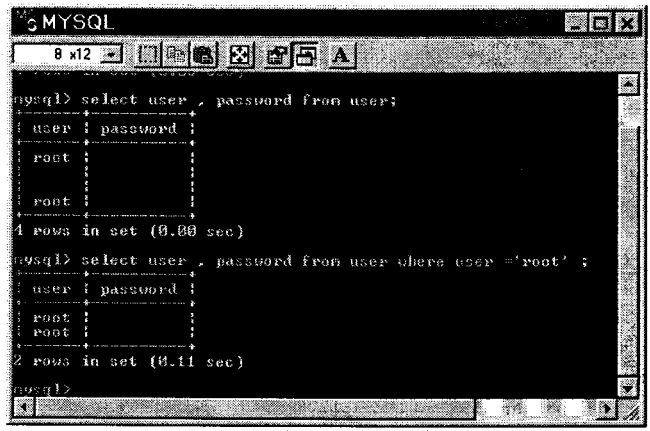
تظهر جميع البيانات الموجودة فى الجدول مثل :



لمعـــرفة بيانات مستخدم معين مثل المستخدم root نكتب الأمر محددا هذا المستخدم على الصورة :

Select user , password from user where user = 'root' ;

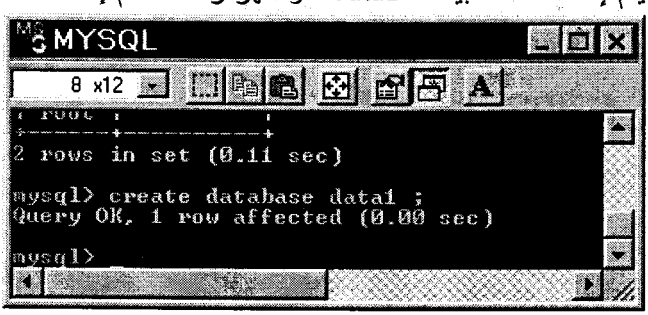
ليتم عرض ما يشبه الشكل التالى :



لإنشـــاء قاعدة بيانات يستخدم أمر إنشاء Create مع اسم قاعدة البيانات الجديدة التى سيتم إنشاؤها مثل التالى :

```
Create database data1 ;
```

بهذا الأمر يتم إنشاء قاعدة البيانات data1 ، وتظهر رسالة تمام إنشاء قاعدة البيانات.



يمكن حذف قاعدة البيانات تلك عن طريق الأمر :

```
Drop database data1 ;
```

عند إنشاء جدول فى قاعدة بيانات يجب إنشاء قاعدة البيانات أولا ، ثم استخدام قاعدة البيانات للعمل ، وبعد ذلك يمكن إنشاء أو حذف أو تحرير الجداول فيها ، فى المثال التالى إنشاء قاعة البيانات ، ثم فتحها للاستخدام ، ثم إنشاء جدول فيها :

```
Create database data01 ;
```

```
Use data01 ;
Create table table01 (no INT(6) NOT NULL , name VARCHAR (30) NOT
NULL , sal INT (4) NOT NULL) ;
```

فـــى الســـطر الأول إنشاء قاعدة البيانات ، وفى السطر الثانى فتحها للاستخدام ، فى السطر الثالث طلب إنشاء جدول فى قاعدة البيانات يتكون من ثلاثة حقول (أعمدة) :

الأول فيها no لرقم بعدد صحيح INT طوله ٦ خانات ليس فارغا NOT NULL .

الثانى حقل الاسم name يتكون من ثلاثين حرفا .

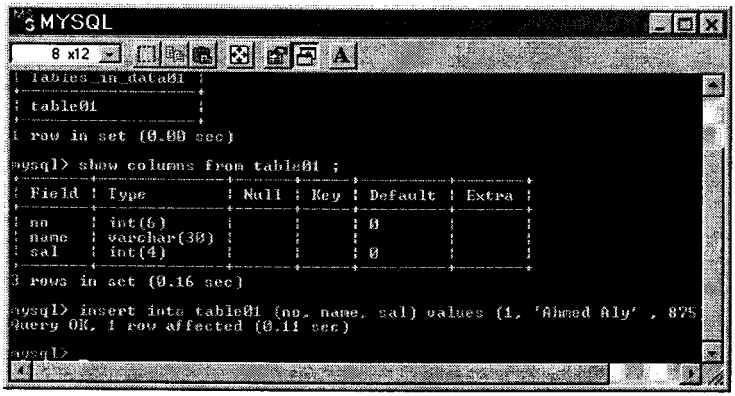
الثالث حقل المرتب مكون من أرقام صحيحة بأربع خانات .

عند تنفيذ تلك الأوامر سوف تظهر رسائل تبين تمام التنفيذ .

يمكن إضافة البيانات للجدول بالأمر :

```
Insert into table01 (no, name,  sal) Values (1, "Ahmed Aly", 875) ;
```

وعند تنفيذ الأمر تظهر رسالة إتمام تنفيذ العملية .



يمكن حذف الجدول باستخدام الأمر :

```
Drop table table01 ;
```

فى كل أحوال صدور أوامر تظهر رسالة تبين ما تم حيال تنفيذ الأمر .

تعليمات لغة SQL

تستخدم كافة قواعد البيانات لغة الاستفسار المهيكلة Structured Query Language (SQL) كلغة قياسية للتعامل مع قواعد البيانات وبيانات قواعد البيانات من أجل إنشاء وحذف وتحرير وتعديل واستعراض البيانات فى حقول وسجلات جداول قواعد البيانات .

تحتوى اللغة على تعليمات للتعامل مع الجداول والسجلات والحقول ، وفيما يلى عرض موجز لبعض هذه التعليمات :

تعليمة الاختيار Select :

تستخدم هذه التعليمة لاسترجاع البيانات من جدول أو عدة جداول فى قاعدة البيانات ، وتستخدم بالصيغة :

```
Select col1, col2, ... ... from table1, table2 , .. ;
```

مثال لاسترجاع رقم واسم ودرجات طالب من جدول الطلاب mydata يجب فتح ملف قاعدة البيانات للاستخدام أولا باستخدام أمر use mydata ثم طرح الاستفسار على الصورة :

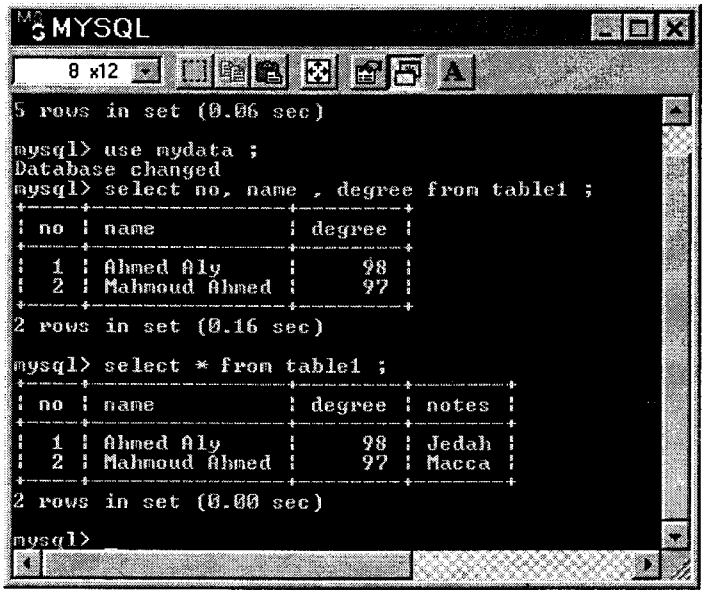
```
Select no, name, degree from table1 ;
```

تفصل بين كل حقل وآخر علامة الفاصلة ، وعند تنفيذ الجملة تحصل على النتيجة .

لاستعراض كل حقول فى الجدول يمكن كتابة :

```
Select * from table1 ;
```

وفى هذه الحالة تظهر النتائج على شكل :



تعليمة الشرط Where :

تستخدم للحصول على بيانات ينطبق عليها شرط أو عدة شروط :

Select col1, .. from table1, table2, .. where condition =??? ;

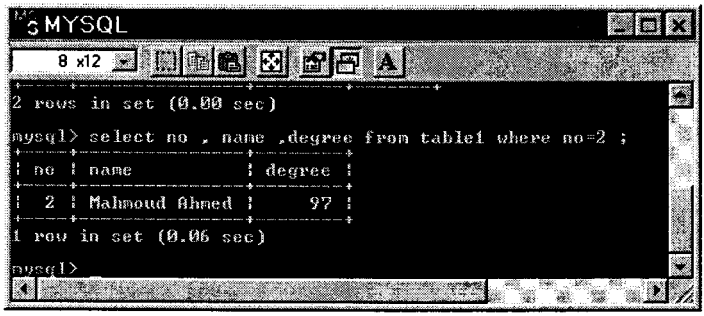
سوف نقوم بإضافة بيانات إلى الطلبة حتى يمكن توسيع نطاق الاستفسارات :

مثال :

لمعرفة بيانات الطالب رقم ٧ فى جدول الطلاب :

Select no , name, degree from table1 where no=2 ;

سوف تظهر بيانات الطالب على الشكل التالى :



مثال آخر : لعرض بيانات الطالب بالاسم :

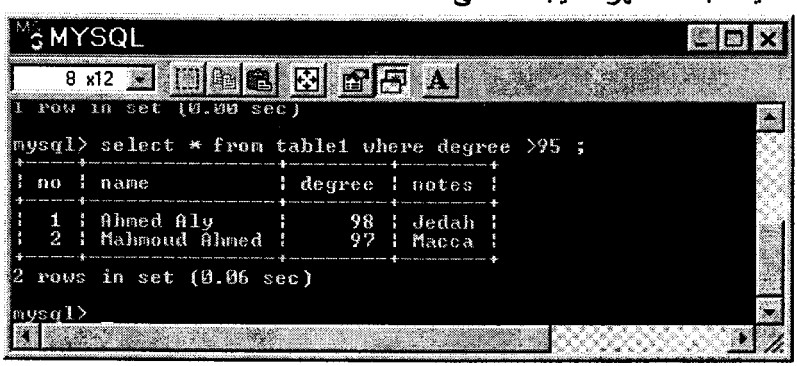
Select * from table1 where name='Ahmed Aly' ;

يتضـــح أن من الأفضل وضع اسم الطالب فى حقل واسم الأب فى حقل واسم العائلة الأخير فى حقل فتقسيم البيانات يساعد على الوصول إلى نتائج أفضل .

يمكـــن أيضا استخدام علامات المقارنة مثل أقل من أو أكبر من أو أكبر من ويساوى أو أقل من ويساوى ، مثال لعرض بيانات الطلاب الحاصلين على درجات تزيد عن ٩٥ درجة :

Select * from table1 where degree >95 ;

عند تنفيذ الجملة تظهر النتيجة كالتالى :



علامة التشابه Like :

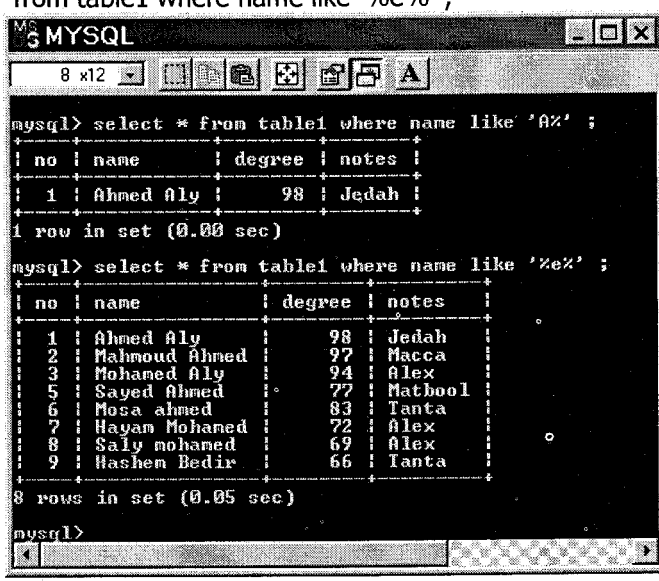
تعـــبر هذه العلامة عن مثل أو شبه أو يحتوى على ، وهى علامة تتيح مقارنة جزء

من المعلومات ، مثلا يمكن عرض جميع أسماء جميع الطلاب بالسجلات التى تحتوى على حرف A فى بداية الاسم نكتب :

Select * from table1 where name like 'A%' ;

فى المثال التالى عرض كل طلاب تحتوى أسماؤهم على حرف e :

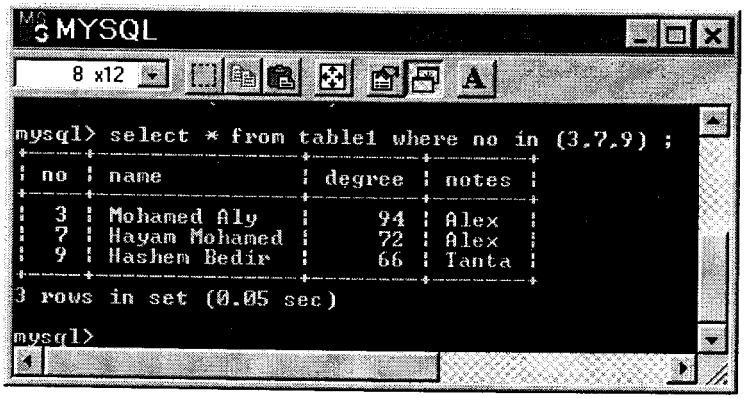
Select * from table1 where name like '%e%' ;



علامة فى IN :

تعـد علامـة تحقق شرط عند وجود أحد معاملاتها ، مثال : لعرض بيانات الطلاب بأرقام معينة نكتب :

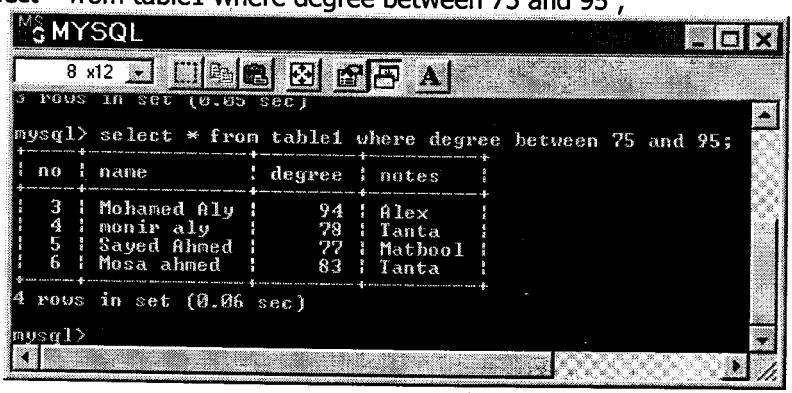
Select * from table1 where no in (3, 7, 9) ;



علامة بين Between :

علامـــة شـــرط عند تحقق وجود الشرط بين حدى معاملاتها ، مثال : لعرض بيانات الطلاب بدرجات بين قيمتين نكتب :

Select * from table1 where degree between 75 and 95 ;

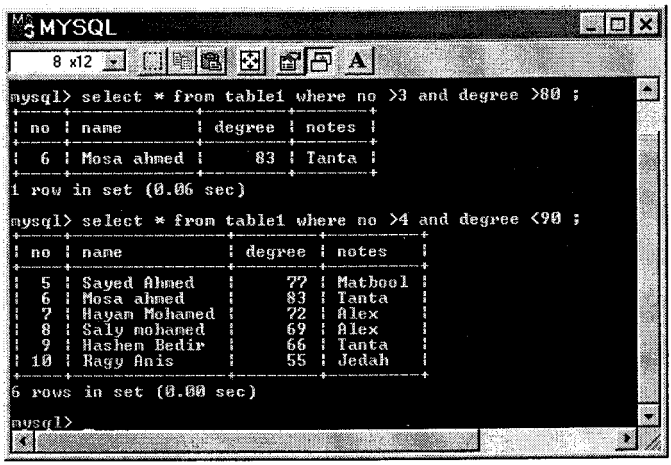


يمكن دمج عدة شروط معا ، مثال : عرض بيانات الطلاب لأرقام معينة ، ودرجات معينة :

Select * from table1 where no > 3 and degree > 80 ;

مثال آخر :

Select * from table1 where no > 4 or degree < 90 ;



تعليمة الترتيب بواسطة Order by :

تستخدم لترتيب بيانات بناء على حقل ترتيبا تصاعديا (من الأصغر إلى الأكبر) ASC ، أو تنازليا (من الأكبر إلى الأصغر) DESC .

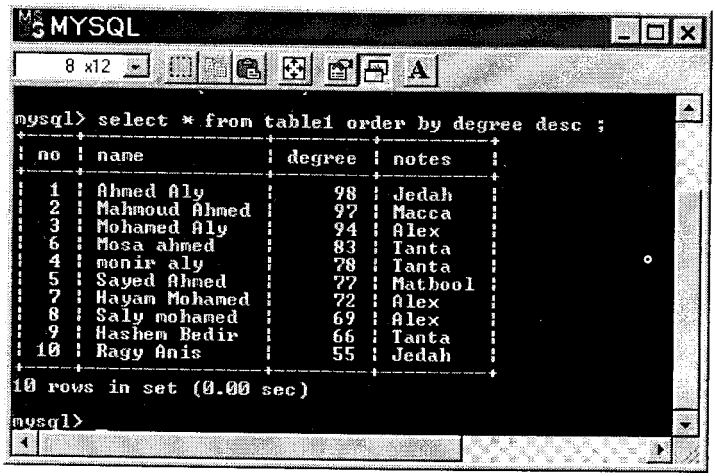
مثال :

للحصول على بيانات الطلاب مرتبة من الأصغر للأكبر نكتب :

Select * from table1 order by degree asc ;

مثال : للحصول على بيانات الطلاب مرتبة من الأكبر للأصغر نكتب :

Select * from table1 order by degree desc ;



تعليمة حدود العرض Limit :

تحدد الحد الأقصى لعدد السجلات التى يمكن الحصول عليها .

مثال لتحديد عدد أربعة سجلات فقط :

```
Select * from table1 limit 4 ;
```

لاستعراض ثلاثة سجلات تالية للسجل الرابع نكتب :

```
Select * from table1 limit 4, 3 ;
```

تعليمة الإدراج Insert :

تستخدم لإدراج بيانات سجل جديد فى جدول .

مثال : لإضافة سجل طالب جديد اسمه maged ، ودرجاته ٧٦ ، فى السجل الحادى عشر نكتب :

```
Insert into table1 (no, name, degree, notes) Values (11, 'Maged Nasef' , 76, 'Madina') ;
```

أو نكتب :

```
Insert into student Values (12, 'Ahmed Bakr' , 91, 'Tanta") ;
```

بشرط ترتيب الحقول كما هى فى الجدول .

عند تنفيذ الأمر نحصل على رسالة تأكيد بإضافة البيانات .

تعليمة التحديث Update :

تستخدم لتعديل بيانات سجل فى جدول .

مثال : لتعديل سجل طالب بدرجاته إلى ٨٦ ، فى السجل التاسع نكتب :

```
Update table1 set degree = 86 where no =9 ;
```

عند تنفيذ الأمر نحصل على رسالة تأكيد تعديل البيانات .

بكتابة الشرط يتم تحديث السجل للطالب رقم 9 فقط ، وإذا لم يكتب الشرط سيتم تحديث جميع بيانات الحقول فى الجدول .

تعليمة الحذف Delete :

تستخدم لحذف سجل فى جدول . مثال : لحذف السجل التاسع نكتب :

```
Delete from table1 where no =9 ;
```

عند تنفيذ الأمر نحصل على رسالة تنفيذ العملية .

بكتابة الشرط يتم حذف سجل الطالب رقم 9 فقط ، وإذا لم يكتب الشرط مثل :

```
Delete from table1 ;
```

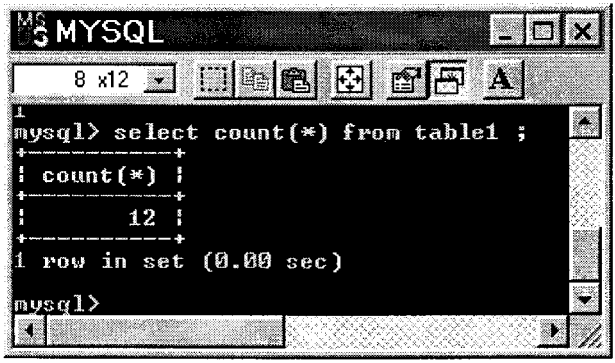
سيتم حذف جميع سجلات الجدول .

تعليمة عد السجلات Count :

تستخدم للحصول على عدد سجلات جدول .

مثال : لمعرفة عدد طلاب نكتب :

```
Select Count (*) from table1 ;
```

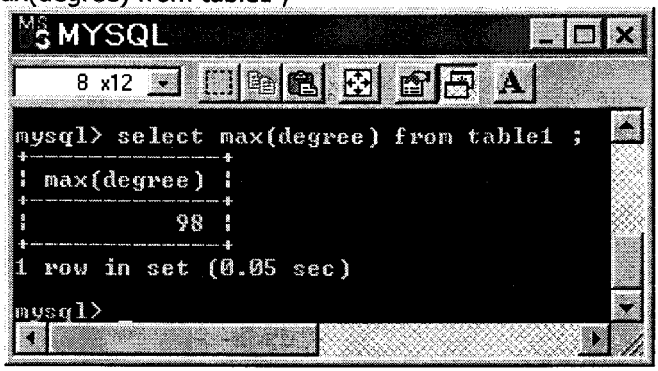


تعليمة الحد الأقصى Max :

تستخدم للحصول على أكبر قيمة فى السجلات لحقل من سجلات جدول .

مثال : للحصول على القيمة العظمى لدرجات طالب نكتب :

```
Select max(degree) from table1 ;
```



تعليمة الحد الأدنى Min :

تستخدم للحصول على أقل قيمة فى السجلات لحقل من سجلات جدول .

مثال : للحصول على القيمة الصغرى لطالب نكتب :

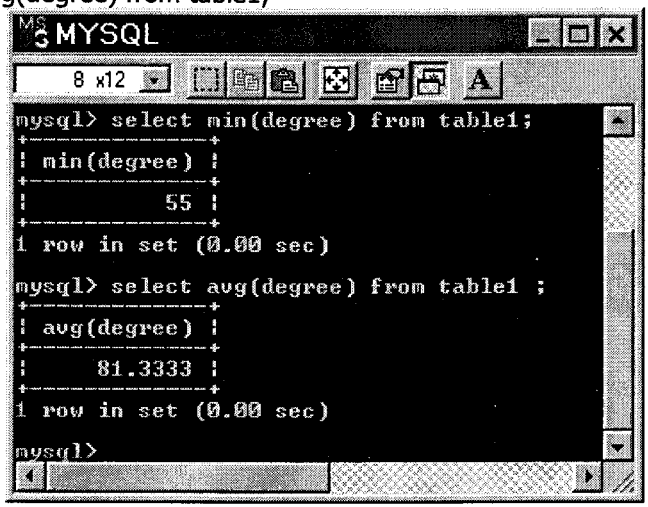
```
Select min(degree) from table1 ;
```

تعليمة المتوسط AVG :

تستخدم للحصول على متوسط حقل لسجلات جدول .

مثال : للحصول على معدل درجات الطلاب نكتب :

```
Select avg(degree) from table1;
```



تعليمة المجموع Sum :

تستخدم للحصول على مجموع قيمة سجلات حقل جدول .

مثال : للحصول على مجموع درجات الطلاب نكتب :

```
Select sum(degree) from table1 ;
```

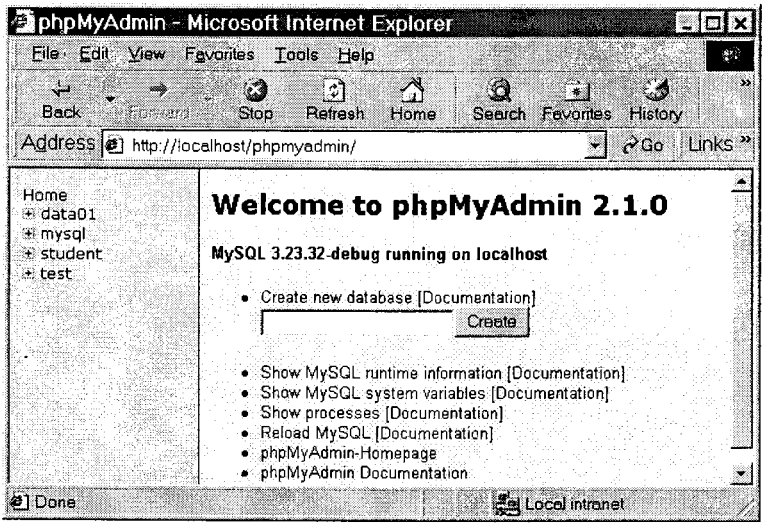
استخدام البرامج الجاهزة

هناك مجموعة من البرامج الجاهزة التى تستخدم لإدارة قواعد البيانات وإنشاء قواعد بيانات وجداول والاستفسار منها .

يمكن استخدام الأدوات المرفقة مع لغة PHP للتحكم فى قواعد البيانات MySQL بصورة أكثر سرعة وسهولة من إصدار الأوامر المكتوبة .

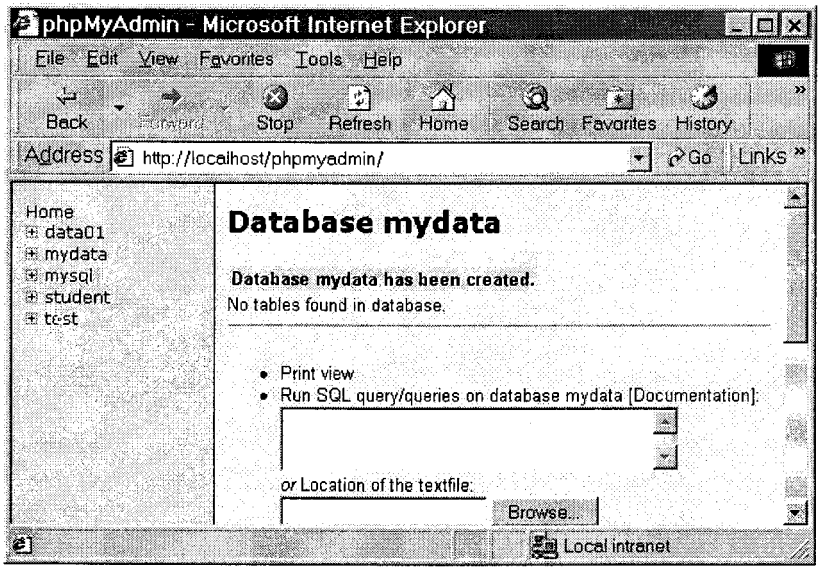
يستخدم برنامج phpMyAdmin المضمن مع أى خادم مستضيف يدعم قاعدة البيانات MySQL ، وعند استخدام الحاسب الشخصى المحلى يكون البرنامج على الحاسب

بالعنوان http://localhost/phpmyadmin وعند كتابة العنوان فى مربع عنوان المتصفح يعمل البرنامج (بشرط تشغيل كل من خادم أباتشى وقاعدة البيانات MySQL)



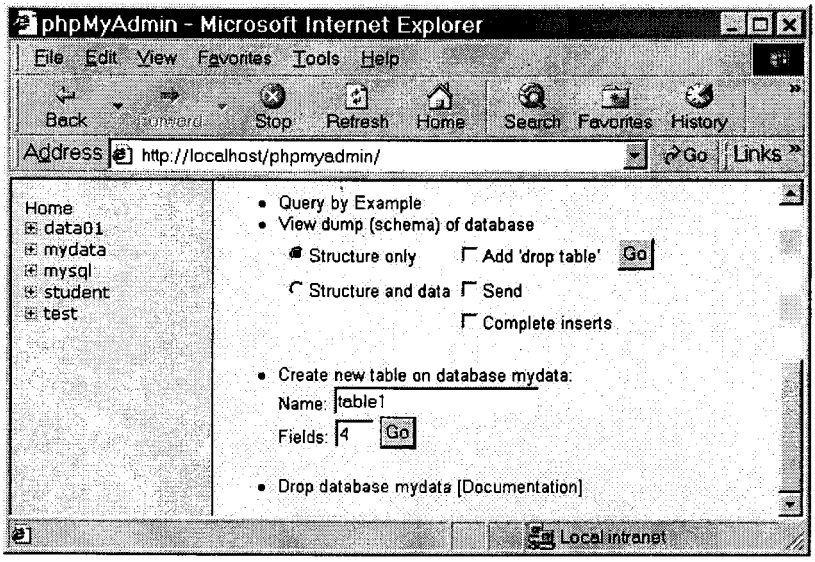
يبين البرنامج فى جهة اليسار قواعد البيانات الموجودة حاليا ، كما يمكن إنشاء قاعدة بيانات جديدة ، وإنشاء جداول فيها .

لإنشاء قاعدة بيانات جديدة يتم كتابة اسم قاعدة البيانات الجديدة فى المربع الموجود أسفل نص إنشاء قاعدة بيانات جديدة Create new database ، ثم نقر زر إنشاء Create ، وسوف يتم إنشاء قاعدة البيانات الجديدة خالية فارغة :



عـــندما تظهـــر الشاشـــة الجديدة تكون خالية من الجداول ، وتجد فى أسفل الشاشة اختيارات إنشاء جدول جديد بكتابة اسم الجدول وتحديد حقول الجدول .

لإنشـــاء جدول جديد فى قاعدة البيانات الجديدة التى تم إنشاؤها يتم كتابة اسم الجدول فـــى مربع اسم Name الجدول أسفل كلمات إنشاء جدول جديد فى قاعدة البيانات مع اســـم قاعدة البيانات :Create new table on database mydata ، وبعد كتابة اسم الجدول يتم كتابة عدد حقول (أعمدة) هذا الجدول بجوار مربع الحقول Fields .

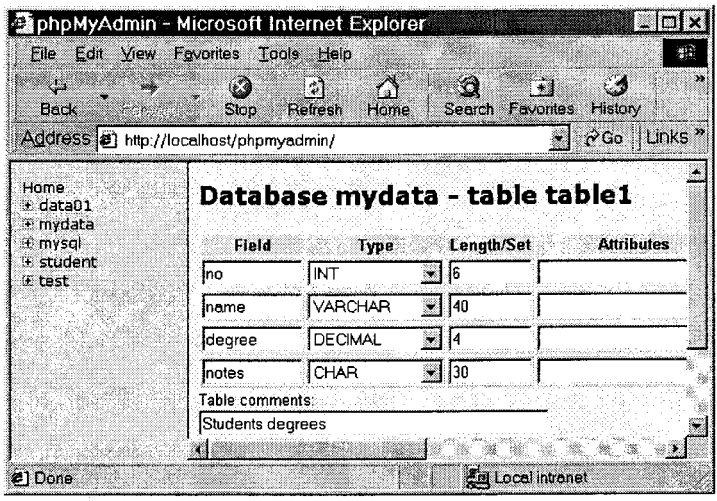


عندما تنقر على زر الذهاب Go يظهر شكل تكوين الجدول .

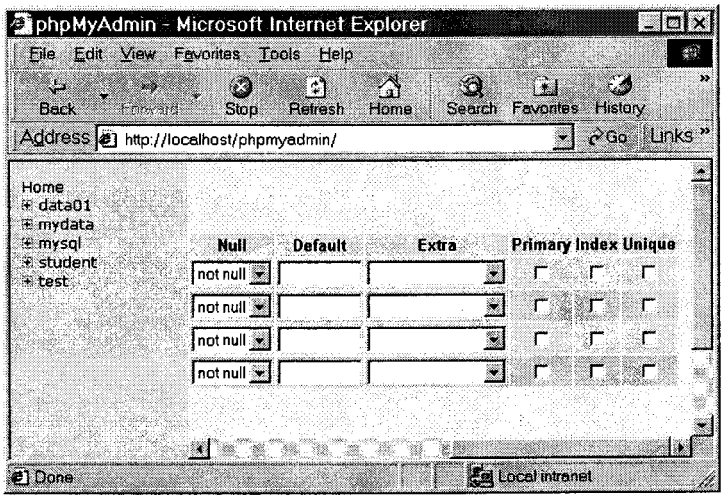
فـــى شكل تكوين الجدول يتم كتابة اسم كل حقل فى مربع الحقل Field ، وتحديد نوع الحقـــل فـــى مربع النوع Type (نص أو رقم أو تاريخ ،) ، وكتابة طول الحقل فى مربع الطول والإعداد Length/Set .

يمكن تحديد مواصفات الحقل فى مربع المواصفات Attribute ، فى ذات الوقت يمكن تحديـــد حالـــة احتواء الحقل على فراغ أو لا Not Null فى خانة الفراغ Null ، كما يمكن كتابة تعليق مصاحب للجدول .

فى الشكل التالى جزء من المعروض على الشاشة .

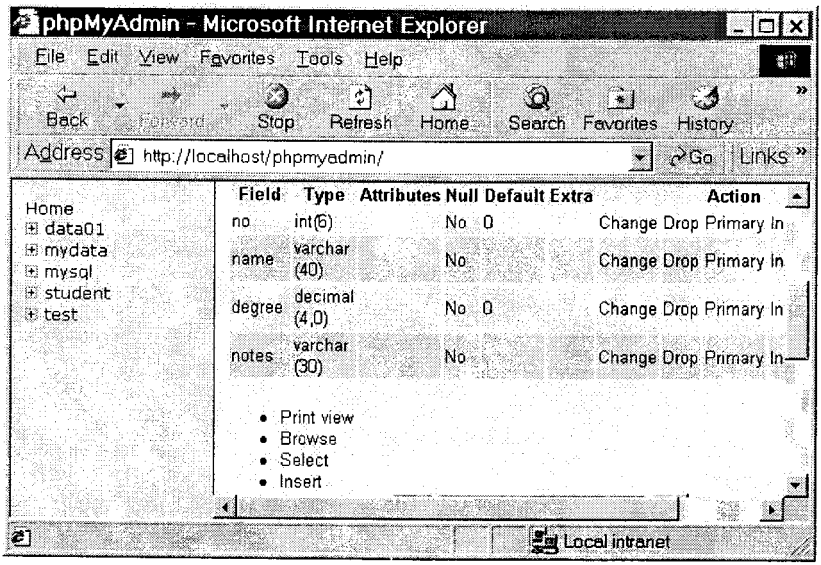


بعد استكمال البيانات الجزء الباقى :



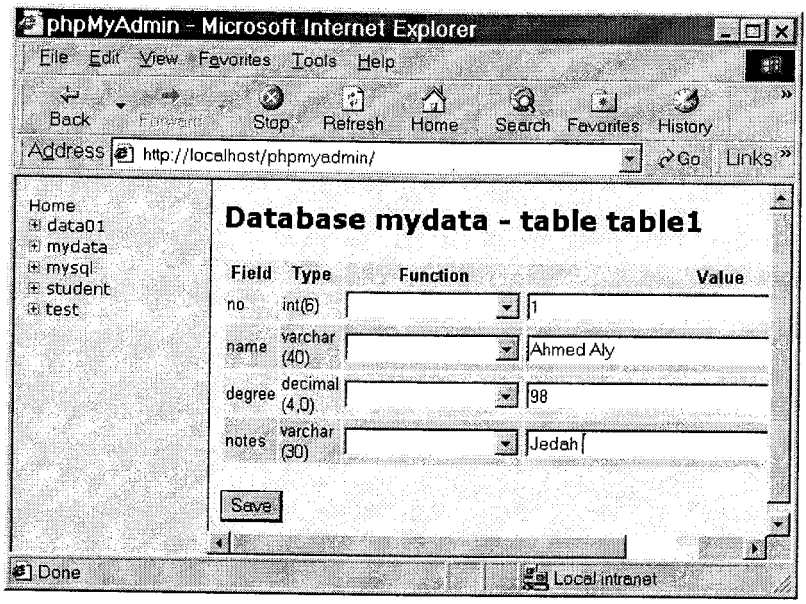
تحت خانة الحالة الافتراضية Default يمكن كتابة قيمة افتراضية لحقل ما ، ويمكن تحديده كمفتاح أساسى Primary ، أو كمفهرس Index ، أو كوحيد Unique بقيمة لا تتكرر .

نقر زر الحفظ Save فى الشاشة بعد إدخال بيانات تكوين الجدول يسبب إنشاء وحفظ الجدول ، وفى هذه الحالة تظهر شاشة تعرض شكل الجدول وبيانات حقوله :



تحـــت شكل تركيب الجدول أوامر طباعة عرض Print View واستعراض Browse واختيار Select وإدراج Insert التى تستخدم للتعامل مع البيانات فى الجدول .

بعـــد تكوين الجدول تكون حقوله فارغة لأنه لم يتم إدخال بيانات فيه ، ويمكن إضافة البـــيانات عن طريق نقر زر الإدراج Insert لتظهر بيانات الجدول فارغة بسجلات خالية :



بعد وضع قيم البيانات فى الجدول يمكن حفظها بنقر زر الحفظ Save .

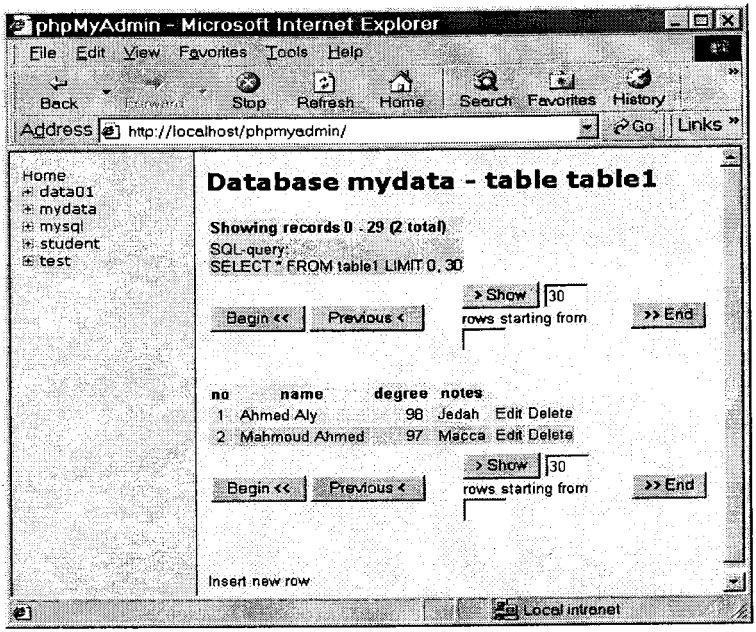
لاستعراض محتويات الجدول يستخدم زر الاستعراض Browse الموجود أسفل بيانات الجدول ، وعندما تكون البيانات كثيرة يتم عرض أول ثلاثين سجلا من سجلات الجدول ، وتستخدم أزرار التنقل على النحو التالى :

زر البداية Begin لعرض أول ثلاثين سجلا .

زر النهاية لعرض آخر ثلاثين سجلا .

زر السابق Previous يعرض الثلاثين سجلا السابقة .

زر العرض Show يعرض ثلاثين سجلا بداية من سجل معين أو من آخر سجل ظاهر .



يمكن تحرير Edit سجل بنقر زر التحرير Edit المجاور للسجل .

يمكن حذف Delete سجل بنقر زر الحذف Delete المجاور للسجل .

كما يمكن إدخال سجل (صف) Row جديد بنقر زر إدراج صف جديد Insert new row في أسفل الشاشة .

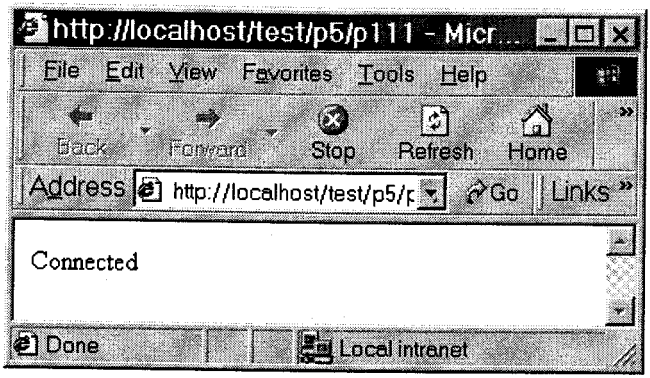
استخدام لغة MySQL فى لغة PHP

يجد الزائر شاشات استخدام قاعدة البيانات فى الموقع مباشرة ، ويتم الاستفادة من قاعدة البيانات عن طريق :

١- الاتصال أولا بقاعدة البيانات باستخدام دالة الاتصال mysql_connect ، وتحتاج الدالة ثلاثة متغيرات هى : اسم الخادم Server Name ، واسم المستخدم User Name ، وكلمة السر Password ، وتعود الدالة بمقبض الاتصال عند نجاح

الاتصال ، أو رسالة خطأ عند فشل الاتصال مثال :

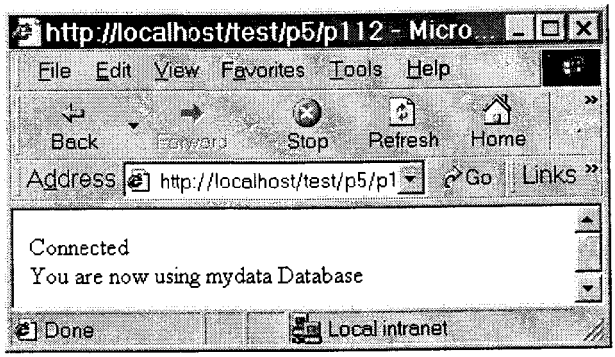
```
<? Php
$link1=mysql_connect ("localhost" , "root" , "") or die ("Not connected")
;
Print ("Connected") ;
Mysql_close ($link1) ;
?>
```



فـــى المثال عند تمام الاتصال تظهر عبارة Connected ، وعند فشل الاتصال تظهر عبارة Not connected ، كما تم إنهاء الاتصال بالعبارة mysql_close .

٢- تحديـــد قاعدة البيانات التى يتم التعامل معها من خلال دالة اختيار قاعدة البيانات mysql_select_db ، لـــتحديد قـــاعدة البـــيانات التى سيتم التعامل معها حاليا ، وتحتاج الدالة إلى متغيرين هما : اسم قاعدة البيانات ، ومقبض الاتصال ، وتعود الدالة بقيمة true عند نجاحها ، وقيمة false عند الفشل أو حدوث خطأ ، مثال :

```
<? Php
$link1=mysql_connect ("localhost" , "root" , "") or die ("Not connected")
;
Print ("Connected") ;
Mysql_select_db ("mydata" ,$link1) ;
Echo "<BR> You are now using mydata Database " ;
?>
```



٣- بعد نجاح الاتصال وتحديد قاعدة البيانات يمكن الحصول على جداول قاعدة البيانات والتحكم فيها عن طريق الدوال المتوفرة فى لغة PHP مثل :

دالة عدد الصفوف (السجلات) المتأثرة mysql_affected_rows بعملية ما مثل الحذف Delete أو الإدراج Insert أو التعديل Update ، وتحتاج هذه الدالة مقبض الاتصال .

عند استخدام تعليمة الحذف delete بدون استخدام شرط Where يتم حذف كل سجلات الجدول ، وتعيد هذه الدالة القيمة صفر باعتبار أنه لا توجد سجلات .

دالة عدد الصفوف mysql_num_rows تعيد عدد السجلات التى يتم الحصول عليها بعد استخدام تعليمة select فقط ، ولا تستخدم مع تعليمات الحذف Delete أو الإدراج Insert أو التعديل Update .

دالة الاستفسار mysql_query تعود بقيم ناتجة عن الاستعلام عن استفسار ما فى قاعدة البيانات ، مثال :

```
$Rout =mysql_query ("SELECT * from student") ;
```

بوضع القيم فى متغير $Rout يمكن الاستفادة منها فى دوال أخرى مثل :

دالة جلب قيم مسترجعة من استفسار mysql_fetch_row التى تضع القيم فى مصفوفة يمكن التعامل معها ، مثال :

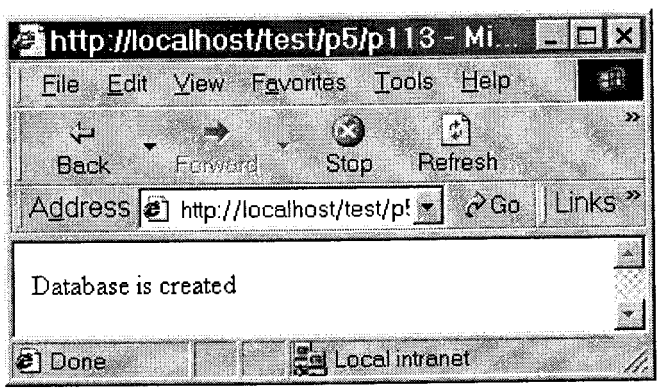
```
$rec1= mysql_fetch_array ($Rout) ;
```

ويمكن بهذا استخدام المصفوفة للتعامل معها .

دالة إنشاء قاعدة بيانات mysql_create_db

تستخدم لإنشاء قاعدة بيانات جديدة ، وتحتاج وضع اسم قاعدة البيانات ، وتعود بالقيمة True عند نجاح تنفيذ المهمة ، مثال :

```
<?
$link1 = mysql_connect ("localhost" , "root" ,"") or die ("not connected")
;
if (mysql_create_db ("products")) {
Print ("Database is created") ;
}
else {
Print ("Not created") ;
}
?>
```

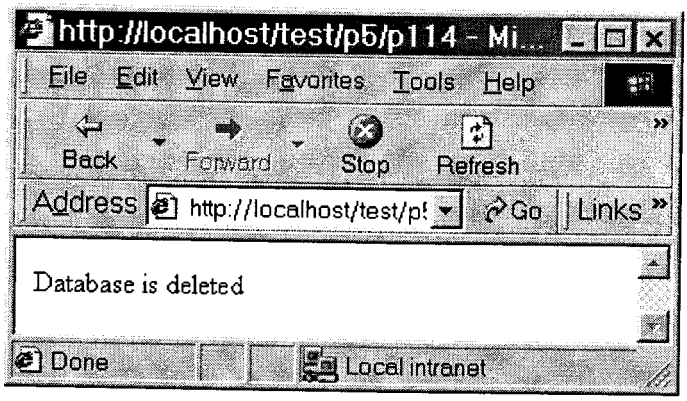


تستخدم دالة حذف قاعدة بيانات mysql_drop_db لحذف قاعدة بيانات موجودة ، وتعود بالقيمة True عند نجاح العملية .

مثال :

```
<?
$link1 = mysql_connect ("localhost" , "root" ,"") or die ("not connected")
;
```

```
if (mysql_drop_db ("products")) {
Print ("Database is deleted ") ;
} else {
Print ("Not found ") ;
}
?>
```



تستخدم دالة قائمة قواعد البيانات mysql_list_dbs لمعرفة قواعد البيانات المتوفرة ، مثال :

```
$d01 = mysql_list_dbs ($link1) ;
```

ويمكن الاستفادة بالناتج الموجود فى المتغير باستخدام دالة الجلب mysql_fetch_row لوضع الناتج فى مصفوفة يمكن التعامل معها .

تستخدم دالة قائمة الجداول mysql_list_tables لمعرفة جداول قاعدة بيانات ، وتحتاج الدالة اسم قاعدة البيانات ، مثال :

```
$l01= mysql_list_tables("products") ;
```

وبوضع الناتج فى متغير يمكن الاستفادة منه باستخدام دالة الجلب mysql_fetch_row لوضع الناتج فى مصفوفة يمكن التعامل معها .

تستخدم دالة قائمة الحقول mysql_list_fields لمعرفة حقول جدول من جداول قاعدة بيانات ، وتحتاج الدالة اسم قاعدة البيانات واسم الجدول ، وتعود الدالة بمقبض

يستخدم لمعرفة معلومات حقول الجدول مثل الاسم والنوع والطول والشروط بالدوال:

دالة معرفة اسم الحقل : mysql_Filed_name .

دالة معرفة نوع الحقل : mysql_Filed_type .

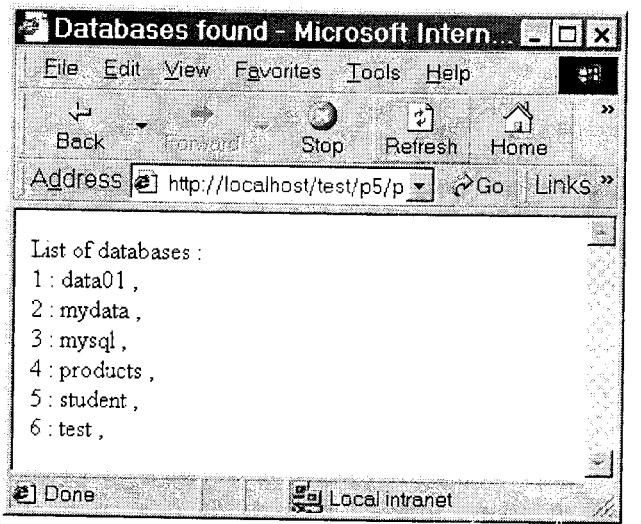
دالة معرفة طول الحقل : mysql_Filed_len .

دالة معرفة شروط الحقل : mysql_Filed_flags .

أمثلة التعامل مع قواعد البيانات

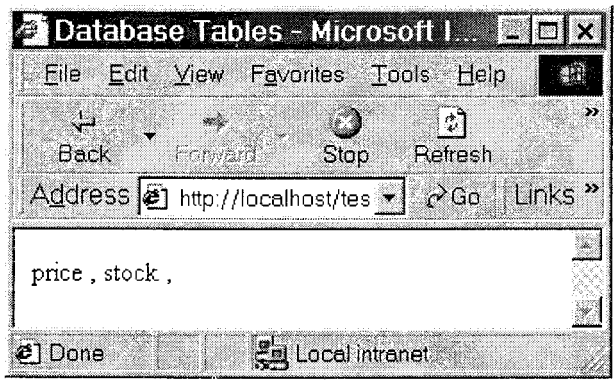
مثال : استعراض قواعد البيانات :

```
<HTML>
<HEAD>
<TITLE> Databases found </TITLE>
</HEAD>
<BODY>
<?php
$link1=mysql_connect ("localhost" , "root" , "") or die ("Not connected")
;
// List of database
$dbase01 = mysql_list_dbs($link1) ;
// Print List of databases
Print " List of databases : <BR> ";
$I=1 ;
while ($D01 =mysql_fetch_row ($dbase01))
{
Print ($I . " : " ) ;
print ($D01 [0] . " , <BR> " ) ;
++$I ;

}
?>
</BODY>
</HTML>
```



مثال : استعراض جداول قاعدة بيانات :

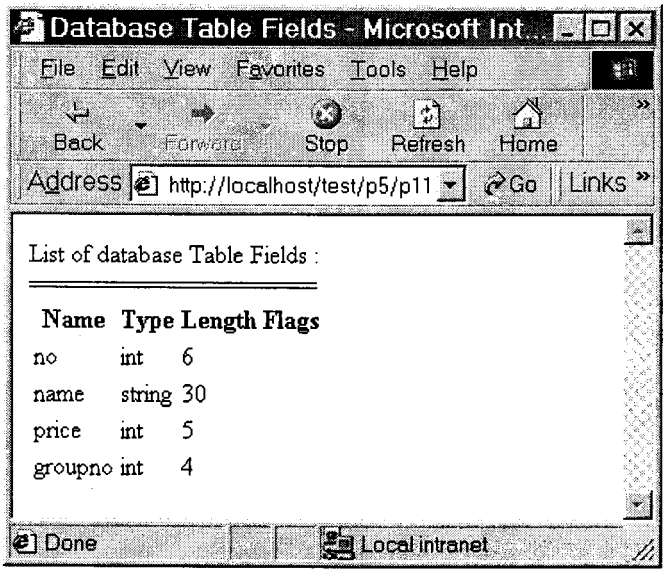
```
<HTML>
<HEAD>
<TITLE> Database Tables </TITLE>
</HEAD>
<BODY>

<?php
$link1 =mysql_connect ("localhost" , "root" , "") ;
// List of database Tables
$dbase02 = mysql_list_tables (products) ;
// Print List of database Tables
while ($L1 =mysql_fetch_row ($dbase02))
{
print ($L1 [0] . " , " ) ;
}
?>

</BODY>
</HTML>
```



مثال : استعراض حقول جدول قاعدة بيانات :

```
<HTML>
<HEAD>
<TITLE> Database Table Fields </TITLE>
</HEAD>
<BODY>

<?php
$link1 =mysql_connect ("localhost" , "root" , "") ;

// List of database Table Fields
Echo "List of database Table Fields : <BR> ";
Echo "==================== <BR>";

$dbase03 = mysql_list_fields ("products" , "price") ;
// Print Table fields
print ("<TABLE>\n") ;
print ("<TR>\n") ;
print ("<TH>Name </TH> \n") ;
print ("<TH>Type  </TH> \n") ;
print ("<TH>Length </TH> \n") ;
print ("<TH>Flags </TH> \n") ;
print ("</TR>\n") ;

// Print List of database Table fields
```

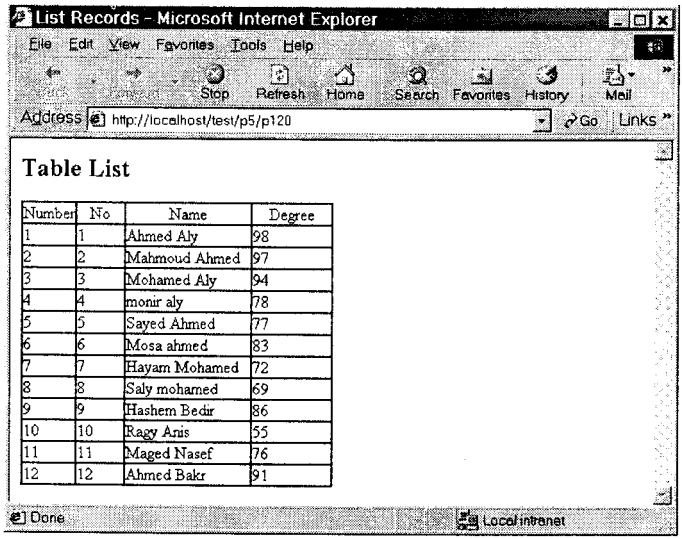
```
for ($j=0 ; $j <mysql_num_fields ($dbase03) ; $j++)
{
print ("<TR>\n") ;
print ("<TD>\n".mysql_field_name ($dbase03, $j)."</TD> \n");
print ("<TD>\n".mysql_field_type ($dbase03, $j). "</TD>\n") ;
print ("<TD>\n".mysql_field_len  ($dbase03, $j). "</TD> \n") ;
print ("<TD>\n".mysql_field_flags ($dbase03, $j). "</TD>\n") ;
print ("</TR>\n") ;
}
print ("</TABLE>\n") ;
?>
</BODY>
</HTML>
```



مثال : الاستعلام من قاعدة بيانات وتنسيق البيانات :

```
<HTML>
<TITLE> List Records </TITLE>
<BODY>
<h2> Table List </h2>
```

```
<?php
$link1 =mysql_connect ("localhost" , "root" , "") ;
$dbase04=mysql_db_query( "mydata" , "select * from table1") ;

echo "<table border=1 cellpadding =0 cellspacing = 0 style = border-
collapse: collapse  bordercolor= #111111 width=50%>" ;
echo "<td width=5% align =center > Number </td>" ;
echo "<td width=20% align = center > No </td>" ;
echo "<td width= 45% align = center > Name  </td>" ;
echo "<td width=30% align = center > Degree </td>" ;

while ($std=mysql_fetch_array ($dbase04)) {
echo "<tr>" ;
echo "<td width= 5% align = left> $std[no]  </td>" ;
echo "<td width= 20% align = left> $std[no]  </td>" ;
echo "<td width= 45% align = left> $std[name]  </td>" ;
echo "<td width= 30% align = left> $std[degree]  </td>" ;
echo "</tr>" ;
}

echo "</table>" ;
mysql_free_result ($dbase04) ;
mysql_close ($link1) ;

?>

</BODY>
</HTML>
```



دوال قاعدة بيانات MySQL

تسمح هذه الدوال بالتعامل مع قاعدة البيانات MySQL ، ويجب تشغيل قاعدة البيانات كمـــا يجب أيضا ترجمة لغة php بدعم قاعدة البيانات mysq باستخدام معامل خيار قاعدة البيانات with-mysql option-- .

الدالة	الوظيفة	الوظيفة
mysql affected rows	Get number of affected rows in previous MySQL operation	عدد الصفوف المتأثرة فى عملية سابقة .
mysql change user	Change logged in user of the active connection	تغيير مستخدم
mysql close	Close MySQL connection	إغلاق اتصال
mysql connect	Open connection to a MySQL Server	فتح اتصال

الدالة	الوظيفة	الوظيفة
mysql_create_db	Create database	إنشاء قاعدة بيانات
mysql_data_seek	Move internal result pointer	تحريك المؤشر
mysql_db_name	Get result data	اسم قاعدة بيانات
mysql_db_query	Send a MySQL query	إرسال استفسار
mysql_drop_db	Delete database	حذف قاعدة بيانات
mysql_errno	Returns numerical value of error message from previous MySQL operation	إعادة رقم رسالة خطأ
mysql_error	Returns text of the error message from previous MySQL operation	إعادة نص رسالة خطأ
mysql_fetch_array	Fetch a result row as an associative array, a numeric array, or both.	جلب مصفوفة صف
mysql_fetch_assoc	Fetch a result row as an associative array	
mysql_fetch_field	Get column information from a result and return as an object	جلب معلومات حقل
mysql_fetch_lengths	Get the length of each output in a result	جلب طول خرج
mysql_fetch_object	Fetch a result row as an object	جلب صف ككائن
mysql_fetch_row	Get a result row as an enumerated array	

الدالة	الوظيفة	الوظيفة
mysql_field_flags	Get flags associated with specified field in a result	علامات حقل
mysql_field_name	Get name of specified field in a result	اسم حقل
mysql_field_len	Returns the length of the specified field	طول حقل
mysql_field_seek	Set result pointer to a specified field offset	وضع المؤشر فى حقل
mysql_field_table	Get name of the table the specified field is in	اسم حقول جدول
mysql_field_type	Get type of specified field in a result	نوع الحقول
mysql_free_result	Free result memory	تحرير الذاكرة
mysql_insert_id	Get the id generated from the previous INSERT operation	المعـرف المولـد فى عملية سابقة
mysql_list_dbs	List databases available on a MySQL server	لائحـة قواعد البيانات المتاحة على الخادم
mysql_list_fields	List MySQL result fields	لائحة نتيجة الحقول
mysql_list_tables	List tables in a MySQL database	لائحة الجداول
mysql_num_fields	Get number of fields in result	عدد الحقول
mysql_num_rows	Get number of rows in result	عدد الصفوف
mysql_pconnect	Open a persistent connection to a MySQL Server	فتح اتصال

الدالة	الوظيفة	الوظيفة
mysql_query	Send a MySQL query	إرسال استفسار
mysql_result	Get result data	نتيجة البيانات
mysql_select_db	Select database	اختيار قاعدة بيانات
mysql_tablename	Get table name of field	اسم جدول

أمثلة :

```
<?php
// MySQL create database example
// إنشاء قاعدة بيانات
   $link = mysql_pconnect ("kron", "jutta", "geheim")
      or die ("Could not connect");
   if (mysql_create_db ("my_db")) {
      print ("Database created successfully\n");
   } else {
      printf ("Error creating database: %s\n", mysql_error ());
   }
?>
```

مثال البحث فى قاعدة بيانات :

```
<?php
// البحث فى قاعدة بيانات
// MySQL data seek example

   $link = mysql_pconnect ("localhost", "root", "")
      or die ("Could not connect");

   mysql_select_db ("samp_db")
      or die ("Could not select database");

   $query = "SELECT last_name, first_name FROM friends";
   $result = mysql_query ($query)
```

```
    or die ("Query failed");

  # fetch rows in reverse order
  for ($i = mysql_num_rows ($result) - 1; $i >=0; $i--) {
    if (!mysql_data_seek ($result, $i)) {
      printf ("Cannot seek to row %d\n", $i);
      continue;
    }

    if(!($row = mysql_fetch_object ($result)))
      continue;
    printf ("%s %s<BR>\n", $row->last_name, $row->first_name);
  }

  mysql_free_result ($result);
?>
```

مثال اسم قاعدة بيانات :

```
<?php
// Mysql_db_name() example
error_reporting(E_ALL);
mysql_connect('localhost', 'username', 'password');
$db_list = mysql_list_dbs();
$i = 0;
$cnt = mysql_num_rows($db_list);
while ($i < $cnt) {
  echo mysql_db_name($db_list, $i) . "\n";
  $i++;
}
?>
```

مثال استعراض قاعدة بيانات :

```
<?php
mysql_connect("marliesle");
echo mysql_errno().": ".mysql_error()."<BR>";
mysql_select_db("nonexistentdb");
echo mysql_errno().": ".mysql_error()."<BR>";
$conn = mysql_query("SELECT * FROM nonexistenttable");
```

```
echo mysql_errno().": ".mysql_error()."<BR>";
?>
```

مثال مصفوفة البحث :

```
<?php
//Example Mysql_fetch_array()
mysql_connect ($host, $user, $password);
$result = mysql_db_query ("database","select user_id, fullname from
table");
while ($row = mysql_fetch_array ($result)) {
   echo "user_id: ".$row["user_id"]."<br>\n";
   echo "user_id: ".$row[0]."<br>\n";
   echo "fullname: ".$row["fullname"]."<br>\n";
   echo "fullname: ".$row[1]."<br>\n";
}
mysql_free_result ($result);
?>
```

مثال بحث الحقول :

```
<?php
// Example Mysql_fetch_field()
mysql_connect ($host, $user, $password)
   or die ("Could not connect");
$result = mysql_db_query ("database", "select * from table")
   or die ("Query failed");
# get column metadata
$i = 0;
while ($i < mysql_num_fields ($result)) {
   echo "Information for column $i:<BR>\n";
   $meta = mysql_fetch_field ($result);
   if (!$meta) {
      echo "No information available<BR>\n";
   }
   echo "<PRE>
blob:        $meta->blob
max_length:  $meta->max_length
multiple_key: $meta->multiple_key
name:        $meta->name
```

```
not_null:    $meta->not_null
numeric:     $meta->numeric
primary_key: $meta->primary_key
table:       $meta->table
type:        $meta->type
unique_key:  $meta->unique_key
unsigned:    $meta->unsigned
zerofill:    $meta->zerofill
</PRE>";
   $i++;
}
mysql_free_result ($result);
?>
```

فيما يلى مثالان متكاملان :

المثال الأول

يفتح قاعدة بيانات باسم wise فيها جدول باسم wise يحتوى الجدول على ثلاثة حقول ، أولهـــم حقل نص Text يحتوى على تعليق ، والثانى حقل تاريخ الإضافة ، والثالث حقل باسم شخص قام بالإضافة .

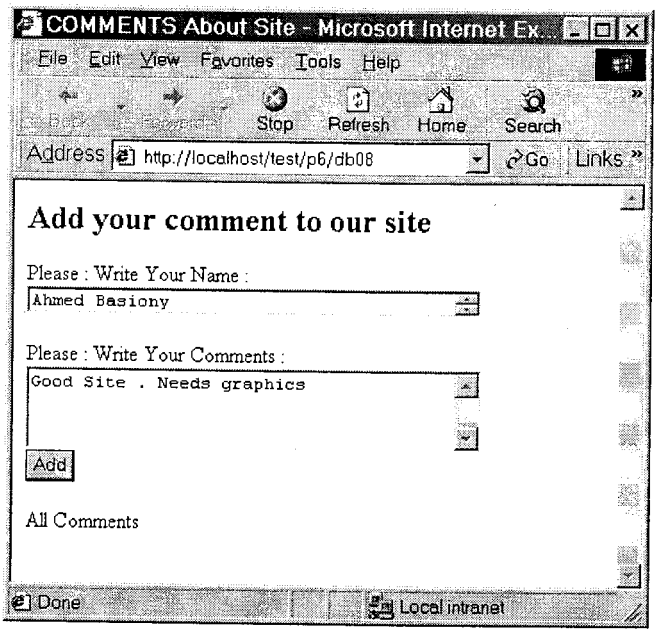
يقوم المثال بعرض نصوص قاعدة البيانات وإضافة نصوص إليها :

```
<HTML>
<BODY>
<TITLE> COMMENTS About Site </TITLE>
<h2> Add your comment to our site </h2>
<form action = "<?php echo($php_self) ; ?>"
method =post> <p> Please : Write Your Name  : <br>
<textarea name ="wisename" rows=1 cols=40 wrap>
</textarea> <br>

<p> Please : Write Your Comments :  <br>
<textarea name ="wisetext" rows=4 cols=40 wrap>
</textarea> <br>

<input type=submit name = "submitwise" value ="Add">
```

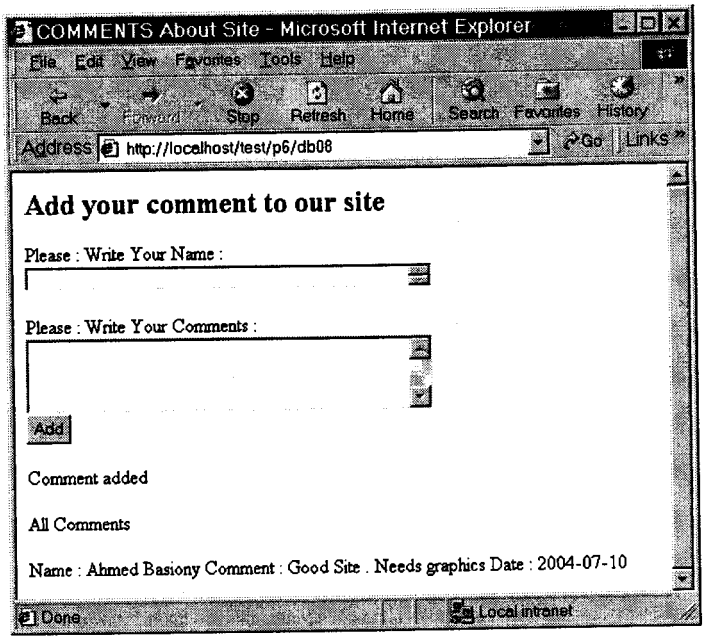
```
</form>

<?php
$dcn=@mysql_connect ("localhost" , "root" , "") ;
if (! $dcn) {
echo ("<p> Can not connect to server </p>") ;
exit() ;
}

if (! @mysql_select_db("wise"))
{
echo ("<p> Can not connect to database </p> ") ;
exit () ;
}

if ($submitwise == "Add") {
$sql= "Insert into wise set wisetext = '$wisetext' , wisename
='$wisename' ,wisedate=curdate()" ;
if (mysql_query($sql)) {
echo ("<p> Comment added  </p> ") ;
} else {
echo ("<p> Error : " . mysql_error() . "</p>") ;
}
}

echo ("<p> All Comments </p> ") ;
$result =mysql_query ("select * from wise") ;
if (! $result) {
echo ("<p> Error : " . mysql_error() . "</p>") ;
exit() ;
}

while ($row=mysql_fetch_array($result)) {
echo ("Name : " .$row [wisename]) ;
echo ("  Comment : " .$row [wisetext]) ;
echo ("  Date : " .$row [wisedate]. "<BR>") ;
}
```

```
?>

</BODY>
</HTML>
```

عند تشغيل البرنامج لأول مرة فسوف تكون قاعدة البيانات خالية :



عند كتابة تعليق وإضافته تظهر النتيجة على الشكل :



المثال الثانى

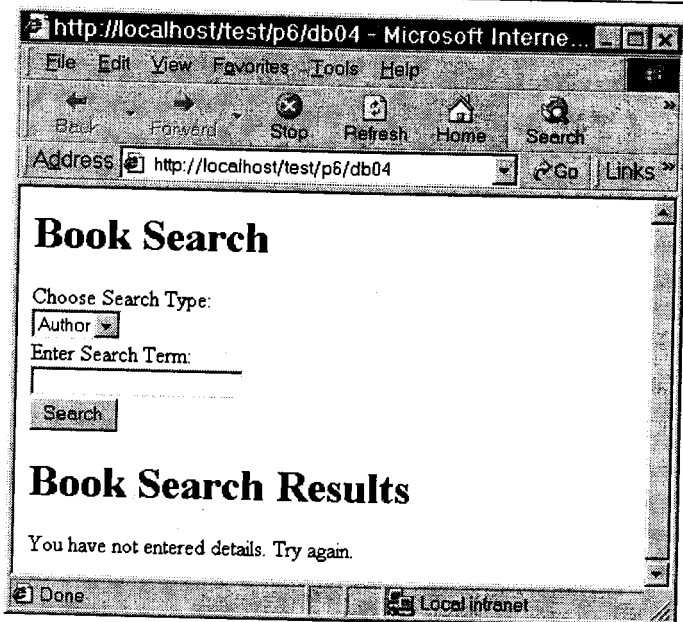
يفتح قاعدة بيانات الكتب Books التى تحتوى على جدول الكتب Books يحتوى الجدول بدوره على حقل رقم الإيداع ISBN وحقل المؤلف Author وحقل عنوان الكتاب Title وحقل سعر الكتاب Price ، ويسمح المثال بالبحث عن الكتب بدلالة الحقول فى قاعدة البيانات :

```
<HTML>
<BODY>
<h1> Book Search </h1>
  <form action="" method="post">
    Choose Search Type:<br>
    <select name="searchtype">
      <option value="author">Author
```

```
      <option value="title">Title
      <option value="isbn">ISBN
    </select>
    <br>
    Enter Search Term:<br>
    <input name="searchterm" type=text>
    <br>
    <input type=submit value="Search">
  </form>
<h1> Book Search Results </h1>
<?
  if (!$searchtype || !$searchterm)
  {
    echo "You have not entered details.  Try again.";
    exit;
  }
  $searchtype = addslashes($searchtype);
  $searchterm = addslashes($searchterm);
  @ $db = mysql_pconnect("localhost", "root", "");
  if (!$db)
  {
echo "Error: Not connected . Try later.";
    exit;
  }
  mysql_select_db("books");
  $query = "select * from books where ".$searchtype." like
'%".$searchterm."%'";
  $result = mysql_query($query);
```

```
$num_results = mysql_num_rows($result);
echo "<p>Number of books found: ".$num_results."</p>";
for ($i=0; $i <$num_results; $i++)
{
   $row = mysql_fetch_array($result);
   echo "<p><strong>".($i+1).". Title: ";
   echo stripslashes($row["title"]);
   echo "</strong><br>Author: ";
   echo stripslashes($row["author"]);
   echo "<br>ISBN: ";
   echo stripslashes($row["isbn"]);
   echo "<br>Price: ";
   echo stripslashes($row["price"]);
   echo "</p>";
 }
?>

</BODY>
</HTML>
```

النتيجة عند تشغيل البرنامج ستكون :



وعند اختيار نوع البحث مثل البحث عن طريق المؤلف Author أو العنوان Title أو رقم الإيداع ISBN ، وإدخال عنصر البحث فى مربع إدخال عنصر البحث ، ثم نقر زر البحث Search تظهر النتائج فى الجزء الأسفل من الشاشة .

الفصل السادس 6

متنوعات

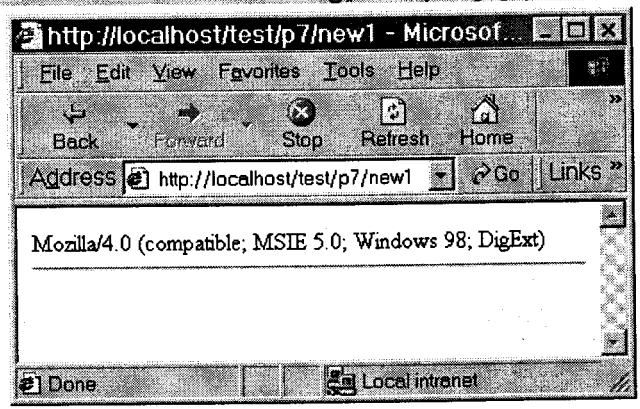
تستخدم دالة معرفة المستعرض بالصيغة :

get_browser ([string user_agent])

يبين المثال التالى كيفية تنفيذ هذا :

```
<?php
// Example Get_browser()
echo "$HTTP_USER_AGENT<hr>\n";
$browser = get_browser();
?>
```

النتيجة : المستعرض ونظام التشغيل :

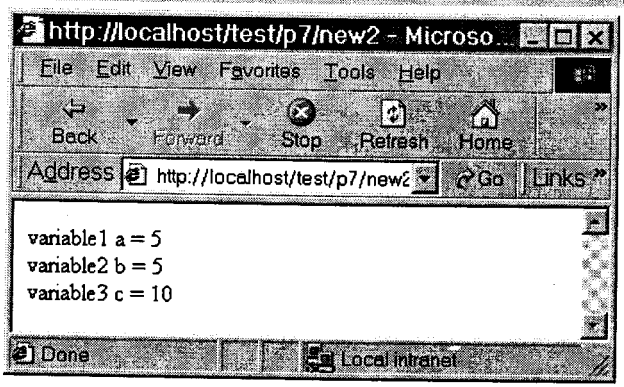


المرجعية Reference تعنى استدعاء نفس محتويات المتغير بأسماء أخرى مثل :

```
<?php
```

```
$b=5 ;
$a =& $b ;
$c=9 ;
// $a and $b point to the same variable.
function foo (&$var) {
   $var++;
}
foo ($c);
// will make $c to be 10
echo "variable1 a = ".$a ;
echo "<BR>variable2 b = ".$b ;
echo "<BR>variable3 c = ".$c;
?>
```

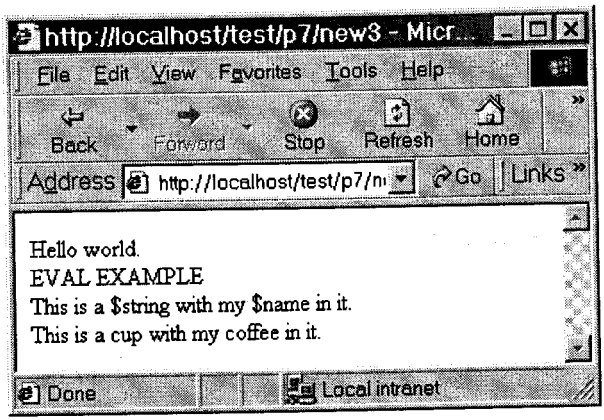
النتيجة :



لتعريف الثوابت والحصول على ربط نص :

```
<?php
// Example Defining Constants
define ("CONSTANT", "Hello world.");
echo CONSTANT; // outputs "Hello world."
Echo "<BR> EVAL EXAMPLE <BR>" ;
// Example 1. Eval() example - simple text merge
$string = 'cup';
$name = 'coffee';
$str = 'This is a $string with my $name in it.<br>';
echo $str;
```

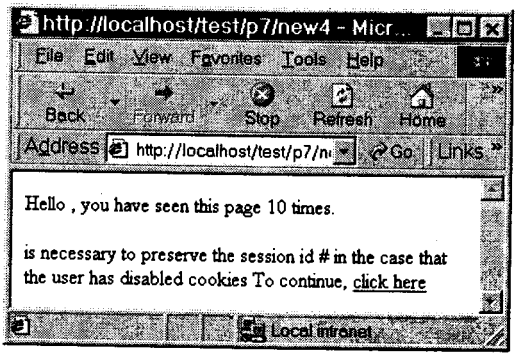
```
eval ("\$str = \"$str\";");
echo $str;
// The above example will show:
// This is a $string with my $name in it.
// This is a cup with my coffee in it.
?>
```



المثال التالى قد يكون فى بداية الصفحة لتعريف الزائر بعدد مرات دخوله الصفحة :

```
<?php
// Example Counting the number of hits of a single user
session_register ("count");
$count++;
echo("Hello , you have seen this page ".$count ." times.<p>") ;
# the <?=SID?> is necessary to preserve the session id
# in the case that the user has disabled cookies
To continue, <A HREF="nextpage.php?<?=SID?>">click here</A>
```

نتيجة تنفيذ المثال تختلف باختلاف عدد مرات الزيارة :



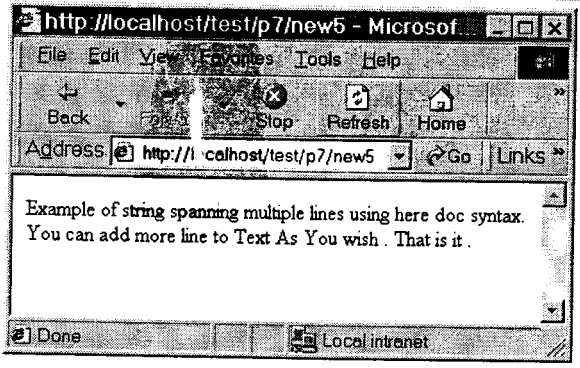
سلاسل النصوص ودوال السلاسل النصية

فى الأمثلة التالية عدة نماذج لأمثلة النصوص :

مثال تجاوز النص عدة سطور وإحاطة النص :

```
<?php
// Example doc string quoting example
$str = <<<EOD
Example of string
spanning multiple lines
using here doc syntax. You can add more line to Text
As You wish . That is it .
EOD;
Echo $str ;
?>
```

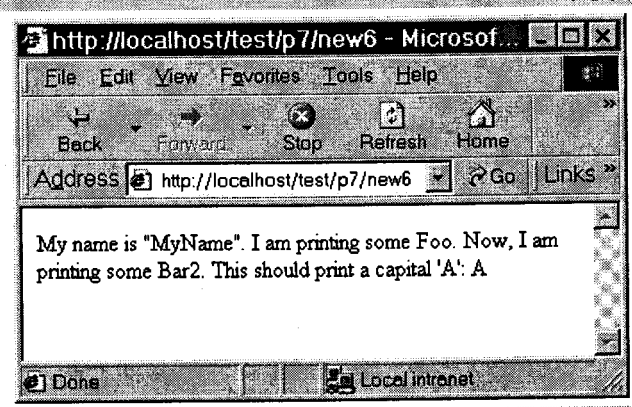
نتيجة المثال :



مثال آخر :

```
<?php
// مثال آخر
/* More complex example, with variables. */
class foo {
   var $foo;
   var $bar;
   function foo() {
      $this->foo = 'Foo';
      $this->bar = array('Bar1', 'Bar2', 'Bar3');
   }
}
$foo = new foo();
$name = 'MyName';
echo <<<EOT
My name is "$name". I am printing some $foo->foo.
Now, I am printing some {$foo->bar[1]}.
This should print a capital 'A': \x41
EOT;
?>
```

نتيجة المثال :

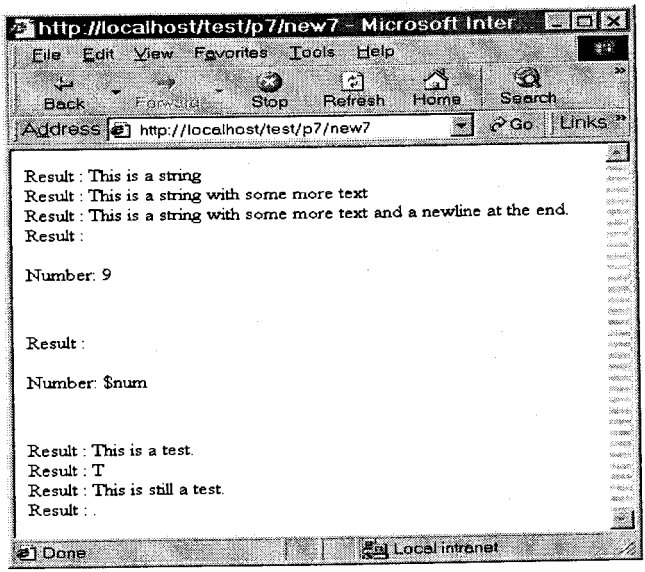


مثال لوصل النصوص :

```
<?php
```

```
// يمكن وصل النص بعلامة النقطة '.' (dot)

// لا تستخدم علامة الجمع + .
/* Assigning a string. */
$str = "This is a string";
echo " Result : " . $str. "<BR>" ;

/* Appending to it. */
$str = $str . " with some more text";
echo " Result : " . $str. "<BR>" ;

/* Another way to append, includes an escaped newline. */
$str .= " and a newline at the end.\n";
echo " Result : " . $str. "<BR>" ;

/* This string will end up being '<p>Number: 9</p>' */
$num = 9;
$str = "<p>Number: $num</p>";
echo " Result : " . $str. "<BR>" ;

/* This one will be '<p>Number: $num</p>' */
$num = 9;
$str = '<p>Number: $num</p>';
echo " Result : " . $str. "<BR>" ;
/* Get the first character of a string  */
$str = 'This is a test.';
$first = $str[0];
echo " Result : " . $str. "<BR>" ;
echo " Result : " . $first. "<BR>" ;

/* Get the last character of a string. */
$str = 'This is still a test.';
$last = $str[strlen($str)-1];
echo " Result : " . $str. "<BR>" ;
echo " Result : " . $last. "<BR>" ;

?>
```

نتيجة المثال :



مثال تحويلات السلاسل النصية :

```
<?php

// String conversion

// تحويلات السلاسل النصية
$foo = 1 + "10.5";                // $foo is double (11.5)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
$foo = 1 + "-1.3e3";              // $foo is double (-1299)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
$foo = 1 + "bob-1.3e3";           // $foo is integer (1)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
$foo = 1 + "bob3";                // $foo is integer (1)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
$foo = 1 + "10 Small Pigs";       // $foo is integer (11)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
$foo = 1 + "10 Little Piggies"; // $foo is integer (11)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
$foo = "10.0 pigs " + 1;          // $foo is integer (11)
```

```
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
$foo = "10.0 pigs " + 1.0;      // $foo is double (11)
echo "\$foo==$foo; type is " . gettype ($foo) . "<br>\n";
?>
```

دوال سلاسل النصوص :

الدالة	الوظيفة	الوظيفة
AddCSlashes	Quote string with slashes in C style	إحاطة النص بعلامات
AddSlashes	Quote string with slashes	إحاطة النص بعلامات
bin2hex	Convert binary data into hexadecimal	تحويل ثنائى إلى سداسى عشرى
Chop	Remove trailing white space	حذف المسافات الأخيرة
Chr	Return character	حرف
chunk_split	Split a string into smaller chunks	تقسيم نص إلى أجزاء
convert_cyr_string	Convert from one Cyrillic character set to another	تحويل من تنسيق لآخر
count_chars	Return information about string characters	معلومات حروف نص
crypt	DES encrypt a string	تشفير نص
echo	Output strings	إخراج نص
explode	Split a string by string	تقسيم نص بنص
get_html_translation_table	Returns translation table used by **htmlspecialchars()** and **htmlentities()**	جدول ترجمة لغة النص المتشعب
get_meta_tags	Extracts meta tag content attributes from file and returns array	فصل وسوم
htmlentities	Convert all applicable characters to HTML entities	تحويل رموز مدخلات نص متشعب

الدالة	الوظيفة	الوظيفة
htmlspecialchars	Convert special characters to HTML entities	تحويل رموز إلى مدخلات نص متشعب
join	Join array elements with a string	ربط عناصر مصفوفة مع نص
levenshtein	Calculate Levenshtein distance between two strings	حساب مسافة بين نصين
ltrim	Strip white space from beginning of string	فصل مسافات بداية نص
nl2br	Converts new lines to HTML line breaks	تحويل سطور جديدة إلى فواصل سطور نص متشعب
Ord	Return ASCII value of character	قيمة آسكى لحرف
parse_str	Parses the string into variables	إعراب نص لمتغير
print	Output a string	إخراج نص
printf	Output formatted string	إخراج نص بصيغة
rtrim	Remove trailing white space	تشذيب نص بحذف نهاياته
sscanf	Parses input from a string according to a format	إعراب دخل من نص بناء على صيغة
similar_text	Calculate the similarity between two strings	حساب التشابه بين نصين
sprintf	Return formatted string	إرجاع نص منسق
strchr	Find first occurrence of a character	إيجاد أول حرف
strcmp	Binary safe string comparison	مقارنة ثنائية
strcspn	Find length of initial segment not matching mask	طول مقطع

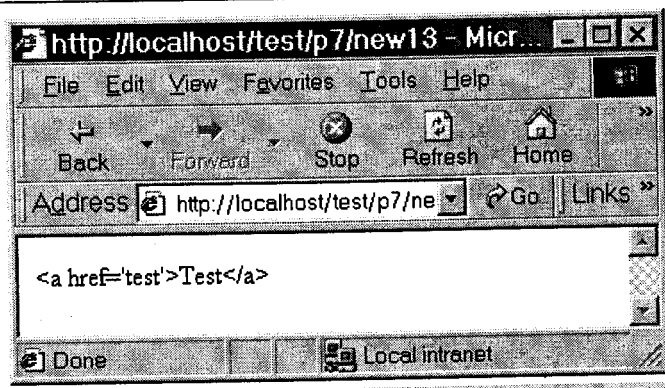
الدالة	الوظيفة	الوظيفة
strip_tags	Strip HTML and PHP tags from a string	فصل علامات لغات نص متشعب ولغة PHP من نص
stripcslashes	Un quote string quoted with **addcslashes()**	إلغاء علامات حول نص
stripslashes	Un quote string quoted with **addslashes()**	إلغاء علامات حول نص
stristr	Case insensitive **strstr()**	حساسية حالة الأحرف
strlen	Get string length	طول نص
strnatcmp	String comparisons using a "natural order" algorithm	مقارنة نص
strnatcasecmp	Case insensitive string comparisons using a "natural order" algorithm	مقارنة نص بحالة حساسية الأحرف
strncmp	Binary safe string comparison of the first n characters	مقارنة ثنائية
str_pad	Pad a string to a certain length with another string	حشو نص لطول معين بنص آخر
strpos	Find position of first occurrence of a string	إيجاد موقع أول حدوث فى نص
strrchr	Find last occurrence of a character in a string	إيجاد آخر حدوث فى نص
str_repeat	Repeat a string	تكرار نص
strrev	Reverse a string	عكس نص
strrpos	Find position of last occurrence of a char in a string	إيجاد موقع آخر حدوث لحرف فى نص
strspn	Find length of initial segment matching mask	الطول الابتدائى لمقطع

الدالة	الوظيفة	الوظيفة
		مطابق
strstr	Find first occurrence of a string	إيجاد أول حدوث لنص
strtolower	Make string lowercase	تحويل نص إلى حروف كبيرة
strtoupper	Make string uppercase	تحويل نص لحروف صغيرة
Str_replace	Replace all occurrences of needle in haystack with str	استبدال
strtr	Translate certain characters	ترجمة حروف معينة
substr	Return part of a string	جزء نص
substr_count	Count the number of substring occurrences	عدد حدوث تركيبة نص
Substr_replace	Replace text within a portion of a string	استبدال نص داخل جزء
trim	Strip white space from beginning and end of string	تشذيب نص من بدايته ونهايته
ucfirst	Make a string's first character uppercase	عمل أول حرف كبير
ucwords	Uppercase first character of each word in string	أول حرف فى كل كلمة كبير
wordwrap	Wraps string to given number of characters using string break character.	لف نص

فيما يلى أمثلة لاستخدام هذه الدوال :

```
// Example Addcslashes()
$escaped = addcslashes ($not_escaped, "\0..\37!@\177..\377");

// Example Chop()
$line= " Good Morning : " ;
```

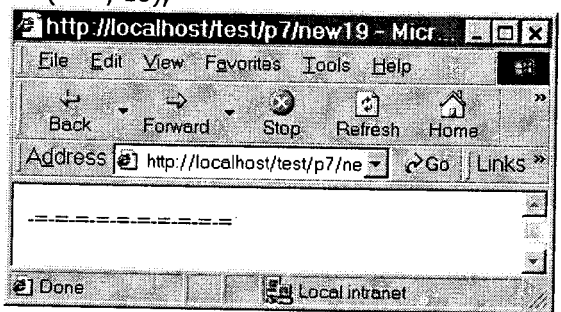
```
$trimmed = chop ($line);
echo $trimmed ;

// Chr() example
$str .= chr (27); /* add an escape character at the end of $str */
/* Often this is more useful */
$str = sprintf ("The string ends in escape: %c", 27);
echo $str

// Chunk_split() example
# format $data using RFC 2045 semantics
$new_string = chunk_split (base64_encode($data));

echo "Hello World";
echo "This spans
multiple lines. The newlines will be
output as well";
echo "This spans\nmultiple lines. The newlines will be\noutput as well.";

// Explode() example
$pizza = "piece1 piece2 piece3 piece4 piece5 piece6";
$pieces = explode (" ", $pizza);
// تقسيم النص بعلامة المسافة ، وبالتالى يصبح مصفوفة
echo $pizza ;
echo "<BR>" . $pieces ;

// htmlspecialchars() example
// الحصول على نص متشعب بعلاماته
$new = htmlspecialchars("<a href='test'>Test</a>", ENT_QUOTES);
echo $new ;
```



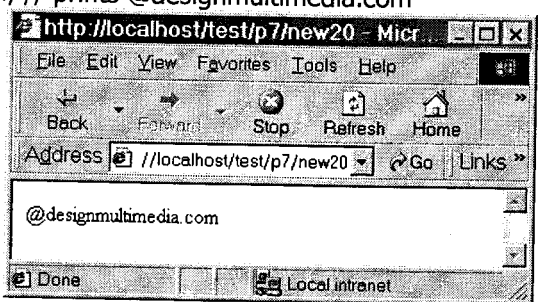
مثال صياغة المعروض على الشاشة :

```
//Example Sprintf() formatting currency
$money1 = 68.75;
$money2 = 54.35;
$money = $money1 + $money2;
echo $money ;
// will output "123.1"
$formatted = sprintf ("%01.2f", $money);
echo $formatted ;
// will output "123.10"

// Example strcasecmp()
$var1 = "Hello";
$var2 = "hello";
if (!strcasecmp ($var1, $var2)) {
 echo '$var1 is equal to $var2 in a case-insensitive string comparison';
}

$arr1 = $arr2 = array
("img12.png","img10.png","img2.png","img1.png");
echo "Standard string comparison\n";
usort($arr1,"strcmp");
print_r($arr1);
echo "\nNatural order string comparison\n";
usort($arr2,"strnatcmp");
```

```
print_r($arr2);

// Example str_pad()
$input = "Alien";
print str_pad($input, 10);                      // produces "Alien     "
print str_pad($input, 10, "-=", STR_PAD_LEFT);  // produces "-=-=-
Alien"
print str_pad($input, 10, "_", STR_PAD_BOTH);   // produces
"__Alien___"


// Str_repeat() example
echo str_repeat ("-=", 10);
```

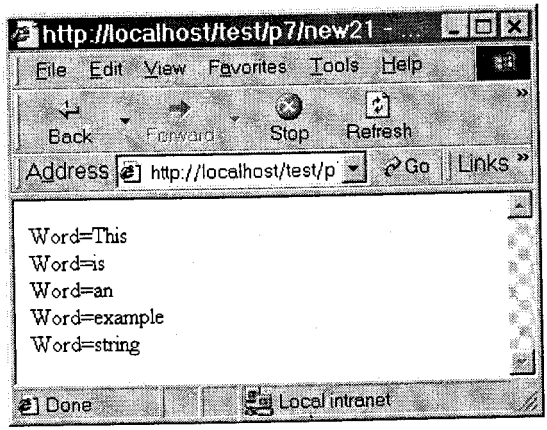



```
// Strstr() example
$email = 'sterling@designmultimedia.com';
$domain = strstr ($email, '@');
print $domain; // prints @designmultimedia.com
```

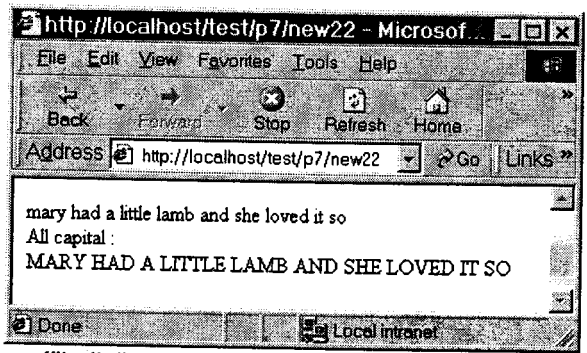



```
// Strtok() example
$string = "This is an example string";
```

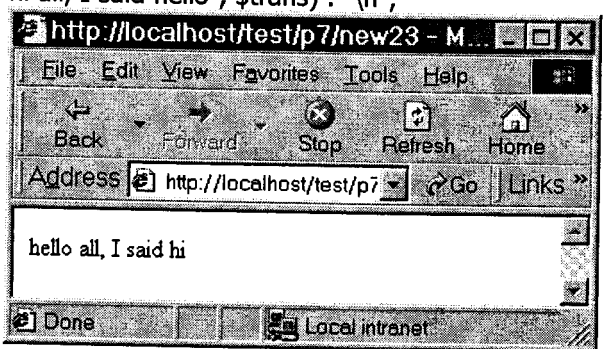
```
$tok = strtok ($string," ");
while ($tok) {
   echo "Word=$tok<br>";
   $tok = strtok (" ");
}
```




```
// تحويل النص إلى حروف صغيرة
// Strtolower() example
$str = "Mary Had A Little Lamb and She LOVED It So";
$str1 = strtolower($str);
print $str1; # Prints mary had a little lamb and she loved it so
// تحويل النص إلى حروف كبيرة
echo "<BR>All capital :<BR>" ;
// Strtoupper() example
$str2 = strtoupper ($str);
print $str2; # Prints MARY HAD A LITTLE LAMB AND SHE LOVED IT SO
```

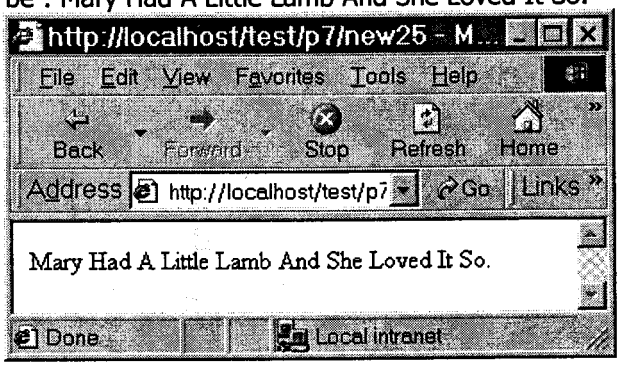



```
$trans = array ("hello" => "hi", "hi" => "hello");
echo strtr("hi all, I said hello", $trans) . "\n";
```

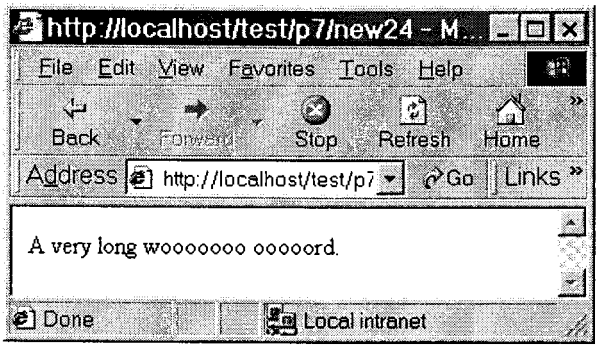



```
// string substr (string string, int start [, int length])
$rest = substr ("abcdef", 1);    // returns "bcdef"
$rest = substr ("abcdef", 1, 3); // returns "bcd"
$rest = substr ("abcdef", -1);   // returns "f"
$rest = substr ("abcdef", -2);   // returns "ef"
$rest = substr ("abcdef", -3, 1); // returns "d"
$rest = substr ("abcdef", 1, -1); // returns "bcde"

// Example substr_count() example
print substr_count("This is a test", "is"); // prints out 2

//Substr_replace() example
$var = 'ABCDEFGH:/MNRPQR/';
echo "Original: $var<hr>\n";
```

```
/* These two examples replace all of $var with 'bob'. */
echo substr_replace ($var, 'bob', 0) . "<br>\n";
echo substr_replace ($var, 'bob', 0, strlen ($var)) . "<br>\n";

/* Insert 'bob' right at the beginning of $var. */
echo substr_replace ($var, 'bob', 0, 0) . "<br>\n";

/* These next two replace 'MNRPQR' in $var with 'bob'. */
echo substr_replace ($var, 'bob', 10, -1) . "<br>\n";
echo substr_replace ($var, 'bob', -7, -1) . "<br>\n";

// Example Ucfirst()
$text = 'mary had a little lamb and she loved it so.';
$text = ucfirst ($text); // $text is now Mary had a little lamb
                         // and she loved it so.

// ucwords() example
$text = "mary had a little lamb and she loved it so.";
$text = ucwords($text);
echo $text;
//Text will be : Mary Had A Little Lamb And She Loved It So.
```




```
// Example wordwrap()
$text = "A very long woooooooooooord.";
$newtext = wordwrap( $text, 8, "\n", 1);
echo "$newtext\n";
```



الكعك Cookies

الكعك ملف صغير يتم زرعه فى جهاز المستخدم ، ويمكن زراعة ملفات لعمل مهام متعددة .

تقوم دالة وضع كعكة setcookie بإرسال كعكة وتكون على الصيغة :

int setcookie (string name [, string value [, int expire [, string path [, string domain [, int secure]]]]])

تقوم بتعريف كعكة Setcookie() لإرسالها مع معلومات تصدير ، ويجب إرسال الكعكة قبل إرسال تصدير آخر وبالتالى يجب وضع الدالة قبل أى رموز نص متشعب أو تصدير <html> or <head> tags .

كل معاملات إنشاء الكعكة اختيارية ماعدا معامل الاسم name وإذا تواجد معامل الاسم فقط فإن الكعكة بهذا الاسم سوف تحذف من الزبون ، ويجب أيضا استبدال المعامل الاختيارى برموز نص فارغ "" لتجاوز هذا المعامل ، معامل توقيت الانتهاء ومعامل المصدر expire and secure arguments أعداد لا يمكن تجاوزها بنص فارغ ، ويوضع الصفر بديلا عنها (0) zero .

معامل توقيت الانتهاء هو دالة من دوال توقيت نظام يونكس التى تتعامل مع دوال الوقت time() or mktime() ، بينما المصدر secure يشير إلى أن الكعكة سوف

ترسل فقط عبر اتصال بروتوكول آمن لنقل النص المتشعب Secure HTTPS connection .

لن تكون الكعكة مرئية حتى التحميل التالى للصفحة التى تظهر لها الكعكة . يجب حذف الكعكة بنفس المعامل الذى يقوم بإعدادها .

يمكن استدعاء عدد من الكعك فى نفس النص البرمجى ، ولحذف كعكة قبل إدراج أخرى يجب وضع نص الإدراج قبل الحذف . أمثلة :

```
// Setcookie() send examples
setcookie ("TestCookie", "Test Value");
setcookie ("TestCookie", $value, time()+3600);  /* expire in 1 hour */
setcookie ("TestCookie", $value, time()+3600, "/prog/", ".net1.eg", 1);

//Examples follow how to delete cookies send in previous example :
// setcookie() Delete examples
setcookie ("TestCookie");
// set Expiration date to one hour ago
setcookie ("TestCookie", "", time() - 3600);
setcookie ("TestCookie", "", time() - 3600, "/prog/", ".net1.eg", 1);
```

عند حذف كعكة يجب التأكد من أن تاريخ انتهاء عملها expiration date فى الماضى.

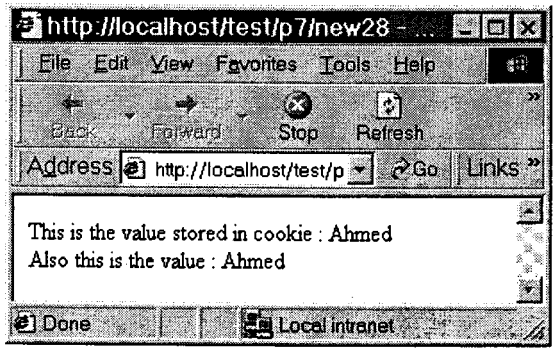
لمعاينة محتويات كعكة يستخدم ما يشبه المثال التالى :

```
echo $TestCookie;
echo $HTTP_COOKIE_VARS["TestCookie"];
```

مثال :

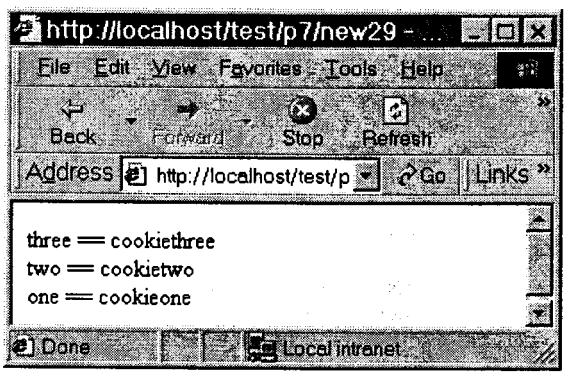
```
<?php
// Setcookie() send examples
$value= "Ahmed" ;
setcookie ("TestCookie", $value, time()+3600);  /* expire in 1 hour */

echo ("This is the value stored in cookie : " .$TestCookie."<BR>") ;
echo "Also this is the value : ". $HTTP_COOKIE_VARS["TestCookie"];
// setcookie() Delete
```

```
?>
```



يمكـــن أيضا إعداد مصفوفة كعك باستخدام اسم مصفوفة ، لكن عند استقبال عناصر الكعك توضع فى مصفوفة باسم الكعكة array with cookie's name :

```
// لاحظ ترتيب الإنشاء
setcookie ("cookie[three]", "cookiethree");
setcookie ("cookie[two]", "cookietwo");
setcookie ("cookie[one]", "cookieone");
if (isset ($cookie)) {
   while (list ($name, $value) = each ($cookie)) {
      echo "$name == $value<br>\n";
   }
}
```



لزرع الكعكة فى الغالب نستخدم الصيغة المبسطة :

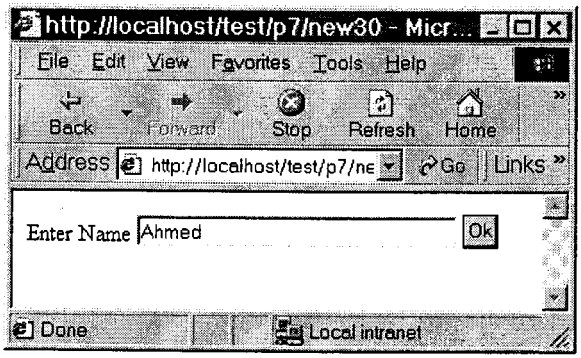
setcookie ("variable name " , "variable value" , "expire time") ;

أى يتضمن : اسم المتغير ، وقيمة المتغير ، وتوقيت انتهاء الكعكة .

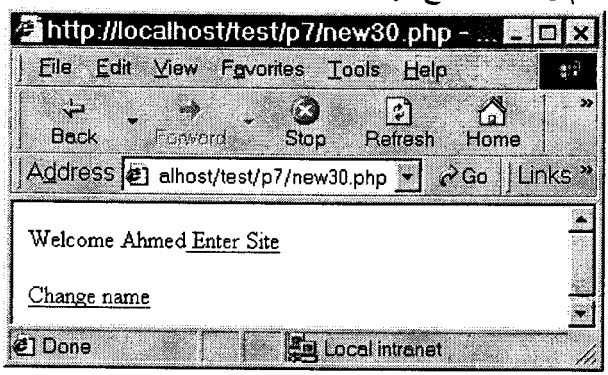
مثال : استقبال زائر وتسجيل اسمه :

```
<?php
// التأكد من وجود قيمة وإلا سيتم إدخالها
If ($value1) {
$expdate = time ()+(3600) ;
// تعود من الإدخال $value1 وبقيمة Testوضع الكعكة باسم
setcookie ("Test" , $value1 , "$expdate") ;
print "Welcome  " .$Test ;
echo '<a href ="new31.php"> Enter Site </a> ' ;
echo '<p> <a href ="new30.php"> Change name </a> </p>' ;
}
else {
// نموذج إدخال اسم الزائر
echo "<html>" ;
echo "<form method =POST action =new30.php>" ;
echo "<span> Enter Name </span> " ;
echo "<input type =text name =value1 size=30> <input type=submit
value= Ok =B1> </p>" ;
echo "</form>";
}
?>
```

عند تشغيل هذا الملف يتم زرع كعكة فى الجهاز فى دليل كعك Cookies بدليل ويندوز Windows ، وسوف يظهر :



عند إدخال الاسم وضغط مفتاح الإدخال أو نقر زر موافق Ok تنتقل للشاشة التالية :

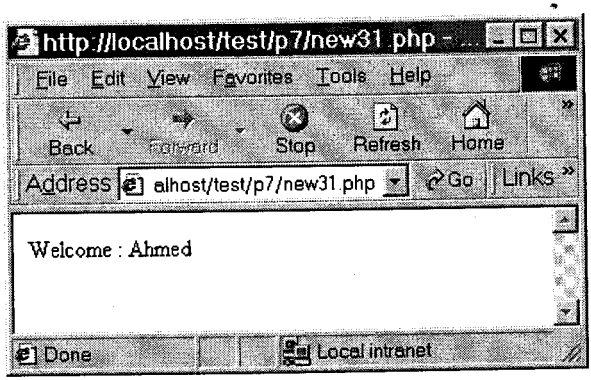


عند نقر زر تغيير الاسم Change Name تعود للصفحة السابقة .

عـند نقـر زر دخول الموقع Enter Site ينتقل العمل للبرنامج التالى الذى يجب أن يكون فى نفس المجلد : ليقوم بعرض الاسم المخزن فى ملف الكعك :

```
<?php
// التأكد من وجود الكعكة باسمها
If ($Test)
{
echo "<html>" ;
// طباعة عبارة الترحيب وبعدها المتغير المخزن بقيمة ما بالكعكة
Print "Welcome : " .$Test ;
```

```
} else {
print " Did not sign" ;
echo ' <p> <a herf ="new30.php" Click to Sign </a> </p> ' ;
}
?>
```



ويمكن طبعا وضع ملف متكامل للعمل .

الفئات والكائنات Classes and Objects

الفئة class عبارة عن تجميع من المتغيرات والدوال التى تعمل مع هذه المتغيرات ، وتعرف الفئة باستخدام الصيغة التالية :

```
class ClassName {
   variables

   function {
      Objects
   }
}
```

مثال لفئة عربة تسوق cart :

```
<?php
class Cart {
   var $items;  // Items in our shopping cart

   // Add $num articles of $artnr to the cart
```

```
    function add_item ($artnr, $num) {
      $this->items[$artnr] += $num;
    }

    // Take $num articles of $artnr out of the cart
    function remove_item ($artnr, $num) {
      if ($this->items[$artnr] > $num) {
        $this->items[$artnr] -= $num;
        return true;
      } else {
        return false;
      }
    }
}
?>
```

هذا التعريف يعرف فئة باسم Cart تتكون من مصفوفة عناصر ودالتين لجمع وحذف العناصر من عربة التسوق . مثال للنوع :

```
$cart = new Cart;
$cart->add_item("10", 1);
```

هـذا المـثال يقوم بإنشاء كائن $cart من فئة Cart ، دالة add_item() لهذا الكائن تستدعى لزيادة عنصر واحد 1 item للمقطع article number 10 للعربة .

يمكـن أن تكـون الفئات امتدادا لفئات أخرى ، وتكون للفئات الممتدة نفس متغيرات ودوال الفئة الأساسية ويتم ذلك عن طريق الامتداد مثال :

```
class Named_Cart extends Cart {
  var $owner;

  function set_owner ($name) {
    $this->owner = $name;
  }
}
```

هـذا يقـوم بتعريف فئة باسم Named_Cart لها كل متغيرات ودوال الدالة الأصلية Cart إضافة إلى المتغير $owner مع دالة إضافية باسم set_owner() .

يمكن إنشاء عربة باسم مالك أو استخدام دوال العربة فى هذه الفئة :

```
$ncart = new Named_Cart;    // Create a named cart
$ncart->set_owner ("kris"); // Name that cart
print $ncart->owner;        // print the cart owners name
$ncart->add_item ("10", 1); // (inherited functionality from cart)
```

فـــى داخل دوال فئة فإن المتغير $this يعنى كائنا object ، ويجب استخدام $this->something للوصول إلى أى متغير أو دالة .

التركيبات Constructors هى دوال يتم استدعاؤها تلقائيا automatically called عند إنشـــاء جديـــد new instance لفئة ، وتصبح الدالة تركيبا عندما يكون لها نفس اسم الفئة :

```
class Auto_Cart extends Cart {
   function Auto_Cart () {
      $this->add_item ("10", 1);
   }
}
```

إن هـــذا يقوم بتعريف فئة Auto_Cart وهى فئة Cart إضافة إلى تركيب يقوم بإعداد العـــربة بعنصـــر واحد من العنصر article number "10" ، فى كل مرة يتم عمل Auto_Cart بجديـــد "new" ، التركيبات يمكن أيضا أن تأخذ معاملات ، ويمكن لهذه المعاملات أيضا أن تكون اختيارية .

```
class Constructor_Cart extends Cart {
   function Constructor_Cart ($item = "10", $num = 1) {
      $this->add_item ($item, $num);
   }
}
// Shop the same old boring stuff.
$default_cart  = new Constructor_Cart;

// Shop for real...
$different_cart = new Constructor_Cart ("20", 17);
```

بالنسبة للفئات المدفوعة Derived classes فإن التركيب الخاص بالفئة الأصلية لا يتم استدعاؤه تلقائيا عند طلب تركيبة فئة مدفوعة .

فائدة الفئات هو الاستخدام المتكرر للفئة والمتغيرات .

المحتويات

مقدمة ٥

الفصل الأول

تمهيد وتركيب اللغة ٧

الفصل الثاني

أساسيات اللغة ٣٧

الفصل الثالث

التحكم (الشروط والتكرار) ٦١

الفصل الرابع

الدوال Functions ٧٩

الفصل الخامس

قواعد البيانات ١٥١

الفصل السادس

مزيد من معلومات الكائنات ١٩٧

لمست

السـلام الذهبيـة للطباعـة
4 ش الحرية من عبد الغفار عزيز – الملأة – دار السلام
ت 3184164