جامعة حماة
كلية الاقتصاد
التعليم المفتوح
برنامج التسويق و التجارة الالكترونية

أساسيات الانترنت

السنة الثالثة

مقدمة :

ليس صحيحاً ما يعتقده أغلب الناس أنّ الوب حديث العهد وأنـه ولـد بـين ليلـة وضحاهـا، فجـذور الوب تعود إلى أكثر من 25 سنة خلت. لكن الصحيح هو أن انتشار الوب على هـذا النحـو الهـائل لم يكن تدريجياً فقد فاق في تضخمه خلال فترة قصيرة جميع التقنيات الأخرى.

قبل أن نبدأ ركوب أمواج الإنترنت، يجب أن نتسلح بمعرفة بعض المفاهيم الأساسية الـتي مـن المـهم جداً أن نفهم معناها ونستوعبها جيداً لكي نستطيع فهم طريقة عمل الإنترنت. سنشرح هذه المفـاهيم على نحوٍ بسيط ولكنه يكفي لفهم محتواها ودورها في تشكيل الإنترنت دون أن ندخـل في تفصيلاتـها التقنية.

خدمة الويب

إنّ خدمة الوب أو ما يشار إليه باسم World Wide Web (أو اختصـاراً WWW أو W3) هـي نظـام معلومات للوسائط المنهلة Hypermedia على شبكة إنترنت، وهو نظام مبـني علـى تقنيـة النصـوص المنهلة (hypertext) مع توسيعها بحيث تعمل على شبكتي إنترنت وإنترانت.

يُعرّف الوب رسمياً على أنه: "مبادرة لاستخراج معلومات الوسائط المنهلة البعيدة تهدف إلى الوصول إلى عالم ضخم من الوثائق".

يسمح الوب للمستثمرين بالتفاعل فيما بينهم من خلال وثائق مخزنة على حواسيب مرتبطة مع شبكة إنترنت كما لو كانت أجزاء من نص واحد. وهو يعتبر طريقة للتشارك في المعلومات بين أناس عديدين في نفس الوقت حتى لو كانت هذه المعلومات أو المصادر موجودة في أطراف العـالم. أصبـح الـوب الآن متاحاً للعديد من الناس ولم يعد استخدامه يقتصر على رجال العلم والاختصاصيين في مجال إنترنت، بل أصبح وسيلة اتصال بين شريحة كبيرة من المستثمرين.

بروتوكولات التخاطب

البروتوكول هو توصيف للغة التخاطب وكيفية تبادل المعلومات بين جهازين أو تطبيقين برمجيين. ويشمل تعريف البروتوكول توصيف المعلومات التي يمكن تبادلها وتوصيف كل ما يلزم لكي يتم تبادل تلك المعلومات على نحوٍ يحقق الغرض من ذلك التبادل؛ بروتوكول الإنترنت الشهير IP (Internet Protocol) مثلاً يوصف طريقة تبادل رزمة من المعلومات بين جهازين متصلين عن طريق شبكة الإنترنت. يشمل هذا التوصيف طريقة عنونة الأجهزة على الشبكة العالمية بحيث يمكن التخاطب مع جهاز ما بمعرفة عنوانه "IP Address". كما يشمل التوصيف شكل الرزمة "Packet" التي يجري تبادلها مع تحديد دقيق لدلالة مكونات هذه الرزمة.

برتوكول Telnet

يُوصَّف هذا البروتوكول طريقة الدخول عن بعد "Remote login" حيث يسمح لمستخدم على حاسوب (زبون) بأن يتصل مع حاسوب بعيد (مخدم)، ويتصرف وكأنه يعمل مباشرة على هذا الأخير؛ أي يمكنه تنفيذ التعليمات والأوامر على الحاسوب البعيد بإدخالها عن طريق الحاسوب الزبون تماماً كما لو كان يعمل مباشرة على الحاسوب البعيد.

برتوكول نقل الملفات FTP

يسمح هذا البروتوكول (File Transfer Protocol <FTP>) بنقل ملفات من حاسوب إلى اخر. لإجراء ذلك، يقوم الزبون "ftp client" بفتح الاتصال مع الحاسوب المخدم مرسلاً إليه اسم الحساب وكلمة المرور اللازمين لتحديد ما هو مسموح للزبون القيام به. يمكنه بعد ذلك استخدام مجموعة من التعليمات لاستعراض محتوى المخدم و نسخ الملفات منه وإليه.

بروتوكول نقل البريد الإلكتروني SMTP

يستخدم هذا البروتوكول (Simple Mail Transfer Protocol <SMTP>) لنقل البريد على نحوٍ شفاف بالنسبة للمستخدم. حيث يرسل الحاسوب الزبون الرسائل إلى الحاسوب المخدم الذي يتولى عملية إيصالها إلى المرسل إليه. يستخدم هذا البروتوكول على نطاقٍ واسع جداً لتبادل البريد الإلكتروني (E-Mail).

برتوكول نقل النصوص الفائقة HTTP

يرتبط هذا البروتوكول (Hyper Text Transfer Protocol <HTTP>) ارتباطا وثيقا مع الوب "Web" حيث تستخدمه المتصفحات "Browsers" لإيجاد ونقل الوثائق الفائقة. ويُعتبر هذا البروتوكول مستقل الحالة "Stateless" حيث يفتح الزبون الاتصال مع المخدم ويأخذ المعلومات (الوثيقة الفائقة) ومن ثم يغلق الاتصال مباشرة.

كيف يعمل الويب:

يُعرّف الوب رسمياً على أنه: "مبادرة لاستخراج معلومات الوسائط المنهلة البعيدة تهدف إلى الوصول إلى عالم ضخم من الوثائق".

يسمح الوب للمستثمرين بالتفاعل فيما بينهم من خلال وثائق مخزنة على حواسيب مرتبطة مع شبكة إنترنت كما لو كانت أجزاء من نص واحد. وهو يعتبر طريقة للتشارك في المعلومات بين أناس عديدين في نفس الوقت حتى لو كانت هذه المعلومات أو المصادر موجودة في أطراف العالم. أصبح الوب الآن متاحاً للعديد من الناس ولم يعد استخدامه يقتصر على رجال العلم والاختصاصيين في مجال إنترنت، بل أصبح وسيلة اتصال بين شريحة كبيرة من المستثمرين.

تعمل خدمة الوب وفق النموذج زبون/مخدم (client/server)، أي أن برنامجا زبونا، يعمل عادة على حاسوب المستثمر (المستعرض)، يرسل طلباً بالبيانات المرغوبة إلى البرنامج المخدم الذي يعمل على حاسوب آخر في مكان ما على شبكة إنترنت. عندما يتلقى المخدم الطلب فإنه يرسل البيانات إلى برنامج المستعرض عبر الشبكة. يقوم المستعرض بعد تلقيه جواب المخدم بتفسير البيانات وعرضها على الشاشة

تشرح الخطوات التالية عملية الاتصال بين المستعرض ومخدم الوب:

1. يقوم المستثمر بتشغيل برنامجٍ زبونٍ يدعى مستعرض الوب (web browser) على حاسوبه الشخصي.

2. يتصل المستثمر مع الإنترنت باستخدام طريقة محددة عبر مزود خدمة الإنترنت Internet Service Provider (ISP). تعتمد طريقة الاتصال على ما هو متوفر عند مزود خدمة الإنترنت، الطريقة الشائعة هي الاتصال الشبكي الهاتفي (dial-up connection) باستخدام جهاز موديم، كما يمكن للاتصال أن يجري عبر وصلات ISDN أو T1 أو T3.

3. يطلب المستثمر صفحة من موقع ما على مخدم وب. يرسل المستعرض رسالة عبر شبكة إنترنت تتضمن المعلومات التالية:

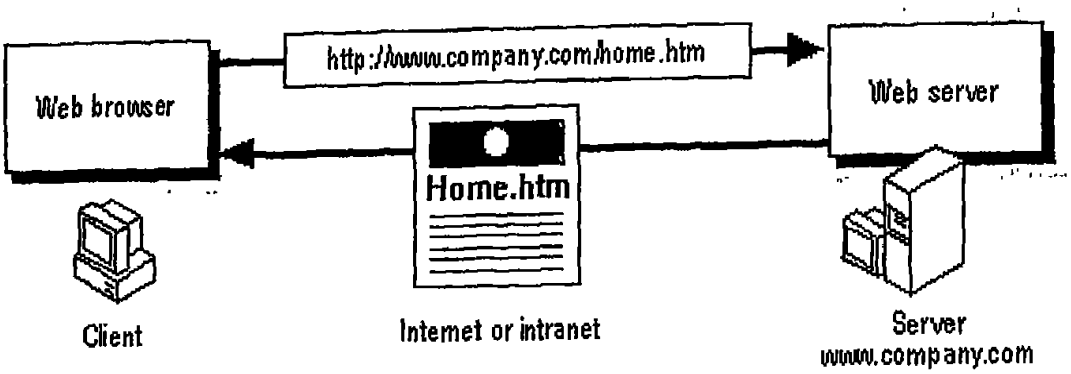
- برتوكول النقل المستخدم (http://).
- عنوان المخدم أو ما ندعوه محدد المصادر العام Uniform Resource Locator (URL). مثلا www.nice.com

4. يستلم مخدم الوب طلب المستثمر ثم يستخرج الصفحة المطلوبة المكتوبة بلغة HTML.

5. يرسل المخدم الصفحة المطلوبة عبر شبكة إنترنت إلى حاسوب المستثمر.

6. يستلم المستعرض نص HTML ثم يفسره ويعرض النتيجة على شاشة المستثمر.

تتضمن أغلب شبكات الحاسوب تقنية أمنية تدعى الجدار الناري (Firewall). والجدار الناري هو نظام برمجي (يترافق غالبا مع أجهزة خاصة) يشكل حاجزاً يمنع المستثمرين غير المرخص لهم من خارج الشبكة من الوصول إلى الموقع. إذا كان النظام مزوداً بجدار ناري فيتوجب على المستثمرين ضمن الشبكة استخدام برامج وكيلة (Proxy Programs) للوصول إلى إنترنت. مع أن هذا الأمر يضيف خطوات إضافية إلى عملية نقل البيانات بين المخدم والزبون إلا أن العملية الأساسية تبقى نفسها.



الحاسوب الزبون

:

يحتاج المرء لحاسوب زبون لاستعراض المعلومات المتناثرة على شبكة الوب. يجب أن يكون هذا الحاسوب مرتبطا مع شبكة إنترنت كما يجب أن يعمل عليه برنامج يدعى مستعرض الوب web browser.

يمكن أن يكون الاتصال مع شبكة الإنترنت مباشرا، كما يمكن استخدام الاتصالات الشبكية الهاتفية dial up عبر جهاز موديم للوصول إلى مزود خدمة إنترنت Internet Service Provider (ISP).

الحاسوب المخدم

:

من جانب مزود خدمة الوب يحتاج الأمر إلى حاسوب مخدم مرتبط مع شبكة إنترنت ويشغل برنـامج مخدم الوب الذي يدعى عادةً HTTPD أي Hyper Text Transfer Protocol Daemon.

يحتاج مخدم الوب إلى وصلة مع الإنترنت أفضل وأكثر متانة مما يحتاجـه زبـون الـوب، فيجـب أن يكون المخدم مرتبطاً مع إنترنت بواسطة وصلات مخصصة مثـل الوصـلات مـن النـوع T1 و T3 وأن تكون هذه الوصلات مفتوحة دائماً وإلا فلن يستطيع المستثمرون الذين يحاولون الوصول إلى موقع الوب على المخدم الاتصال معه.

يجب أن يملك مخدم الوب رقم IP خاصاً به وثابتاً وأن يكون هذا الرقم معروفاً على شـبكة إنـترنت. هنالك مخدمات وب من أجل العديد من البنـى الحاسـوبية المختلفـة، كمـا أن هنـالك مخدمـات وب لأغلب أنظمـة Unix التجاريـة وحتـى الحـرة، وكذلـك مخدمـات تعمـل علـى أنظمـة Macintosh و Windows NT وغيرها.

يجب أن يكون نظام التشغيل الذي يعمل عليه مخـدم الـوب متعـدد المـهام (Multitasking) ومتعـدد المسالك (Multithreading) بحيث يستطيع المخدم معالجة عدة طلبات في وقت واحد. كما أن القـدرة التخزينية له يجب أن تكون كبيرة إذ أن موقع الوب يمكن أن يتضاعف باستمرار.

الاتصال بين الزبون والمخدم:

لقد ذكرنا أن الاتصال بين الزبون والمخدم يجري باستخدام البروتوكول http. يُعتبر اتصال http اتصالاً غير دائم Stateless أي ليس هنالك اتصال مستمر بين مخدم الوب والزبون طوال فترة الطلب. فبعد أن يرسل الزبون طلبه يُقطع الاتصال. بعد ذلك يرسل المخدم الجواب إلى الزبون ثم يُقطع الاتصال. تعاد هذه العملية من أجل كل طلب وفي بعض الأحيان من أجل أجزاء من الطلب. يشار إلى طريقة التخاطب هذه بنموذج الاستفسار والاستجابة query–response model. هذا ما يفسر إظهار المستعرض للرسالة " Waiting for reply... " ضمن سطر الحالة في المستعرض.

يصوغ المستعرض طلبه بشكل قياسي ندعوه محدد المصادر العام (Uniform Resource URL Locator) والذي يدعى عادةً عنوان الصفحة أو الموقع. يأخذ URL دائماً الشكل التالي:

http://ServerAddress/Path/WebPageName

حيث :

"http://": يشير إلى أن المستعرض قد طلب إجراء عملية نقل باستخدام البروتوكول http؛ أي أنه يريد صفحة وب.

"ServerAddress": هو اسم المجال للمخدم الذي يطلب منه المستعرض صفحة الوب.

"Path": هو اسم المسار على القرص الصلب لمخدم الوب الذي توجد ضمنه صفحة الوب المطلوبة.

"WebPageName": هو اسم صفحة الوب المطلوبة.

ملاحظات

- إن URL هو سطر نصي مستمر بدون فراغات.

- يمكن لجزء البروتوكول ضمن URL أن يكون https بدلاً من http وهو ما يشير الاتصال بين الزبون والمخدم آمن.

يمكن لمحدد المصادر العام URL أن يشير إلى أية خدمة على الإنترنت وليس إلى صفحات الوب بما في ذلك خدمة نقل الملفات FTP وخدمة غوفر Gopher وخدمة WAIS وخدمة الأخبار enet Telnet.

يبين الجدول (3–2) صيغة البروتوكول المستخدم مع بقية المخدمات.

يمكن لجزء اسم مجال المخدم ServerAddress أن يحوي رقم البوابة التي يعمل عليها المخدم النحو التالي:

//ServerAddress:PortNb/Path/WebPageName

يطلب هذا من المخدم الوصول إلى الموقع عبر البوابة المحددة PortNb. رقم البوابة الافتراضي لـ الوب عادةً هو 80، وعند عدم ذكر رقم البوابة يُستخدم هذا الرقم.

صيغة البرتوكول	نمط الوصول
file://	ملف على الحاسوب المحلي
ftp://	مخدم FTP
Gopher://	مخدم Gopher
http://	صفحة وب
news://	مجموعة إخبارية على Usenet
telnet://	جلسة مع موقع telnet
wais://	مخدم WAIS

- يمكن لجزء البروتوكول ضمن URL أن يكون https بدلاً من http وهو ما يشير إلى أن الاتصال بين الزبون والمخدم آمن.

يمكن لمحدد المصادر العام URL أن يشير إلى أية خدمة على الإنترنت وليس إلى صفحات الوب فقط بما في ذلك خدمة نقل الملفات FTP وخدمة غوفر Gopher وخدمة WAIS وخدمة الأخبار Usenet و Telnet.

ينتهي اسم ملف صفحة الوب عادةً باللاحقة htm. أو html. التي تبين أن الملف يحوي نصاً مكتوباً بلغة HTML، وهي التي تعلم مستعرض الوب بضرورة قراءة الملف وتفسيره ثم عرضه ضمن نافذته. يمكن الوصول إلى أغلب الصفحات الابتدائية على مواقع الوب دون الحاجة إلى كتابة مسار الملف أو اسم الصفحة بل يُكتفى بكتابة عنوان الموقع بالشكل التالي:

http://ServerAddress/

يطلب هذا الأمر من المخدم الصفحة الافتراضية المخزنة على الدليل الجذر على المخدم، ولذلك فهي ليست بحاجة إلى مسار لأنها لا توجد ضمن دليل فرعي، وتسند أغلب مخدمات الوب اسماً افتراضياً إلى الصفحة الابتدائية مثل home.html أو index.htm. في حالة عدم ذكر اسم صفحة الوب يعيد مخدم الوب الصفحة الافتراضية.

يمكن أن يشير محدد المصادر العام URL إلى ملفات أخرى غير HTML، فمثلاً يمكن أن يكون الملف صورة أو برنامجاً تنفيذياً أو ملفاً مضغوطاً. مثلاً يعرض محدد المصادر العام http://ServerAddress/logo.gif الصورة المثلة بالملف logo.gif، أما محدد المصادر العام http://ServerAddress/Path/Program.zip فيمكن، بحسب إعداد المستعرض، أن يظهر رسالة تطلب حفظ الملف على القرص الصلب أو تعرض محتوى الملف المضغوط ضمن المستعرض.

أنواع الشبكات

- شبكات محلية *LANslocal area networks* - : بإمكانها ربط عدد كبير من الحاسبات بواسطة كبلات وأجهزة اتصالات كالمودم وتكون قريبة من بعضها جغرافيا، كأن تكون في نفس المبنى.
- شبكات واسعة *WANswide area network* - s : تربط الحاسبات بع شبكات تمتد عبر مناطق جغرافية واسعة وتتصل ببعضها عبر خطوط الهاتف أو الموجات اللاسلكية.
- شبكات المجمعات *CANs – sarea network campus* : تربط الحاسبات في منطقة جغرافية محدودة مثل الحرم الجامعي أو القواعد الحربية.
- شبكات متروبولية *MANs metropolitan area network* - s : شبكات للبيانات مصممة لخدمة المدن.
- شبكات منزلية *HANs – home area networks* : شبكات محصورة داخل المنازل وتربط الأجهزة الخاصة بشخص.
- شبكات المناطق العالمية Global Area Network وهي تسمى شبكات الموبايل

كيف نتصل بشبكة الانترنت:

ببساطة نحن نتصل بالانترنت عن طريق جهاز الكمبيوتر الذي يجب أن يحوي جهاز يسمى مودم , Modem طبعا هذه أكثر الطرق بساطة وبدائية

ما هو المودم : modem وهو جهاز يقوم بتحويل لغة الكمبيوتر الرقمية إلى نبضات تماثلية تشابه نبضات الهاتف وبالعكس يحول النبضات الهتاة إلى إشارات رقمية يفهما الحاسب بعد المودم نحن نحتاج لخط هاتفي نتصل به ونطلب رقم هاتف ما وهذا الرقم هو لجهة تعطينا اسم مستخدم وكلمة سر ندخل عن طريقها هذه الجهة وهي على الأغلب شركة خاصة تسمى مزود خدمة انترنت

ISP وهذا المزود يقدم خدمة الاتصال بانترنت لأنها يمثل بوابة لنا متصلة بمدخل الدولة كلها إلى الشبكة

وعند دخولنا الانترنت فأننا نحصل على شيء ما يسمى بالIP Address وهذا الرقم مميز أي لا يتكرر لكمبيوترين معا و يتألف هذا الرقم من عدة أرقام تحدد الدولة التي يأتي منها وحتى المدينة ومكان النفوذ للانترنت وهذا الرقم مضمن مع كل ضغطة تضغطها لطلب صفحة أو إرسال ايميل إلى أي مكان من جهازك.

كيف ترسل المعلومات عبر الشبكة :

أولا فلنتحدث عن شي ما اسمه بروتوكولات protocols وهي مجموعة من القواعد كما في الحياة العادية تتبعها الكمبيوترات فيما بينها أثناء عملية الاتصال والذي يهمنا هنا بروتوكول يسمى بروتوكول انترنت أو TCP/IP وهذا البروتوكول يحدد كيف تتدفق البيانات ضمن الشبكة وهذه البيانات تقسم إلى رزم صغيرة وترقم وترسل إلى الهدف وهناك قسمين هامين هما ال HEADER ويحوي رقم ال IP لجهازك وبعض المعلومات عن الملف المرسل و الجهاز الهدف. والقسم الثاني المقسم بدوره لأقسام صغيرة يتضمن بقية البيانات التي قمنا بإرسالها وهنا مثلا يستطيع صاحب موقع مكتوب مثلا أن يعرف أن جهازك دخل لمدونتي من المعلومات المضمنة في ال HEADER وهذا أمر يجب أن يكون هام وهو خطر أيضا عند إرسال معلومات خطيرة.

وهذه الأقسام ترسل إلى الهدف وتمر بنا يسمى عقد وهي نقط التقاء مسارات ضمن الشبكة ويجب أن تمر المعلومات من اقصر الطرق للهدف وان تمر أيضا كما في المرور الطبيعي بالعقد الأقل ازدحاما فإذا واجهت ازدحام عند نقطة مرورية ما تغير اتجاهها و تقوم بهذه العملية أجهزة تسمى موجهات.(ROUTERS)

طرفي اللقاء على الشبكة :

لدينا على الشبكة جهازين الأول هو أنت المستخدم Client فأنت تطلب موقع ما بعملية تسمى request وهي ضمن بروتوكول http:// وهذا هو أيضا بروتوكول انترنت أخر نضع بعده رمز ال .www وهو رمز شبكة الويب التي نستخدمها وهي جزء من الانترنت العامة ومن ثمة اسم الموقع yahoo مثلا نتبعه ب نقطة فاللاحقة له وهي هنا com مثلا وتسمى Domain أو نطاق بالعربية وهذا الاسم للموقع هو مميز أي لا يتكرر ولكن أين يكون الموقع هذا يكون مخزنا على الطرف الأخر من اللقاء العنكبوتي وهو الجهاز الخادم Server وهو جهاز بمواصفات كبيرة ومساحة تخزينية كبيرة وإمكانيات عالية فالخادم يحوي الكثير الكثير من المواقع لكن مثل ياهو او MSN او google تكون موزعه على عدة اجهزة خادمه لوحدها فقط.. والجاهز الخادم يقدم خدمة الايميل وهي خدمة تتيح لك اخذ اسم ما على هذا الموقع تستقبل به رسائل من أسماء أخرى لأشخاص على مواقع أخرى يتألف اسم الايميل من اسم مميز لاحق الإشارة @ ثم DOMAIN للموقع. لدينا أيضا ما يميز الخادم هو أو الموقع هو مصطلح يدعى الـ Bandwidth وهو يعبر عن عرض الحزمة التي يستطيع الموقع أو الخادم استيعابها في نفس اللحظة أي كمية المعلومات التي تمر خلال ثانية لهذا عندما يحصل ازدحام على موقع ما لا مكن دخوله لأن حزمة المرور تكون قد امتلأت

اهميه الشبكات وفوائدها

أهمية الشبكات في المكتبات ومراكز المعلومات

يُعد بناء الشبكات ضرورة استراتيجية في المكتبات ومراكز المعلومات للأسباب التالية:

-المشاركة في الاطلاع على المعلومات.

-نقل المعلومات باتباع سلوك منتظم ، ومن خلال تنظيم أفضل لمصادر الحوسبة.

-تقليص ازدواجية المعلومات.

-تطوير سرعة الوصول إلى المعلومات بسهولة ويسر.

-تطوير التفاعل بين المستفيدين من خلال المشاركة في المعلومات . Information sharing

أحد - CD-ROM تُعد المشاركة في قواعد بيانات الأقراص المدمجة الدوافع الرئيسة لإنشاء الشبكات في العديد من المؤسسات.

الذي يتيح متطلبات - Electronic-mail استعمال البريد الإلكتروني التفاعل بين المستفيدين وتبادل المعلومات والخبرات بينهم.

عن طريق الاتصال عن بعد - Library forums إقامة الندوات المكتبية بين المشاركين.

من خلال - Educational & Research Support دعم التعليم والبحث

المشاركة وتبادل المعلومات.

إذ يُكتفى - software's الحد من اقتناء أكثر من نسخة من البرمجيات بنسخة واحدة) مرخصة للشبكات وفق نظام حماية الحقوق (يتم استعمالها١٢ من قِبل جميع أطـــراف الشبكة، هذا علاوة على المشاركة في الملفات والأقراص المدمجة plotters والراسمات printersوالطابعاتfiles وغيرها من أجهزة تخزين البياناتCD-ROMs .

وزيادة فعاليتها الإنتاجية، ، - workgroups تكوين جماعات العمل وتيسير التعاون بين المستفيدين لتحديث بيانات المشروعات والجداول وقواعد البيانات والمشاركة فـــي معالجة بيانات الوثائق.

-الاتصال بالشبكات المحلية والعالمية وشبكة الإنترنت للاســتفادة مـــن المعطيـــات الا محدودة للشبكات

فائدة الشبكات

المشاركة في الموارد

مستخدمي الشبكة في مختلف أطرافها "نقاط عملها" يستطيعون أن يتشاركوا في المعلومات وفي استخدام آله طابعة وماسح رسوم واحدا Scanner أو المودم وأي معدات غالية الثمن . وعلى سبيل المثال إذا كان لديك كمبيوتر في البيت واشتريت كمبيوتر آخر لأولادك فإن من غير المنطقي أن يكون لكل كمبيوتر منها طابعه ولكل منها ماسح رسوم أو غيره ، بل انه من الممكن المشاركة في مودم واحد وبالتالي يكون كل كمبيوتر منها متصل بالإنترنت.

الأمن والسلامة

إن الشبكات لها مواصفات متقدمة من طرق الحماية ، وهذه المواصفات تسمح أو ترفض بشكل قاطع على العاملين على الأجهزة الأخرى من الوصول إلى المعلومات المخزونة في جهاز ما

تنظيم العمل ومركزيته

تسمح الشبكة بمركزيه قاعدة المعلومات ولذا فإن المستخدمين المتواجدين في إدارات أخرى يستطيعون الوصول إلي نفس مكان وجود المعلومات وهذا يقلل من الحاجة إلى تخزين المعلومات على كل جهاز

الاتصال السهل

تقدم الشبكات طريقه اتصال سهله ومريحة مثل الرسائل الإلكترونية والتراسل والاتصال بين مكانين أو أكثر . بل أن بإمكانك أن يقوم الكمبيوترين فيما بينها بمباريات العاب من الأمور الجيدة في هذا المجال أن برنامج التشغيل ويندوز Windowوما بعده يحتوي على برنامج إدارة شبكه كمبيوتر مما جعل إمكانية عمل هذه الشبكة سهلا ، بالإضافة إلى ذلك فإن تكاليفها قليلة وقد لا يصل تكلفه عمل شبكه بين كمبيوترين بأكثر من خمسين دولارا أو أقل .

إن الحد الأدنى الذي تحتاجه لهذا العمل هو

عدد اثنين من القطع الإلكترونية الوسيطة للشبكة (كارت الشبكة البيني Network Inteface Card.) واحدة لكل كمبيوتر .

•توصيله (كابل) من نوع ايثيرنت الرفيع Thin Ethernet ويشبه توصيله السلك الواصل ما بين الهوائي "إيريال" أو "الدش" والتلفزيون كما يستعمل قطعه إدخال Plug Daisy Chain مشابهة . ويتكون هذا الكابل من شبكه سلك لولبيه مغلفه لسلك مركزي موجود داخل عازل بلاستكي ويطلق عليه أيضا "Bus topology"وعند توصيله بجميع أجهزة كمبيوتر الشبكة فإنه يصبح في الواقع وكأن كابل واحد يسير بينها .

هناك طريقه أفضل ولكنها أكثر تكلفة وفيها يتم طريقه التوصيل باستخدام كيبل من نوع السلك اللولبي المزدوج Twisted Pair ويطلق عليه ١٠ Base-T وهي أغلى لاحتياجها إلى موزع شبكه خاص Network hub والذي يمكن وصف عمله وكأنه سنترال أو بداله تلفون أوتوماتيكية. وفي الواقع فإن هذه الطريقة تعتبر الحل الأفضل على المدى البعيد. يخرج من موزع الشبكة عدد من الأسلاك اللولبية التي ينتهي كل منها بجهاز كمبيوتر وبالتالى تتكون الشبكة

ولهذا تكون خلاصة فوائد الشبكات

يمكنك مشاركة المعلومات والمصادر على الشبكة، وهذا يقدم عدة فوائد :

١. يستطيع مشاركة طرفيات غالية الثمن مثل الطابعات حيث تستطيع كل الحواسيب استخدام نفس الطابعة .

٢. تستطيع نقل الـ **Data** أو البيانات المختلفة بين المستخدمين بدون استخدام الأقراص المرنة **FDD** . إن نقل الملفات على الشبكة يخفض الوقت اللازم لنسخ الملفات على الأقراص ومن ثم نسخها إلى حاسوب آخر .

٣. يستطيع جعل برامج معينة مركزية مثل الملفات المالية والحسابات ، فمعظم المستخدمين قد يحتاجون لاستخدام نفس البرنامج أو الولوج إلى نفس المعطيات معاً ، وبالتالي فهم يستطيعون العمل بشكل متزامن وبدون ضياع الوقت .

٤. تستطيع إجراء عملية النسخ الاحتياطي بشكل تلقائي وكامل وبذلك توفر الوقت وتضمن بأن كل عملك آمن .

أما في شبكات **WAN** فإن المصادر والمعلومات يمكن مشاركتها على مساحات جغرافية أوسع هذا يقدم عدد من الميزات :

٥. تستطيع أن ترسل وتستقبل البريد الألكتروني **E-mail** من والى كل أنحاء العالم ، ونقل وتبليغ الرسائل إلى أناس عدة في نفس الوقت وفي مساحات واسعة ومختلفة وبسرعة فائقة وتكلفة زهيدة

– تستطيع نقل الملفات من وإلى الشركاء في مواقع مختلفة، أو الدخول إلى شبكة الشركة من المنزل أو من أي مكان في العالم.

٦. يمكنك الدخول إلى مصادر ضخمة على الأنترنت والـ (**World Wide Web (www**

أنواع الشبكات من حيث علاقة الاجهزة مع بعضها البعض:

شبكات الند للند Peer-to-Peer Networks :

المقصود بشبكات الند للند أن الحواسب في الشبكة يستطيع كل منها تأدية وظائف الزبون و المزود في نفس الوقت ، و بالتالي فإن كل جهاز على الشبكة يستطيع تزويد غيره بالمعلومات و في نفس الوقت يطلب المعلومات من غيره من الأجهزة المتصلة بالشبكة.

إذا تعريف شبكات الند للند : هي شبكة حاسب محلية **LAN** مكونة من مجموعة من الأجهزة لها حقوق متساوية و لا تحتوي على مزود **Server** مخصص بل كل جهاز في الشبكة ممكن أن يكون مزودا أو زبوناً أي إن شبكات الند للند تنتمي لشبكات الإدارة الموزعة.

وهذا النوع من الشبكات يطلق عليه أيضا اسم مجموعة عمل **Workgroup** .

يمكن فهم مجموعة العمل بأنها مجموعة من الأجهزة التي تتعاون فيما بينها لإنجاز عمل معين ، وهي عادة تتكون من عدد قليل من الأجهزة لا يتجاوز العشرة ، حيث يستطيع أعضاء مجموعة العمل رؤية البيانات و الموارد المخزنة على أي من الأجهزة المتصلة بالشبكة و الاستفادة منها.

تعتبر شبكات الند للند مناسبة لاحتياجات الشبكات الصغيرة التي ينجز أفرادها مهام متشابهة، و نشاهد هذا النوع من الشبكات في مكاتب التدريب على استخدام الحاسوب مثلا .

يعتبر هذا النوع من الشبكات مناسبا في الحالات التالية فقط:

١. أن يكون عدد الأجهزة في الشبكة لا يتجاوز العشرة .

٢. ان يكون المستخدمون المفترضون لهذه الشبكه متواجدون في نفس المكـــان العـــام الذي توجد فيه هذه الشبكة .
٣. أن لا يكون أمن الشبكة من الأمور ذات الأهمية البالغة لديك .
٤. أن لا يكون في نية المؤسسة التي تريد إنشاء هذه الشبكة خطـــط لتنميـــة الـــشبكة وتطويرها في المستقبل القريب .

مميزات شبكات الند للند :

١. من المميزات الرئيسة لشبكات الند للند هو أن تكلفتها محدودة .
٢. هذه الشبكات لا تحتاج إلى برامج إضافية على نظام التشغيل .
٣. لا تحتاج إلى أجهزة قوية ، لأن مهام إدارة موارد الشبكة موزعة على أجهزة الشبكة و ليست موكلة إلى جهاز مزود بعينه.
٤. تثبيت الشبكة وإعدادها في غاية السهولة ، فكل ما تحتاجه هو نظام تشبيك بسيط من أسلاك موصلة إلى بطاقات الشبكة في كل جهاز من أجهزة الشبكة .

أما العيب الرئيسي لهذا النوع من الشبكات هو أنها غير مناسبة للشبكات الكبير ة و ذلك لأنه مع نمو الشبكة و زيادة عدد المستخدمين تظهر المشاكل التالية :

١. تصبح الإدارة اللامركزية للشبكة سبباً في هدر الوقت و الجهد و تفقد كفاءتها .
٢. يصبح الحفاظ على أمن الشبكة أمراً في غاية الصعوبة .
٣. مع زيادة عدد الأجهزة يصبح إيجاد البيانات و الاستفادة من موارد الشبكة أمراً مزعجاً لكل مستخدمي الشبكة .

بالنسبة لأنظمة التشغيل التي أصدرتها مايكروسوفت و تدعم شبكات الند للند فهي :

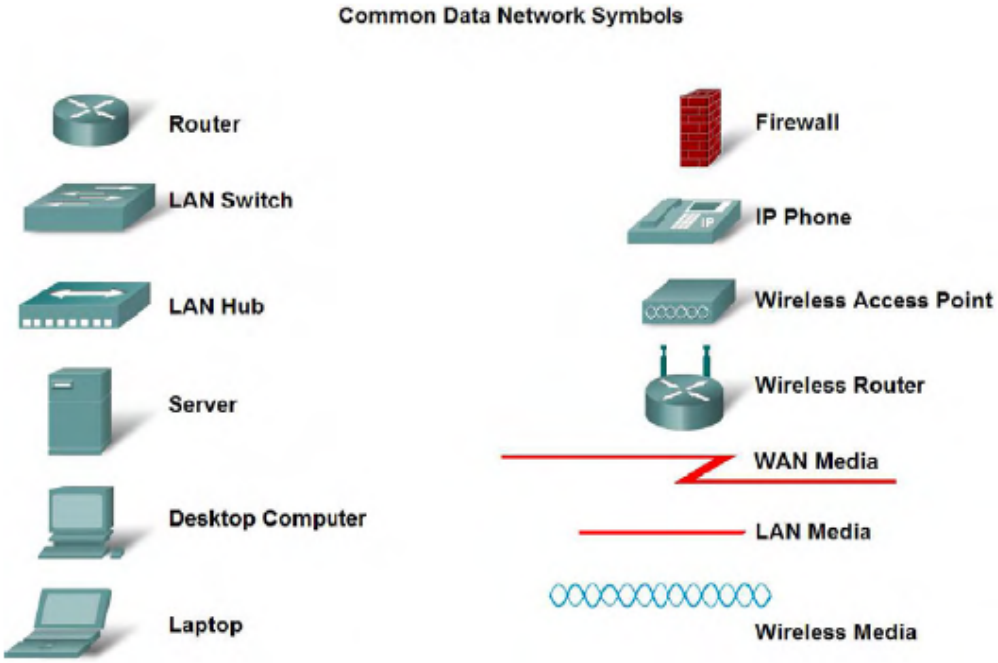
١. Windows 3.11 for Workgroup
٢. Windows 95
٣. Windows 98
٤. Windows Me
٥. Windows NT 4.0 Workstation
٦. Windows NT 4.0 Server
٧. Windows 2000 Professional
٨. Windows 2000 Server

و تعتبر أنظمة **Windows NT & Windows 2000** أفضل من باقي الأنظمة نظراً للأدوات التي تقدمها لإدارة الشبكة و المستوى العالي من الأمان الذي توفره للشبكة . و نلفت النظر أنه فيما يتعلق بشبكات الند للند فإن الأنظمة الأربع الأخيـــرة تتميـــز عـــن الأنظمة الأربع الأولى بالمميزات التالية :

١. يسمح لكل مستخدم بالاستفادة من موارد عدد غير محدود من الاجهزة المرتبطه بالشبكة.
٢. يسمح لعدد لا يزيد عن عشرة مستخدمين للاستفادة من موارد جهاز معين في الوقت نفسه .
٣. يسمح لمستخدم واحد بالتحكم عن بعد عن طريق خدمة الوصول بالتحكم عن بعد RAS (Remote Access Service) بجهاز مستخدم آخر .
٤. يوفر مميزات للحماية و الأمن غير متوفرة في أنظمة Win 9x

العتاد الصلب للشبكات

الأجهزة المستخدمة في الشبكات



بطاقة الشبكة

لكي يتمكن جهاز الكمبيوتر من الاتصال بالشبكة لابد له من بطاقة شبكة **Network Card Adapter**

والتي يطلق عليها أيضا الأسماء التالية:

١- **Card (NIC Network Interface).**

٢- **LAN Card.**

٣- **LAN Interface Card.**

٤- **Adapter LAN.**

تعتبر بطاقة الشبكة هي الواجهة التي تصل بين جهاز الكمبيوتر و سلك الشبكة، وبدونها لا تستطيع الكمبيوترات الاتصال فيما بينها من خلال الشبكة.

تثبت بطاقة الشبكة في شق توسع فارغ **Expansion Slot** في جهاز الكمبيوتر ، ثم يتم وصل سلك الشبكة إلى البطاقة ليصبح الكمبيوتر متصل فعليا بالشبكة من الناحية المادية و يبقى الإعداد البرمجي للشبكة .

دور بطاقة الشبكة

* تحضير البيانات لبثها على الشبكة.
* إرسال البيانات على الشبكة.
* التحكم بتدفق البيانات بين الكمبيوتر و وسط الإرسال .
* ترجمة الإشارات الكهربية من سلك الشبكة إلى بايتات يفهمها معالج الكمبيوتر ، و عندما تريد إرسال
بيانات فإنها تترجم إشارات الكمبيوتر الرقمية إلى نبضات كهربية يستطيع سلك الشبكة حملها .

كل بطاقة شبكة تمتلك عنوان شبكة فريد ، و هذا العنوان تحدده لجنة IEEE
و هذا اختصار ل
(Institute of Electrical and Electronic Engineers)
و هذه اللجنة تخصص مجموعة من العناوين لكل مصنع من مصنعي بطاقات الشبكة .
يكون هذا العنوان مكونا من ٤٨ بت و يكون مخزن داخل ذاكرة القراءة فقط ROM
في كل بطاقة شبكة يتم إنتاجها ، و يحتوي أول ٢٤ بت على تعريف للمصنع بينما تحتوي ٢٤ بت
الأخرى على الرقم المتسلسل للبطاقة.
تقوم البطاقة بنشر عنوانها على الشبكة ، مما يسمح للأجهزة بالتخاطب فيما بينها و توجيه البيانات إلى وجهتها الصحيحة.
ناقل البيانات هو المسئول عن نقل البيانات بين المعالج و الذاكرة .
لكي تعمل البطاقة كما يجب ، فإنها لابد أن تكون متوافقة مع نوعية ناقل البيانات في الكمبيوتر.
في بيئة عمل الأجهزة الشخصية هناك أربع أنواع لتصميم ناقل البيانات :

ISA
MCA
EISA
PCI

الشبكات المحلية .

ما هي الشبكة المحلية؟

الشبكة المحلية هي شبكة كمبيوتر (computer network) تنقل المعلومات بسرعة عالية ضمن مساحة جغرافية محدودة (مثلا: بناية واحدة أو عدة بنايات). وترتبط هذه الشبكة مجموعة من محطات العمل (workstations) مع بعضها، وذلك بما يُتيح لهذه المحطات تشارُك موارد الشبكة من عتاد (hardware) وبرمجيات (software)، إضافة إلى تمكين مستخدِمي الشبكة من تبادُل الملفات والاتصال فيما بينهم عبر البريد الإلكتروني (Email) والجلسات الحوارية.(chat)

ما هي طُرُق الولوج إلى الشبكة المحلية؟

كي تتمكَّن الأجهزة الموجودة في الشبكة المحلية من تبادُل المعلومات فيما بينها؛ لا بد لها من مجموعة من قواعد الاتصال المعيارية المتفَق عليها مُسبَقاً، وتدعى هذه القواعد بروتوكولاً(protocol) ، فمن أجل إرسال رسالة من جهاز إلى آخر عبر الشبكة، تُجزَّأ الرسالة في الطرف المرسِل إلى وحدات بيانات تُدعى الحُزَم(packets) ، وتُرسَل هذه الحُزَم عبر خطوط الاتصال ليُعاد تجميعُها في الطرف المستقبِل .

وهناك عدة بروتوكولات تُستَخدَم لِحلِّ مشكلة تشارُك وسط النقل (transmission medium) في الشبكات المحلية. وتعتمد هذه البروتوكولات إحدى الطريقتين التاليتين للوصول إلى الشبكة:

١. التنافُس :(contention) تطرأ الحاجة إلى التنافُس عند محاولة أكثر من جهاز كمبيوتر استخدام وسط النقل في الوقت نفسه، مما يؤدي إلى حدوث تصادم (collision).أما آليات تخفيف ذلك التصادم فهي عديدة، ومنها :

- تحسُّس وسط النقل :(carrier sensing) آليةٌ تعتمد على تأكد أجهزة الكمبيوتر من خُلوّ وسط النقل قبل استخدامه .

- تحرّي وسط النقل :(carrier detection) في هذه الآلية، تبقى أجهزة الكمبيوتر تراقب وسط النقل حتى أثناء استخدامها له .

ويُدعى البروتوكول الذي يستخدم كلا هاتين الآليتين بروتوكول CSMA/CD) اختصار للمصطلح الأجنبي carrier sense multiple access collision detect)، وهذا البروتوكول مستخدَم في جميع أنواع شبكات إيثرنت .(Ethernet)

٢. تمرير الشارة :(token passing) في هذه الطريقة، ينتظر جهاز الكمبيوتر الذي يريد استخدام الشبكة مرور شارة (token) تدور في الشبكة، وتخبره عند وصولها إليه متى يُسمَح له باستخدام الشبكة. ويُدعى البروتوكول الذي يستخدم هذه الطريقة بروتوكول توكن رينغ (token ring protocol).

٣. وتُعدّ طريقة تمرير الشارة (token passing) أفضل وأكثر معولّية من طريقة التنافس(contention) ، ولكنها- بالمقابل -أكثر كلفة .

٤. ما هي طُرُق الإرسال في الشبكات المحلية؟

تُرسَل المعلومات في الشبكات المحلية إلى العُقَد الأخرى بإحدى ثلاث طُرُق، وفي كل طريقة منها تُرسَل حزمة واحدة من المعلومات إلى عقدة أو أكثر، ففي الإرسال الأحادي (unicast) يتم الإرسال إلى عقدة واحدة، أما في الإرسال المتزامن المتعدِّد الوُجهات (multicasting) فيتم الإرسال إلى أكثر من عقدة، بينما في النوع الأخير المُسمى الإرسال العام أو البث (broadcasting) فتُرسَل حزمة المعلومات إلى جميع العُقَد في الشبكة .

التقنيات الرئيسة في الشبكات المحلية

هناك مجموعة من التقنيات التي تُستخدَم في الشبكات المحلية، وتتفاوت هذه التقنيات في سُرعاتها، وفي البروتوكولات التي تستخدمها، ونوعية الأوساط الناقلة فيها. ومن هذه التقنيات:

١. الإيثرنت(Ethernet): أكثر تقنيات الشبكات المحليةَ انتشاراً، وهي تستخدم الهيكليات الخطيةَ (bus topology) والنجميةَ(star topology) ، وتنقل المعلوماتِ بسرعةَ ١٠ ميغابت/ثانيةَ. وتعتمد جميع شبكات الإيثرنت بروتوكول (CSMA/CD)في الولوج إلى الشبكةَ، كما تستخدم- غالبا- كوابل محوريةَ (coaxial cables)وبعض أصناف الكوابل المَجدولةَ .(twisted pair)
وهناك أنواع جديدة مطوَّرة من الإيثرنت نذكر منها :

- إيثرنت السريعةَ:(fast Ethernet) تنقل المعلومات بسرعةَ ١٠٠ ميغابت/ثانيةَ، وتستخدم الكوابل المجدولةَ .(twisted pair)

- غيغابت إيثرنت:(gigabit Ethernet) تعتمد غيغابت إيثرنت- بشكل رئيس- على استخدام الألياف الضوئيةَ(optical fibers) ، وتصل سرعةَ نقلها للمعلومات إلى ١٠٠٠ ميغابت/ثانيةَ، فهي تتفوَّق على إيثرنت السريعةَ في هذا المجال. وتتوافق غيغابت إيثرنت بشكل كامل مع سابقاتها من شبكات إيثرنت .

٢. شبكةَ توكن رينغ المحليةَ :(Token ring LAN) تعتمد هذه الشبكةَ بروتوكول توكن رينغ، وهي تستخدِم طريقةَ تمرير الشارة(token passing) لمنع التصادُم الذي قد ينجُم عن قيام أكثر من كمبيوتر باستخدام الشبكةَ في الوقت نفسه. وترتبط أجهزةَ الكمبيوتر في هذه الشبكةَ وفق هيكليةَ حلقيةَ أو نجميةَ أو خطيةَ. وتنتقل المعلومات عبر هذه الشبكةَ بسرعةَ تتراوح بين ٤ و ١٦ ميغابت/ثانيةَ .

٣. شبكةَ البيانات الموزَّعةَ بالألياف الضوئيةَ (fiber-distributed data interface- FDDI):تستخدَم هنا خطوط من الألياف الضوئيةَ لنقل المعلومات في الشبكةَ المحليةَ ضمن مساحةَ تصل إلى ٢٠٠ كم .وتعتمد شبكات FDDI على طريقةَ تمرير الشارةَ(token passing) التي تعتمد بروتوكول توكن رينغ (token ring)؛ ولكنها تحتوي على حلقتَي توكن تكون إحداهما احتياطيةَ في حال تعطل الأخرى، وتعمل على نقل المعلومات في الحالات العاديةَ مما يضاعف سرعةَ النقل إلى ٢٠٠ ميغابت/ثانيةَ. وتكمن فائدةَ هذه النوعيةَ من الشبكات في أنها

تغطي آلاف المستخدمين، وتُستَخدَم على أنها عمود فقري (backbone) للشبكات الواسعة .(WAN)

٤. ويوجد تقنيات عديدة أخرى للشبكات المحلية مثل:

التحويل المتعدِّد الطبقات(Multilayer switching) ، والشبكات التي تعتمد بروتوكول.STP (spannig tree protocol)

أجهزة الارتباطية(connectivity devices)في الشبكات المحلية كي يتمّ الاتصال عبر الشبكة، لا بُدَّ من استخدام بعض تقنيات وأجهزة الارتباطية. ونستعرض فيما يلي بعضاً من هذه الأجهزة والوظائف التي تقوم بها.

١. المودم:(modem)

من المعلوم أن أجهزة الكمبيوتر تتعامل مع الإشارات الرقمية (digital signals)فقط، ولكن خطوط الهاتف العادية لا تنقل سوى الإشارات التواصلية (analog signals).ولهذا، لا بد من وجود جهاز- هو المودم -(modem)يحوِّل الإشارات الرقمية (digital) إلى تواصلية (analog)في الطرف المرسِل عبر عملية تُدعى التعديل(modulation) ، ثم تُرسَل الإشارات الناتجة عبر خطوط الهاتف، ليصار إلى تحويلها من تواصلية إلى رقمية في الطرف المستقبِل عبر عملية تدعى فك التعديل .(demodulation)ومن هنا كانت أجهزة المودم قادرة على ربط أجهزة كمبيوتر، أو حتى شبكات كاملة بعيدة عن بعضها باستخدام خطوط الهاتف.

وتَصِل سُرعات المودم حاليا إلى ٥٦ كيلوبت/ثانية، ولكن هنالك تقنيات جديدة قد تحل محل المودم العادي، وهي تنقل المعلومات بسُرعات كبيرة جداً، ومن هذه التقنيات: الشبكة الرقمية للخدمات المتكاملة ISDN (Integrated services digital network)، والخط الرقمي للمشترك (digital subscriber line).

٢. الموزِّع الشبكي:(hub)

تتصل أجهزة الكمبيوتر في معظم أنواع الشبكات المحلية- عدا شبكات إيثرنت التي تستخدم كوابل محورية -(coaxial cables) بجهاز يقوم بدور نقطة وصل

مركزية بين أجهزة الشبكة، وهو يدعى الموزع الشبكي(hub) ، ووظيفته هي ربط قِطع الشبكة (segments) ببعضها. ومن أنواع الموزِّعات :

- الموزِّع المنفعِل :(passive hub) يُمرِّر هذا النوع الإشارات الواردة من القِطع (segments) المختلفة للشبكة، وتستطيع جميع الأجهزة الموصولة معه استقبال حُزَم (packets) المعلومات المارة عبره .
- الموزِّع الفاعل :(active hub) يحوي هذا الموزِّع أجزاء إلكترونية تُعيد توليد (regenerate) الإشارات المارة في الشبكة. وتكمن فائدته في زيادة معوِّلية الشبكة، والسماح بمسافات أكبر بين أجهزتها. ويوجد منه نوع محسَّن يُدعى الموزع الشبكي الذكي.(intelligent hub)

٣. المكرِّر(repeater)

تتعرَّض الإشارة أثناء عملية الإرسال للتشويش والتشويه عبر خطوط النقل، مما ولَّد الحاجة إلى تصميم جهاز يدعى المكرِّر (repeater) يستخدم لإنعاش الإشارة المرسلة عبر الشبكة، بحيث تبقى قوية عند وصولها إلى محطات العمل المستقبلة لها. ويوجد نوعان من هذه المكرِّرات: تواصلي (analog) يضخِّم الإشارة وحسب، ورقمي (digital) يعيد بناء الإشارة لتصبح قريبة جداً من الأصلية .

٤. الجسر(bridge)

لتوسيع حجم الشبكات الموجودة صُمِّم جهاز يدعى الجسر (bridge) يمكنه ربط قطعتين (segment) من شبكة محلية، كما يمكنه ربط شبكتين محليتين تستخدمان البروتوكول نفسه.

وقد صُمِّم جهاز آخر يدعى المحوِّل (switch) لتحديد المسار الذي تُنقل عبره حزم (packets)المعلومات بين القِطع (segments) المختلفة للشبكة المحلية، وتدعى الشبكات المحلية التي تستخدمه.(switched LAN)

٥. الموجِّه(router)

مع الازدياد الهائل في عدد الشبكات المحلية، لم يكن الجسر (bridge) قادراً على إجراء هذا الربط، فكان الحل في جهاز يدعى الموجِّه (router) يقوم بهذا الربط. ويمرر هذا الجهاز حزم (packets) المعلومات بالاعتماد على عناوين منطقية،

كما يتبع خوارزمية تمكنه من اختيار المسار (route) الأفضل لنقل حزم المعلومات إلى هدفها عبر الشبكات الأخرى. أما في الإنترنت، فيمكن أن يكون الموجه جهازاً أو برنامجاً يحدد المسار الأفضل عبر العقد للوصول إلى الهدف.

٦. البوابة(gateway)

أدّى عدم مقدرة الموجه (router) على ربط شبكات محلية تستخدم بروتوكولات مختلفة- إلى استخدام ما يدعى البوابة(gateway) ، وهي مجموعة من الأجهزة والبرامج التي تربط بين شبكات تستخدم بروتوكولات مختلفة، إذ تنقل المعلومات وتحولها إلى صيغة تتوافق مع بروتوكولات الشبكة الأخرى .

الشبكات المتوسطة.

إن الشبكات المتوسطة MAN (وتجمع MANS وليس MEN) هي نسخة مكبرة من LAN وغالباً ما تستخدم نفس مخطط التوصيل ويمكنها أن تغطي مجموعة مكاتب متجاورة أو حتى موزعة ضمن مدينة واحدة . كما يمكن أن تكون خاصة أو ذات ملكية عامة ، ويمكن لشبكات MAN أن تدعم نقل المعطيات والصوت ويكنها أن تستعمل شبكة التلفزيون الكابلي في المدينة . ولا تحوي شبكة MAN أي أجهزة تبديل كما أنها يمكن أن تتألف من كابل رئيسي واحد أو كابلين .www.tartoos.com

إن حقيقة عدم وجود عناصر تحويل في الشبكة يساعد كثيراً في تبسيط تصميمها . إن السبب الأساسي الذي يجعلنا نضع هذه الشبكة في فئة مستقلة بذاتها هو أنه تم إنشاء معيار خاص بها يدعى IEEE802.6 أو الممر المضاعف ذو خط الانتظار الموزع DQDB (Distributed Queue Dual Bus)

وتتألف DQDB من ممري وحيدي الاتجاه ويتم وصل كل الأجهزة اليهما وكل ممر له نهاية رأسية وهو جهاز يقوم بتهيئة عملية النقل وتعبر المعلومات المتوجهة الى حاسوب يقع على يمين المصدر الممر العلوي ، بينما يستعمل الممر السفلي للرزم المتوجهة الى حاسوب على يسار المصدر .

السمة الرئيسية لشبكة MAN وهو وجود وسط للبث العام ، في حالة 802.6 هو كابلين يتم وصل كل الأجهزة عليهما وهذا ما يبسط التصميم مقارنة مع باقي أنواع الشبكات .

الشبكات الواسعة.

ما هي الشبكة الواسعة(WAN)

الشبكة الواسعة (WAN) هي شبكة كمبيوتر لِتبادُل المعلومات الرقمية ضمن مساحة جغرافية واسعة (قد تشمل عدة دول)؛ وهي أكبر من الشبكة المحلية(LAN) ، وقد تستخدِم خطوط الهاتف والأقمار الصناعية وغيرها من وسائط نقل البيانات .وفي بعض الأحوال، قد تتكوَّن الشبكة الواسعة من ربط عدة شبكات محلية معا.

ما أهميتها وفوائدها؟

تكمن فائدة الشبكات الواسعة في أنها تُتيح نقلا آمنا وسريعا للمعلومات بين العُقد المختلفة، ناهيك عمّا يمتاز به نقل المعلومات عبر الشبكة الواسعة من موثوقية عالية، وانخفاض الكُلفة.

ولعلّ المنظمات والشركات الكبيرة التي تنتشر فروعها في أرجاء العالم المختلفة -هي من يُحقِّق الاستفادة الكُبرى من الشبكات الواسعة؛ لأن هذه الشبكات تُتيح لها الاتصال مع موظفيها وزبائنها وشركائها عبر العالم. وللشبكات الواسعة دور كبير في تشجيع وحفز الأعمال الإلكترونية (e-business) التي انتشرت في عصر الإنترنت .

وفي الغالب، تقوم شركات الاتصالات الحكومية (public telecommunications companies- PTT) في البلاد المختلفة بالإشراف على الشبكات الواسعة وصيانتها؛ كما تقدِّم هذه الشركات خدمات معيَّنة لمستخدِمي الشبكات الواسعة مثل خدمة الخط المستأجَر.(leased line)
وفي الفقرات التالية، سنتعرف على طرق توصيل الأجهزة ببعضها في الشبكات الواسعة :

الوصل نقطة بنقطة :(point-to-point connection)

تعتمد هذه الطريقة الخطَّ المستأجر (leased line) لوصل مكانين متباعدين على الشبكة بوساطة وصلة وحيدة كما في الشكل (١). ويكون الإرسال عبر هذه الوصلة على نوعين، أما الأول فهو إرسال الحُزَم المعنونة[datagram transmission] الذي تُرسَل فيه المعلومات حزمة إثر حزمة، وأما النوع الثاني فهو الإرسال التدفقي للبيانات (data-stream transmission)الذي تُرسَل فيه البيانات بايت إثر بايت. وتتميز هذه الطريقة بأن الخط محجوز بشكل دائم للزبون، ولكنها -بالمقابل- طريقة مرتفعة الكُلفة. أما من كانت ميزانيته محدودة، فينبغي عليه استخدام طُرق أخرى أقل كلفة، ومنها طريقة التحويل عبر دارة.(circuit switching)

ماهو بروتوكول TCP/IP وماوظيفته ؟

إن الإنسان والكمبيوتر لهما ميزتان متشابهتان، وهي أن كل منهما يستعمل لغةً معقدة للتفاهم. فإذا أراد شخصان يتحدثان لغتين مختلفتين، ولنقل العربية واليابانية مثلا أن يتفاهما، فإن عليهما أن يستخدما مترجما بينهما، أو أن يتحدث الاثنان بلغةٍ ثالثةٍ ولنقل الإنجليزية مثلا . إن أجهزة الكمبيوتر غير موحدة في طريقة صنعها أو تشغيلها، فهي تعمل بلغات وبنظم تشغيل مختلفة، منها نظام دوس ونظام يونكس ونظام ماكينتوش وغيره، ولكي نجعل هذه الأجهزة تتصل مع بعضها بواسطة شبكة واحدة (الإنترنت) وتتفاهم فيما بينها من خلال تلك الشبكة، فإن الإنترنت يستخدم مجموعة بروتوكولات معينة، ودعنا هنا نسميها "لغة" من أجل التقريب، وهي: Transmission Control Protocol// Internet Protocol ويطلق عليها اختصارا TCP/IP لقد تم اختراعها سنة ١٩٧٠، وكانت جزءا من أبحاث مؤسسة DARPA، التي قامت لتوصيل أنواع مختلفة من الشبكات وأجهزة الكمبيوتر. كان تمويل هذه المؤسسة عاما من أجل تطوير هذه "اللغة"، ولذلك فإنها تتصف بعدم تبعيتها لأحد ، والنتيجة أنها أصبحت ملكا عاما، وبالتالي لا يمكن لأحد ادعاء الحق باستخدامها له فقط. وأكثر من هذا فان بروتوكولات TCP/IP تتكون من عتاد Hardware وبرامج Software مستقلة، ولذلك فإن أي شخص يمكن له أن يكون متصلا بالإنترنت، ويشارك في المعلومات، مستخدما أي نوع من أجهزة الكمبيوتر. ما هو البروتوكول؟ البروتوكول بالنسبة للكمبيوتر على الإنترنت عبارة عن مجموعة القواعد التي تحدد كيف يمكن لأجهزة الكمبيوتر أن تتفاهم مع بعضها البعض عبر الشبكة التي تتواجد عليها. وشبكة الكمبيوتر تعني جهازي كمبيوتر أو أكثر متصلة مع بعضها البعض وقادرة على أن تتشارك في المعلومات . عندما تتحادث أجهزة الكمبيوتر مع بعضها البعض فإن ذلك يعني تبادلها مجموعة من الرسائل. وحتى يكون في إمكانها فهم تلك الرسائل والعمل على تنفيذها فإن على أجهزة الكمبيوتر الموافقة على العمل بقواعد واحدة متفق عليها. فإرسال واستقبال البريد الإلكتروني ونقل الملفات والمعلومات وغيرها هي أمثلة على ما تقوم به أجهزة الكمبيوتر عبر الشبكات باستخدام مجموعة القواعد التي تحدد طريقة تفاهم أجهزة الكمبيوتر مع بعضها أو ما أسميناه بالبروتوكول. إن البروتوكول يقوم بوصف الطريقة التي يجب على تلك الأجهزة أن تتبادل فيها الرسائل وتنتقل المعلومات . البروتوكول يختلف باختلاف نوع الخدمة التي تقدمها الشبكة. وعلى سبيل المثال فإن الإنترنت قد تأسس على مجموعة البروتوكولات التي تكون عائلة واحدة هي TCP/IP . في الواقع عبارة عن بروتوكولين مختلفين ولكنهما يعملان معا دوما في نظام

الإنترنت، ولهذا السبب فإنهما أصبحا مقبولين لأن يوصفا بأنهما وكأنهما نظام واحـد.

برتوكول الانترنت

اختصار للعبارة الإنجليزية (Internet protocol)، وينضوي هذا البروتوكول تحت مجموعة بروتوكولات (TCP/IP) التي تتحكم بتجزيء رسائل البيانات المُرسَلة إلى حُزَم (packets)، وتوجيه هذه الحُزَم من المرسِل إلى المستقبِل، إضافة إلى إعادة تجميع الحُزَم لتشكيل رسائل البيانات الأصلية لدى المستقبِل

كيف يتم التطبيق ؟؟؟؟

ولكي يمكن تطبيق هذا الشئ كان لا بد لهم من الوصول الى نظام موحد للشبكة يمكنهم من تطبيق هذا النظام على الإنترنت على:

أولا : أن يكون لهذا النظام الموحد قدرة على ازالة الحواجز الناتجة عن الإختلافات في مواصفات واشكال و أنواع أجهزة الكمبيوتر المرتبطة مع بعضها البعض بواسطة الشبكة

ثانيا : عدم تأثر هذا النظام بالتطورات التي تطرأ على التكنولوجيا المرتبطة بصناعة أجهزة الحاسب

ثالثا : أن يقوم هذا النظام بإرسال و إستقبال المعلومات على شكل حزم صغيرة من المعلومات تكون قادرة على حرية التنقل والحركة من عقدة الى اخرى في الشبكة دون الإعتماد على الإتصال المفتوح و الدائم بين جهازين كما هو الحال في الهاتف وهذ المطلب كان لأسباب عسكرية وتلبية لخاصية عدم الإعتماد على خط تنقل واحد للمعلومات تفاديا لإنقطاع خدمة تراسل المعلومات في حال وجود دمار أو خلل على احد فروع الشبكة وعلى هذه الخلفية تم تطوير نظامين وهما المعروفين ب IP) Internet protocol) TCP) Transmission Control Protocol and)

ماذا يعني TCP/IP

TCP يعني لنا و مهمة هذا البروتوكول هوالتأكد من أن حزمة المعلومات التي أرسلت من نقطة (عقدة) الى أخرى قد وصلت كاملة أم لا وكذلك يقوم هذا البروتوكول بتنبيه الجهاز المرسل في حالة تعثر وصول المعلومة الى وجهتها المطلوبة

IP يعني لنا و هو نظام التوجيه وهي مسؤولة عن تحديد العقد و المسارات التي تسلكها حزم المعلومات للوصول الى الجهاز الهدف

البروتوكول TCP Transmission Control Protocol

كما نعلم أن البرتوكول TCP/IP مكون من بروتوكولات مختلفة كل منها له عمل أو خدمة يقدمها من أجل الإرسال عبر الشبكة وأول بروتوكول هو TCP وهو عبارة عن برتوكول يتحقق من وصول الإرسال وهو من نوع Connection-based ويحتاج إلى أنشاء جلسة عمل قبل أرسال البيانات بين الحواسيب كما يتأكد من أن جميع الرزم التي أرسلت قد تم إستقبالها من الجهاز الاخر وإذا لم تصل هذه الرزم يقوم TCP بإرسالها

مره ثانيه وإذا تم الإستلام يأخذ شهادة مصادقه ويقوم بإرسال الدفعه التاليه..... وتتم عملية Connection Based كما يلي يتفق الحاسبان على الطريقة الأصلح لتحديد كمية البيانات التي سوف يتم إرسالها في وقت واحد وعلى أرقام المصادقة التي سيتم إرسالها عند استلام البيانات وما هو الوقت المناسب لقطع الإتصال ... هذا ما يسمى إنشاء جلسة عمل وكما ترى فإن هذا البروتوكول قد يسبب حملا زائدا عند إرسال كمية كبيرة من البيانات

البروتوكول User Datagram Protocol UDP

أما البرتوكول الثاني فهو UDP وهذا البرتوكول هو من نوع Noconnection-Based بمعنى الإتصال غير الموثق وهو لا ينشئ جلسة عمل بين الحواسيب أثناء الاتصال وهو لا يضمن وصول البينات مثل ما ارسلت به وهو عكس TCP ولكن هذا البرتوكول له مميزات تجعل يستحب إستخدامه في بعض الحالات مثل عند إرسال بينات جماعية عامة وعند الحاجة إلى السرعة وسرعته من عدم حاجته إلى التحقق من دقه الإرسال ويستخدم في نقل الوسائط المتعدده مثل الصوت و الفديو لان الوسائط لا تحتاج إلى دقه الوصول ونستطيع أن نقول أن هذا البروتوكول ذو فاعلية كبيرة وسريع الأداء... ومن أهم الاسباب التي أدت إلى إنشاء البروتوكول UDP أن الإرسال عبر هذا البروتوكول لا يتطلب إلا القليل من الحمل و الوقت إذ أن رزمة UDP لا تحتوي على كل المعطيات التي ذكرت مع البروتوكول TCP لمراقبة الإرسال .. لذلك سمى

بروتوكول الإتصال غير الموثق

البروتوكول IP Internet Protocol

وهو يعد من أهم البروتوكولات لوجود عنصر العنونة الذي يستخدمه لإعطاء كل حاسب على الشبكة رقما خاصا به ويسمى عنوان انترنت IP Address وهو عنوان متفرد ليس له شبيه في النطاق الشبكي ويتميز IP بميزتين مهمتين وهي التوجيه و شطر الرزم و إعاده الرزم فالتوجيه يقوم بفحص العنوان الموجود على الرزمه ويعطيه تصريح تجول في أرجاء الشبكة وهذا التصريح له مده محددة فإذا انتهت هذه الفترة الزمنية ذابت تلك الرزمه ولم تعد تسبب إزدحام داخل الشبكة ..و عملية التشطير تستخدم في التوليف بين بعض انواع الشبكات المختلفة مثل شبكة Token-Ring و Ethernet بسبب ما لشبكة توكن رنغ من سعه في نقل الإشارات لذالك وجب تشطيرها ثم إعادة التجميع مره اخرى

البروتوكول Protocol ICMP Internet Control Message

وهو مسؤول عن رسائل الاخطاء التي تتعلق بتامين وصول IP ويحتوي على رسائل من اشهرها التي تاتي مع الاداة Ping وهي رسالة EchoReqest و Echo Reply البروتوكول ARP Address Resolution Protocol يقوم هذا البروتوكول بعمل جدا مهم وهو وصف وإرشاد خدمة IP عن العنوان الفيزيايئ للعنوان المطلوب اذ يقوم IP عند إستلام طلب الإتصال بحاسب ما مثلا X يتوجه فورا إلى خدمة ARP ويسأله عن مكان هذا العنوان على الشبكة ثم يقوم البروتوكول ARP بالبحث عن العنوان في ذاكرته فإذا وجده قدم خريطة دقيقة للعنوان و إذا كان العنوان لحاسب في شبكة بعيدة يقوم ARP بتوجيه IP إلى عنوان الموجه Router ثم يقوم هذا الموجه بتسليم الطلب ل ARP حتى يبحث عن العنوان الفيزيايئ لرقم ال IP كيف يعرف هذا البوتوكول العنوان الفيزيايئ للحواسيب يعرفه برقم كرت الشبكة إذ كل كرت يصنع من المصانع المختللفة يكون له رقم فريد لا يشبه رقم اخر فيحتفظ ARP بهذه الارقام في ذاكرته التي تشبه قاعدة البيانات بجميع الارقام الخاصة في محيط الشبكة ،،وهذا البروتوكول من أدوت الفحص التي تستخدم في مراقبة الشبكة وتحديد بعض المشاكل

البروتوكول TCP Transmission Control Protocol

كما نعلم أن البرتوكول TCP/IP مكون من بروتوكولات مختلفة كل منها له عمل أو خدمة يقدمها من أجل الإرسال عبر الشبكة وأول بروتوكول هو TCP وهو عبارة عن برتوكول يتحقق من وصول الإرسال وهو من نوع Connection-based ويحتاج إلى أنشاء جلسة عمل قبل أرسال البيانات بين الحواسيب كما يتأكد من أن جميع الرزم التي أرسلت قد تم إستقبالها من الجهاز الاخر وإذا لم تصل هذه الرزم يقوم TCP بإرسالها

العلاقة بين OSI و :

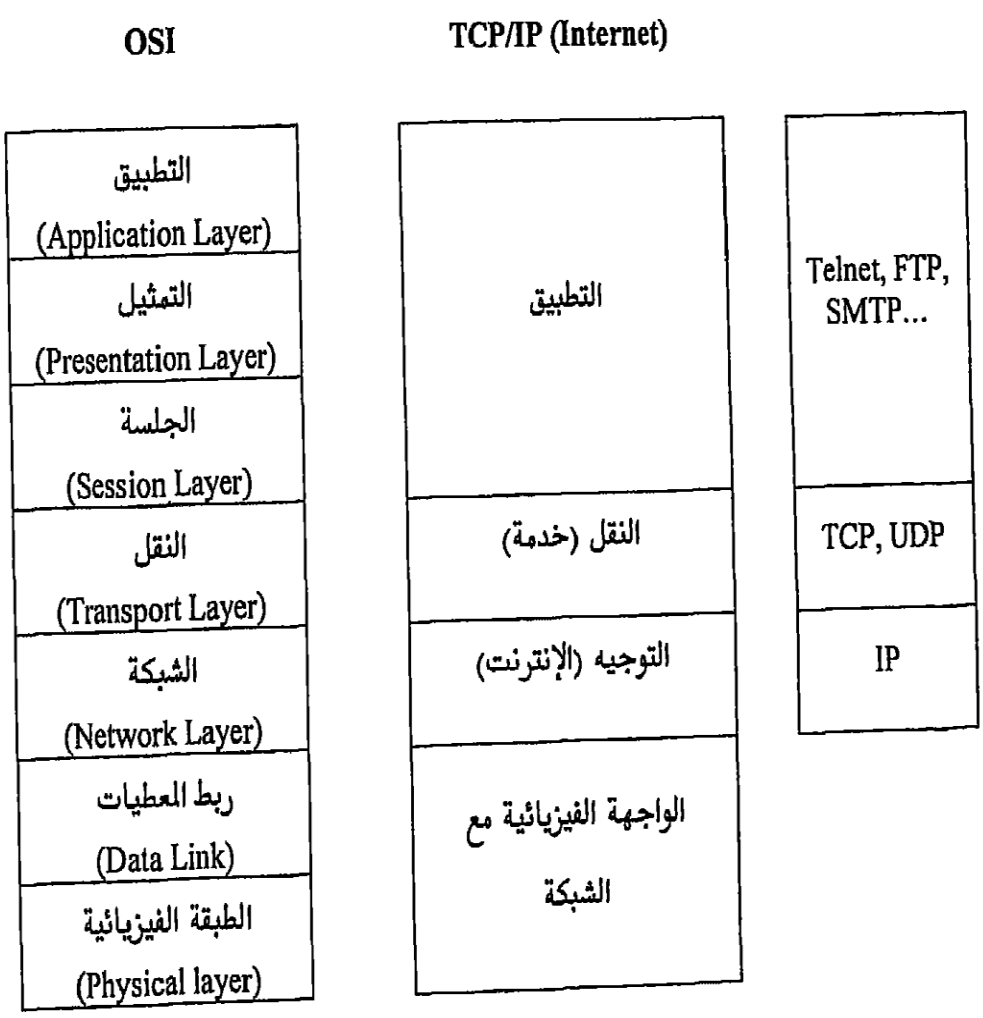
TCP\IP

يُشار إلى البنية التصميمية لـ TCP/IP على أنها البنية التصميمية للإنترنت، فعندما يُذكر TCP/IP يُقصد به أيضاً بنية الإنترنت. تتألف بنية الإنترنت (بنية TCP/IP) من أربع طبقات هي: التطبيق

والنقل والتوجيه والطبقة الفيزيائية. تملك بنية TCP/IP بروتوكولين أساسيين وهامين هما: TCP وIP، الأول هو البروتوكول المستخدم في طبقة النقل، بينما يُستخدم الثاني في طبقة التوجيه.

إن بنية الإنترنت لا تماثل البنية OSI للشبكات (النموذج المعياري للشبكات)، إلا أنها لا تخالفها تماماً. فبنية الإنترنت تتألف من 4 طبقات، بينما تبدو بنية OSI أكثر تفصيلاً فهي تتألف من 7 طبقات. مما يعني استقلالاً أكثر بين الطبقات المختلفة مقارنةً بـ TCP/IP.

تظهر في الشكل (2-4) بنية كل من TCP/IP وOSI، وفيه نلاحظ نقاط التشابه ونقاط الاختلاف، فنموذج TCP/IP لا يملك طبقتي تمثيل وجلسة، كما أنه يدمج طبقة ربط المعطيات والطبقة الفيزيائية في طبقة واحدة هي طبقة الواجهة الفيزيائية مع الشبكة.



البحث عن المعلومات على الوب

إن أكبر المشاكل التي تصادف الأشخاص الذين يبدؤون باستكشاف الوب تتمثل في العثور على المعلومات. في هذه الفقرة نناقش محركات البحث الشهيرة والأدلة ونقدم نصائح حول استخدامها.

قد يبدو العثور على وثائق الوب (مثل صفحات الوب أو المواقع) التي يرغب فيها المستثمر أمرا سهلا كما قد يكون عملا في غاية الصعوبة. يعود هذا من ناحية إلى الحجم الضخم لشبكة الوب، ومن ناحية أخرى إلى أن الوب ليس مفهرسا بطريقة هجائية (مثل فهارس الكتبات التي تتبع مقاييس شائعة أو مثل فهارس الصحف التي تتبع مقياسا خاصا بها).

لهذا السبب، عندما يقوم المستثمر بما ندعوه "بالبحث ضمن الوب" فإنه لا يبحث ضمنه مباشرة فهذا ليس. الوب هو مجموع صفحات الوب العديدة التي تستقر على المخدمات المنتشرة في أنحاء العالم ولا يمكن لحاسب المستثمر أن يذهب إليها مباشرة. لكن المستثمر يستطيع النفاذ إلى عدد من قواعد البيانات الوسيطة التي تحوي معلومات حول صفحات الوب المنتشرة والتي تنظمها بطريقة تسمح بالعثور على الصفحات المطلوبة. يمكن الوصول مباشرة إلى أدوات البحث هذه التي تعيد محددات المصادر العامة (URL) إلى الصفحات المطلوبة. يمكن بعد ذلك النقر على إحدى هذه المحددات والوصول إلى الصفحة المرغوبة.

ندعو المواقع التي تحتضن قواعد البيانات هذه بأدوات البحث وهي تختلف بعضها عن بعض من حيث طريقة عملها. فيما يلي نستعرض أنواع أدوات البحث.

فئات أدوات البحث

يمكن تقسيم أدوات البحث المنتشرة على شبكة إنترنت إلى ثلاث فئات : مواقع الفهرسة ومحركات البحث ومحركات البحث البينية.

مواقع الفهرسة:

تنظم مواقع الفهرسة (Index Sites) المعلومات ضمن بنية محددة. يقوم القائمون على موقع الفهرسة بتصميم البنية المستخدمة لمعلومات الموقع وبتحديد المعطيات الداخلة ضمن هذه البنية. إن لتدخل العنصر البشري في فهرسة البيانات فائدة مهمة تتجلى في اختيار المصادر المفهرسة وفي تصنيف المواقع

الفضلى. لكن لتدخل العنصر البشري سيئة أيضا تتمثل في التغيير المستمر الذي يشهده الـوب ويصبح من المستحيل، مع سرعة التغيير الكبيرة، أن يبقى موقع ما في المقدمة دائما.

نعرض فيما يلي أهم مواقع الفهرسة على الوب حاليا:

آ– موقع الفهرسة: **Yahoo!**

ب– موقع الفهرسة: مكتبة الوب الافتراضية (WWW Virtual Library)

محركات البحث

تتميز محركات البحث (Search Engines) بالتقليل من تدخل العنصر البشري في عملها حيث يكـون هذا التدخل في حده الأدنى. فهي تشغل ما يعرف باسم العناكب (spiders) أو الروبوتـات (Robots) أو الجوالين (Wanderers) أو الديدان (Worms) التي تعمل بصمـت في الخلفيـة. تقـوم هـذه الـبرامج بالولوج إلى المواقع المنتشرة على الوب من خلال تتبع الوصلات التشعبية. عند الوصول إلى صفحة وب جديدة فإنها تقوم بقراءة محتوياتها ثم تبويبها وحفظ المعلومات لتنقلها لاحقا إلى قاعدة بينات محرك البحث.

عندما يستخدم المستثمر محرك بحث فإنه في الواقع يبحث ضمن قاعدة بيانات بنيـت مـن قبـل هـذه البرامج الجامعة للمعلومات. وبالتالي فإن المعومات التي قد يحصل عليها المستثمر من خلال محركـات البحث تكون أكثر تعبيرا عن الواقع الحالي لشبكة إنـترنت بفضـل العمليـة المؤتمتـة لتحديـث قـاعدة بياناتها.

يمكن أن نذكر إذن الخصائص التالية لمحركات البحث:

1. تحتوي نصوصا كاملة لصفحات وب منتقاة.
2. يمكن البحث فيها باستخدام الكلمات المفتاحية، وهي تحاول مطابقـة هـذه الكلمـات بدقـة ضمـن الصفحة.
3. لا تملك فئات بحسب المواضيع.
4. تقوم برامج خاصة تدعى العناكب spiders بملء قـاعدة بياناتـها ويكـون التدخـل البشـري بحـده الأدنى.
5. تختلف حجومها من محركات البحث الصغيرة المتخصصة وحتى محركات بحث تفهرس أكثر من 90 بالمائة من الوب.

نعرض فيما يلي لأهم محركات البحث على شبكة الوب

آ- محرك البحث ليكوس Lycos

ب- محرك البحث إنفوسيك Infoseek

محركات البحث البينية

محركات البحث البينية (Meta Search Engines) هي محركات بحث ناتجة عن محركات البحث الحقيقية. يبحث محرك البحث البيني ضمن عدة محركات بحث حقيقية ثم يدمج جميع الإجابات الواردة من محركات البحث ضمن صفحة واحدة.

وهي تتصف بالخصائص التالية:

1. لا تملك قاعدة بيانات خاصة بها.
2. تبحث بسرعة ودون تعمق ضمن عدة محركات بحث في آن واحد.
3. تعيد النتيجة ضمن صفحة واحدة.
4. تأخذ 10٪ من نتائج البحث في أي من محركات البحث التي تستخدمها.

فيما يلي أهم محركات البحث البينية:

آ- محرك البحث البيني Metacrawler

ب- عناوين أخرى

http://www.dogpile.com/

http://www.metafind.com/

http://www.puresearch.com/

الفصل الأول : بدايتك مع لغة php

نبذة سريعة عن لغة PHP :

هي لغة حرة مفتوحة المصدر ومجانية الإستخدام و مخصصة لتطوير تطبيقات الويب وبيئة تطويرها هي Linux

إن كانت لك سابقة عهود مع أي لغة برمجة لن تجد الأمر غريب لأن المنطق البرمجي واحد وأوامرها تشبه إلى حد كبير أمها لغة. C

إن كنت من مستخدمي أحد اللغات التالية وهي java أو C أو C++ أو C# ستجد مرونة كبيرة توفرها هذه اللغة في التعامل بخلاف ما إعتدت عليه .

أول ما سيصادفك من هذه المرونة أن هذه اللغة لا تحتاج لتعريف متغيرات فقط إسناد القيمة للمتغير وسيقوم مترجم اللغة بالتعرف على القيمة التي يحويها المتغير تلقائياً .

> **ملاحظة :** أوامر لغة PHP غير حساسة لحالة الأحرف بمعنى يمكنك الكتابة بالأحرف الكبيرة أو الصغيرة على حد سواء في أوامر اللغة

وبما أن صفحة الويب يمكن أن تتضمن أكواد غير أكواد لغة PHP إذاً يجب تنبية المترجم أين أكواد PHP ليتم التعرف عليها ولهذا عند كتابة أكواد PHP داخل الصفحة يجب تضمينها ضمن وسم الفتح php?> ووسم الإغلاق ?> هناك أيضاً الشكل المختصر ولكن تم إيقاف إستخدامه لتشغيله يجب عليك التعديل على ملف php.ini

وكما جرى العرف والعادة طباعة جملة إفتتاحية وغالباً تكون ! hello world

للطباعة على المتصفح نستخدم echo بأقواس أو بدون أقواس كالتالي :

```
<?php
   echo ("hello ");
   echo "world !!";
?>
```

ضع هذا الكود في ملف وليكن باسم test.php ونفذ الكود عن طريق كتابة رابط الصفحة في نافذة المُتصفح .

> **ملاحظة :** يُسمح لك باستخدام المسافات الفارغة و الأسطر كيفما تشاء ولكن يجب أن يتم الفصل بين الأوامر البرمجية بالفاصلة المنقوطة ";"

يمكننا تطبيق وسوم الـ HTML وطباعتها كالتالي :

```
<?php
   echo "<div style='color:#F00;'>hello world !!</div>";
?>
```

وقد قمت بإستبدال علامة الإقتباس المزدوجة إلى مفردة حتى لا يحدث تضارب بين العلامتين ويمكن أن يكون الكود أيضاً بهذا الشكل :

```
<?php
   echo '<div style="color:#F00;">hello world !!</div>';
?>
```

وسيأتي الحديث عن الفرق بين الطريقتين لاحقاً .

ويمكن أيضاً إستخدام العلامة \ قبل العلامة التي لا تريد أن يحدث لها تضارب مع علامة أخرى بهذا الشكل :

```
<?php
   echo "<div style=\"color:#F00;\">hello world !!</div>";
?>
```

للدمج نستخدم علامة النقطة . كالتالي :

```
<?php
   echo "hello"." world !!";
?>
```

التعليقات في أكواد php :

- تستخدم العلامتين // أو العلامة # لإضافة تعليق سطر واحد ويمكنك إستخدام بداية التعليق بالرمز*/ وإنتهائه بالرمز /* لحصر ما بينهما

```
<?php
// تعليق سطر واحد
# تعليق سطر واحد
/* حصر التعليق */
/*
حصر تعليق أكثر من سطر
حصر تعليق أكثر من سطر
*/
?>
```

المتغيرات :

- فقط ما نحتاجه لتعريف متغير في لغة php هو أن يسبق اسم المتغير العلامة $ ولا يشترط أن تضع للمتغير قيمة عند بداية التعريف ولكن لايصح إستخدامه قبل تعيين قيمه له ويتم التعرف على نوع البيانات المسندة للمتغير تلقائياً

- تسمية المتغيرات تتبع القواعد العامة بأن يبدأ اسم المتغير بحرف من حروف اللغة الإنجليزية أو من 127 إلى 255 من جدول ASCII ولا يحتوي غير الحروف الإنجليزية والأرقام والعلامة _ ومن 127 إلى 255 من جدول الـ ASCII على هذا يمكن إستخدام اللغة العربية في تسمية المتغيرات .

> **ملاحظة :** من 127 إلى 255 من جدول ASCII تكون مخصصة لرموز اللغة الحالية المستخدمه على الجهاز .

تسمية المتغيرات حساسة لحالة الأحرف أي إستخدامك حرف كبير غير إستخدامك لحرف صغير

والتالي تعريف متغيرات مختلفه تحمل قيم مختلفة :

```
<?php
$var1;                          // عدم اسناد قيمة إبتدائية للمتغير
$var2 = 10;                     // اسناد عدد صحيح
$var3 = 10.23;                  // اسناد عدد كسري
$var4 = null;                   // اسناد القيمة الفارغة
$var5 = false;                  // اسناد قيمة منطقية
$var6 = "Mahmoud";              // اسناد سلسة نصية
$var7 = 'Mostafa';              // اسناد سلسة نصية
$var1 = $var7;                  // $var1 الى المتغير $var7 اسنتد قيمة المتغير
$_ = $var6.$var2;               // دمج متغير بمتغير واسناد القيمة المدمجة لمتغير أخر
$_20 = $var1.$var3;             // دمج متغير بمتغير واسناد القيمة المدمجة لمتغير أخر

// طباعة المتغيرات معاً
echo $var1.$var2.$var3.$var4.$var5.$var6 $var7.$_.$_20;
?>
```

- هناك قيم أخرى يمكن إسنادها للمتغير سنتعرف عليها لاحقا كالمصفوفات والكائنات و العنوان

> **ملاحظة :** القيمة المنطقية false والقيمة الفارغة null لاتظهر في الطباعة والقيمة المنطقية true يطبع عوضاً عنها 1

العمليات الحسابية :

يمكن للغة php كغيرها من لغات البرمجة القيام بمختلف العمليات الحسابية على الأعداد و من هذه العمليات البسيطة الجمع وذلك باستخدام الرمز + و الطرح باستخدام الرمز - و الضرب * و القسمة / و علامة باقي القسمة % . أمثلة على العمليات الحسابية :

```
<?php
$var1 = 10;
// اسناد عدد صحيح
$var2 = 20.23;       // اسناد عدد كسري
$var3 = $var1*$var2;    // عملية ضرب متغيرين
$var4 = $var1/$var2;     // عملية قسمة متغيرين
$var5 = $var1%$var2;           // عملية باقي القسمة
echo '$var1+$var2 = '.($var1+$var2).'<br>';        // طباعة ناتج عملية الجمع وطباعة اسماء المتغيرات
echo "$var1+$var2 = ".($var1+$var2).'<br>';       // طباعة ناتج عملية الجمع وطباعة قيم المتغيرات
echo $var3.'<br>'.$var4."<br>".$var5;              // طباعة باقي المتغيرات
?>
```

المثال السابق يوضح الفرق بين إستخدام علامة الإقتباس المزدوجة والمفردة حيث أن السلسة النصية بين علامتي إقتباس مزدوجتين إذا كان بها اسم متغير يتم طباعة قيمته ولكن في حالة علامتي الإقتباس المفردتين يتم طباعة اسم المتغير وليس قيمته .

من المعروف أن العمليات الحسابية تتم على المتغيرات العددية فقط فهل لغة PHP تتبع هذا النمط كباقي اللغات وتصدر أخطاء عند مخالفة هذا الأمر ؟ حاول تجربة المثال التالي :

```
<?php
$var1 = 30;
$var2 = '10user1';       // سلسة نصية تبدأ برقم
$var3 = 'a120';          // سلسة نصية تبدأ بحرف
```

```
$var4 = true;
$var5 = false;
$var6 = null;
$var7 = '20a60';          // قيمة نصية بها أعداد وحروف
$var8 = '20.13hhr60.12';

echo "$var1+$var2 = ".($var1+$var2).'<br>';
echo "$var1+$var3 = ".($var1+$var3).'<br>';
echo "$var1+$var4 = ".($var1+$var4).'<br>';
echo "$var1+$var5 = ".($var1+$var5).'<br>';
echo "$var1+$var6 = ".($var1+$var6).'<br>';
echo "$var1+$var7 = ".($var1+$var7).'<br>';
echo "$var1+$var8 = ".($var1+$var8).'<br>';
?>
```

- نفذ المثال السابق ولاحظ النتيجة إن لم تستسغ الأمر يمكنك استخدام معاملات التحويل التالية :

int	لتحويل نوع المتغير الى أرقام
double	لتحويل نوع المتغير الى عدد ذو فاصلة عائمة
float	لتحويل نوع المتغير الى عدد طويل
boolean , bool	لتحويل نوع المتغير الى قيمة منطقية
string	لتحويل نوع المتغير الى سلسلة نصية

بالنسبة لـ bool و boolean العمل واحد وأيضاً float و double والمثال التالي يوضح العملية :

```
<?php
$var1 = 10;
$var2 = 20.12;
$var3 = '11234567891234567891234567891234567user1';
$var4 = 'user110';
$var5 = '12.123456789123456789user1';

echo "(double)$var1 = ".(double)$var1."<br>";
echo "(int)$var2 = ".(int)$var2."<br>";
echo "(string)$var1 = ".(string)$var1."<br>";
echo "(string)$var2 = ".(string)$var2."<br>";
echo "(int)$var3 = ".(int)$var3."<br>";
echo "(double)$var3 = ".(double)$var3."<br>";
echo "(int)$var4 = ".(int)$var4."<br>";
echo "(double)$var4 = ".(double)$var4."<br>";
echo "(int)$var5 = ".(int)$var5."<br>";
echo "(double)$var5 = ".(double)$var5."<br>";
echo "(int)null = ".(int)null ."<br>";
echo "(double)null = ".(double)null ."<br>";
echo "(int)false = ".(int)false ."<br>";
echo "(double)false = ".(double)false ."<br>";
echo "(int)true = ".(int)true ."<br>";
echo "(double)true = ".(double)true ."<br>";

?>
```

والمثال التالي يوضح عملية التحويل للقيم المنطقية (وسيأتي ذكر هذه الجزئية بتفصيل بعد حالة الشرط if لاحقاً) :

```
<?php

echo "(bool) = ".(bool)'' ."<br>";
echo "(bool)0 = ".(bool)0 ."<br>";
echo "(bool)'0' = ".(bool)'0' ."<br>";
echo "(bool)12 = ".(bool)12 ."<br>";
echo "(bool)-10 = ".(bool)-10 ."<br>";
echo "(bool)'-100' = ".(bool)'-100' ."<br>";
echo "(bool)12.12 = ".(bool)12.12 ."<br>";
echo "(bool)-13.12 = ".(bool)-13.12 ."<br>";
echo "(bool)12.12user1 = ".(bool)'12.12user1' ."<br>";
echo "(bool)user112.12 = ".(bool)'user112.12' ."<br>";
echo "(int)((bool)0) = ".(int)((bool)0) ."<br>";

?>
```

وبإختصار السلسة النصية إذا كانت فارغة فهي تعني false وإن كان بها قيمة أياً كانت فهي تعني true حتى بدون عملية تحويل وسنرى هذا عند حديثنا عن الشروط , وأيضاً الصفر أو 0.0 يعني false وبخلاف ذلك سواء عدد صحيح أو كسري أو عدد سالب فهو يعني true .

معاملات الزيادة والنقصان :

++ معامل الزيادة

-- معمل النقصان

ففي حالة كونه قبل المتغير أي يُزاد أو يُنقص من قيمة المتغير قبل تنفيذ الكود البرمجي بمقدار واحد ولكن في حالة كونه بعد المتغير ينفذ الكود البرمجي الموجود به ومن ثَم زيادة أو نقصان المتغير بمقدار الواحد والكود التالي يوضح العملية :

```
<?php
$var1 = 0;
$var2 = 0;
```

```
$var3 = 0;
$var4 = 0;
echo '++$var1 = '.(++$var1);
echo '<br>';
echo '$var1 = '.$var1;
echo '<br>';
echo '$var2++ = '.$var2++;
echo '<br>';
echo '$var2 = '.$var2;
echo '<br>';
echo '--$var3 = '.--$var3;
echo '<br>';
echo '$var3 = '.$var3;
echo '<br>';
echo '$var4-- = '.$var4--;
echo '<br>';
echo '$var4 = '.$var4;
?>
```

معاملات العمليات :

+=	جمع قيمة على قيمة المتغير السابقة
-=	طرح قيمة من قيمة المتغير السابقة
/=	قسمة قيمة المتغير السابقة على قيمة
*=	ضرب قيمة في في قيمة المتغير السابقة

%=	إيجاد الباقي لقيمة المتغير السابقة على قيمة
.=	دمج قيمة إلى قيمة المتغير السابقة

والتالي يوضح العملية :

$var1 += $var2; تساوي $var1 = $var1 + $var2;

$var1 -= $var2; تساوي $var1 = $var1 - $var2;

$var1 *= $var2; تساوي $var1 = $var1 * $var2;

$var1 /= $var2; تساوي $var1 = $var1 / $var2;

$var1 %= $var2; تساوي $var1 = $var1 % $var2;

$var1 .= $var2; تساوي $var1 = $var1 . $var2;

مثال على ما سبق :

```
<?php
$var1 = 10;
$var2 = 10;
$var3 = 10;
$var4 = 10;
$var5 = 10;
$var6 = 10;

$var1 += 10;
$var2 -= 10;
$var3 *= 10;
$var4 /= 10;
```

```
$var5 %= 10;
$var6 .= 10;
echo $var1.'<br>'.$var2.'<br>'.$var3.'<br>'.$var4.'<br>'.
$var5.'<br>'.$var6.'<br>';
?>
```

هناك طريقتين لكتابة أكواد php و HTML معاً إما استخدام جملة الطباعة أو إغلاق وسم كود php والبدأ في كتابة أكواد HTML ومن ثَم إعادة فتح وسم php لتكملة كتابة أكواد php :

```
<?php
   $var1 = 'value1';
   $var2 = 'value2';
?>
<!DOCTYPE HTML>
<html dir="rtl">
   <head>
   <link rel="stylesheet" type="text/css" href="style.css"/>
   <meta charset="utf-8">
   <title>
      التمرين
   </title>

   </head>
   <body>
      <div style="color:#F00;">
         <?php echo $var1; ?>
      </div>
      <div style="color:#00F; font-size:28px;">
         <?php echo $var2; ?>
      </div>
```

```
   </body>
</html>
```

والطريقة الثانية :

```
<?php
   $var1 = 'value1';
   $var2 = 'value2';
echo '
<!DOCTYPE HTML>
<html dir="rtl">
   <head>
   <link rel="stylesheet" type="text/css" href="style.css"/>
   <meta charset="utf-8">
   <title>
      التمرين
   </title>

   </head>
   <body>
      <div style="color:#F00;">
         '.$var1.'
      </div>
      <div style="color:#00F; font-size:28px;">
         '.$var2.'
      </div>
   </body>
</html>';
?>
```

وكل طريقة تكون مناسبه في وضع أكثر من الطريقة الأُخرى .

> **ملاحظة :** أكواد الـ HTML تعمل ضمن ملف بإمتداد php -ولكن تحتاج لسرفر- والعكس غير صحيح

وحتي لا نُغضب مبرمجي الـ c والـ c++ واللغات الأخرى منا فهناك دوال أخرى للطباعة والقراءه من سلسة نصية و عملها كعمل هذه الدوال في هذه اللغات وهي print و printf و sprintf و sscanf

الفصل الثاني : الثوابت ودوال الشرط والدوران

الثوابت :

يتم تعريف الثوابت بإستخدام الكلمة المحجوزة const قبل اسم الثابت أو من خلال الدالة define ويتبع اسم الثابت قواعد كتابة اسم المتغير ذاتها غير أنه لا يبدأ بالعلامة **$** ويُفضل أن يُكتب بالحروف الكبيرة . ويجب أن يُعطى الثابت قيمة عند عملية تعريفه ولا يمكن تغيير هذه القيمة فيما بعد ,أمثلة لتعريف الثوابت :

```
<?php
const أحمد = "user1";
const AAA = 'user1';
define("BBB","user2");
echo أحمد.AAA.BBB;
?>
```

حالة الشرط if :

وهي أنه في حالة تحقق الشرط يتم تنفيذ الأمر وإلا لا يتم التنفيذ والشرط في النهاية إما أن يكون محقق true أو غير محقق false .الشكل العام لحالة if البسيطة هو :

```
if(/* الشرط */)
   /* الأمر المراد تنفيذه في حالة تحقق الشرط */;
OR
if(/* الشرط */)
{
   // أمر 1
   // أمر 2
   // أمر 3
}
```

ملاحظة : في حالة تحقق شرط جملة if وعدم وجود أقواس يتم تنفيذ الأمر البرمجي بعد if وصولاً لنهاية الأمر البرمجي المنتهي بالفاصلة المنقوطة ;

حالة الشرط if else

وتكون على الصورة :

```
if(/* الشرط */)
{
    // الجمل البرمجية في حالة تحقق الشرط
}
else
{
    // الجمل البرمجية في حالة عدم تحقق الشرط
}
```

حالة الشرط المتعددة else if وتكون على الصورة :

```
if(/* الشرط */)
{
    // الجمل البرمجية هنا
}
else if(/* الشرط */)
{
    // الجمل البرمجية هنا
}
.
.// وهكذا تكرار غير محدود
.
else if(/* الشرط */)
{
```

```
    // الجمل البرمجية هنا
}
else
{
    // الجمل البرمجية هنا
}
```

ولا يشترط كتابة جملة else المفردة في النهاية وأيضاً يمكن الإستغناء عن أقواس المجموعة إذا كان لدينا جملة واحدة داخل المجموعة . أمثلة على جملة if :

```
<?php
    if(true)
    echo "true<br>";

    if(true)
    {
        echo "<h1>true</h1>";
        echo "<h1>inside if</h1>";
    }

    if(false) echo "false<br>";

    if(false)
        echo "<h2>false</h2>";
        echo "outside if";
?>
```

في حالة true الشرط محقق دائماً أما في حالة false فالشرط غير محقق دائماً

أمثلة إستخدام if مع أنواع البيانات المختلفة وكما بيَّنا في الفصل السابق أن أي عدد بخلاف الصفر فهو يعبر عن القيمة true وأن أي سلسة نصية بخلاف السلسة النصية الفارغة فهي أيضاً

تعبر عن القيمة true والمثال التالي يوضح هذا :

```
<?php
   if(0)
      echo "<h3>0 true</h3>";
   else
      echo "<h3>0 false</h3>";

   if(13)
      echo "<h3>13 true</h3>";
   else
      echo "<h3>13 false</h3>";

   if(-50)
      echo "<h3>-50 true</h3>";
   else
      echo "<h3>-50 false</h3>";

   if(null)
      echo "<h3>null true</h3>";
   else
      echo "<h3>null false</h3>";

   if('')
      echo "<h3>'' true</h3>";
   else
      echo "<h3>'' false</h3>";

   if(' ')
      echo "<h3>' ' true</h3>";
```

```
   else
      echo "<h3>' ' false</h3>";

   if('user1')
      echo "<h3>user1 true</h3>";
   else
      echo "<h3>user1 false</h3>";
?>
```

حالة if المختصرة :

حيث يتم وضع الشرط المُراد التحقق من صحته و يليه علامة الإستفهام و يعدها يتم تعريف الكود الواجب تنفيذه إذا كان الشرط مُحققاً و الكود اللازم تنفيذه إذا كان غير مُحقق و يفصل بينهما الرمز ":"

```
condition?true:false;
```

مثال:

```
<?php
   echo true?"yes":"no";
?>
```

العمليات المنطقية :

كما في باقي اللغات يمكن استخدام مُختلف إشارات العمليات المنطقية في حلقات الشرط وهي and أو && التي تعني "و" , or أو || التي تعني "أو" , والإشارة "!" التي تُفيد النفي و العملية المنطقية XOR , الجدول التالي يوضح ذلك :

الشرط	القيمة	الحالة

a and b	true	a$ و b$ كلاً منهما يكون true
$a && $b	true	a$ وb$ كلاً منهما يكون true
$a or $b	true	أي من a$ أوb$ يكون true
$a || $b	true	أي من a$ أوb$ يكون true
!$a	true	a$ يكون false و false في حالةa$ يكون true
$a xor $b	true	أي من a$ أو b$ يكون true ولكن غير متشابهين

ملاحظة : يمكنك إستخدام أي صيغة لعمليتي and و or حيث الطريقتين متكافئتين اي استخدام "and" أو "&&" لا يؤثر أبداً

الكود التالي ينفذ جدول الصواب والخطأ للعمليات المنطقية السابقة :

```
<?php
   echo "AND && <br>------------<br>true and true = ";
   if(true and true)
      echo "true<br>";
   else
      echo "false<br>";

   echo "true and false = ";
   if(true and false)
      echo "true<br>";
   else
      echo "false<br>";
```

```
echo "false and false = ";
if(false and false)
   echo "true<br>";
else
   echo "false<br>";

echo "<br>OR | | <br>------------<br>true or true = ";
if(true or true)
   echo "true<br>";
else
   echo "false<br>";

echo "true or false = ";
if(true or false)
   echo "true<br>";
else
   echo "false<br>";

echo "false or false = ";
if(false or false)
   echo "true<br>";
else
   echo "false<br>";

echo "<br>XOR  <br>------------<br>true xor true = ";
if(true xor true)
   echo "true<br>";
else
   echo "false<br>";
```

```
   echo "true xor false = ";
   if(true xor false)
      echo "true<br>";
   else
      echo "false<br>";

   echo "false xor false = ";
   if(false xor false)
      echo "true<br>";
   else
      echo "false<br>";

   echo "<br>!  <br>------------<br>!true = ";
   if(!true)
      echo "true<br>";
   else
      echo "false<br>";

   echo "!false = ";
   if(!false)
      echo "true<br>";
   else
      echo "false<br>";
?>
```

عمليات المقارنة :

==	يساوي

!=	لا يساوي
>	أكبر من
<	أصغر من
>=	أكبر من أو يساوي
<=	أصغر من أو يساوي
===	مساواة القيم من نفس النوع
!==	عدم مساواة القيم من نفس النوع

أظن أن اغلبهم واضحين ولكن سأوضح عمل المساواة من نفس النوع وعدم المساواة من نفس النوع

- وكما عرفنا في الأعلى أن الصفر مساوي للقيمة false وأي عدد خلاف الصفر مساوي للقيمة true وقيمة السلسة النصية بخلاف السلسة النصية الفارغة مساوية للقيمة true فلهذا لا يصلح أن أستخدم قيم المساواه العادية وكمثال إذا أردت أن أختبر القيمة على أنها false و false فقط إذاً علي إستخدام عملية المساواة من نفس النوع والمثال التالي يوضح العملية :

```
<?php
if('10user1' == 10)
   echo "10user1 == 0 yes<br>";
else
   echo "10user1 == 0 no<br>";
```

```
if('' == 0)
   echo "'' == 0 yes<br>";
else
   echo "'' == 0 no<br>";

if(0 == false)
   echo "0 == false yes<br>";
else
   echo "0 == false no<br>";

if('' == false)
   echo "'' == false yes<br>";
else
   echo "'' == false no<br>";

if(-10 == true)
   echo "-10 == true yes<br>";
else
   echo "-10 == true no<br>";
?>
```

ولكن عند استخدام عمليات المساواه من نفس النوع سيتم التعرف على القيم ومساواتها من نفس نوعها فالمثال السابق يكون على الشكل التالي :

```
<?php
if('10user1' === 10)
   echo "10user1 == 0 yes<br>";
else
   echo "10user1 == 0 no<br>";
if('' === 0)
   echo "'' == 0 yes<br>";
```

```
else
   echo "'' == 0 no<br>";
if(0 === false)
   echo "0 == false yes<br>";
else
   echo "0 == false no<br>";
if('' === false)
   echo "'' == false yes<br>";
else
   echo "'' == false no<br>";

if(-10 === true)
   echo "-10 == true yes<br>";
else
   echo "-10 == true no<br>";
?>
```

التالي مثال على حالة if else المتعددة , فلنفرض أن لدينا قيمة ولتكن مُعرف الصفحة الـ id وعلى أساس قيمته يتم إنشاء إرتباط تشعُبي لصفحات مختلفه فيكون الكود كالتالي :

```
<?php
$id = 200;
if($id == 100)
{
   echo "<h3><a href='page1.php'> go page1 </a></h3>";
}
else if($id == 200)
{
   echo "<h3><a href='page2.php'> go page2 </a></h3>";
```

```
}
else if($id == 400)
{
   echo "<h3><a href='page3.php'> go page3 </a></h3>";
}
else if($id == 500)
{
   echo "<h3><a href='page4.php'> go page4 </a></h3>";
}
else
{
   echo "<h3><a href='index.php'> go home </a></h3>";
}

?>
```

حالة switch case :

- يمكن عمل نفس المثال السابق بإستخدام جملة switch case كالتالي :

```
<?php
$id = 250;
switch($id)
{
   case 100:
      echo "<h3><a href='page1.php'> go page1 </a></h3>";
      break;
   case 200:
      echo "<h3><a href='page3.php'> go page3 </a></h3>";
      break;
```

```
   case 300:
      echo "<h3><a href='page4.php'> go page4 </a></h3>";
      break;
   default:
      echo "<h3><a href='index.php'> go home </a></h3>";

}
?>
```

حيث أن جملة break هي للخروج بعد تنفيذ الأمر

دالة defined للتعرف على الثابت هل هو موجود أم لا وتعيد القيمة true في حالة وجوده وتعيد القيمة false إن لم يكن موجود

دالة isset للتعرف على المتغير هل موجود ومسند له قيمه أم لا وتعيد القيمة true في وجود المتغير ووجود قيمة مسنده له وتعيد القيمة false في حالة عدم وجود المتغير أو عدم وجود قيمة مسنده له أو أن تكون القيمة المسنده للمتغير هي القيمة الفارغة null والمثال التالي يوضح عملهم :

```
<?php
define("AAA","Mostaf ");
const BBB = "Khaled ";
$var1;
$var2 = null;
$var3 = '';

if(defined("AAA"))
   echo AAA;
if(defined("BBB"))
   echo BBB;
if(defined("CCC"))
```

```
   echo CCC;

if(isset($var1))
   echo '<br>$var1 is set';
if(isset($var2))
   echo '<br>$var2 is set';
if(isset($var3))
   echo '<br>$var3 is set';
if(isset($var4))
   echo '<br>$var4 is set';
?>
```

حلقات الدوران :

حلقة الدوران for :

الشكل العام لها كالتالي :

```
<?php
for(/*بداية الحلقة*/;/*شرط التوقف*/;/*معامل الزيادة أو النقصان*/)
{
   /*
       الكود البرمجي المراد تكراره عدد من المرات
   */
}
?>
```

مثال :

```
<?php
for($i=0;$i<10;$i++)
{
   echo '<h3>$i='.$i.'</h3>';
```

```
}
?>
```

أو كتابتها بهذا الشكل إن كانت تعليمه واحدة

```
<?php
for($i=0;$i<10;++$i) echo '<h3>$i='.$i.'</h3>';
?>
```

لتخطي دورة معينة والإنتقال للتليه نستخدم الكلمة المحجوزة continue

مثال :

```
<?php
for($i=0;$i<10;$i++)
{
   if($i == 5) continue;
   echo '<h3>$i='.$i.'</h3>';
}
?>
```

وإن أردنا الخروج من الحلقة نهائياً نستخدم break

مثال :

```
<?php
for($i=0;$i<10;++$i)
{
   if($i == 5) break;
   echo '<h3>$i='.$i.'</h3>';
}
?>
```

حلقة الدوران while :

الصيغة العامة

```
<?php
```

```
while(/*الشرط*/)
{
    /*
        الكود المراد تكراره
    */
}
?>
```

وتعني الدوران في حالة تحقق الشرط وفي حالة عدم تحققه لا يتم الدخول للحلقه

أمثلة :

```
<?php
$count = 0;
while(10)
{
    echo "<h3> Hi </h3>";
}
while(true)
{
    echo "<h3> Hi </h3>";
}
while('user1')
{
    echo "<h3> Hi </h3>";
}
while($count < 10)
{
    echo "<h3> Hi </h3>";
}
while($count != 10)
{
```

```
    echo "<h3> Hi </h3>";
}
?>
```

جميع الحلقات السابقة حلقات غير منتهيه تسبب تجمد المتصفح والضغط على الخادم والسبب أن الشرط محقق دائماً كما نعلم . أمثلة على حلقات صحيحة ومنتهية :

```
<?php
$count = 1;
while($count <= 10)
{
    echo "<h3> Hi </h3>";
    $count++;
}
while(true)
{
    echo "<h3> YES </h3>";
    if($count++ == 20) break;
}
?>
```

حلقة الدوران do while :

وهي نفس حلقة الدوران while ولكن الفرق عنها أنها تنفذ دوران واحد قبل إختبار تحقق الشرط وصيغتها العامة هي :

```
<?php
do
{
    /*
```

```
        الأكواد المراد تكرارها
    */
}while(/*الشرط*/);
?>
```

أمثلة :

```
<?php
do
{
    echo "<h3>Hi</h3>";

}while(false);

$count = 0;
do
{
    echo '<h3>$count = '.++$count.'</h3>';

}while($count < 10);

?>
```

> **ملاحظة :** في كل حلقات التكرار السابقة يمكن استخدام continue لتخطي حلقة أو الخروج نهائياً من الحلقة بإستخدام break

هناك صيغ أخرى لإستخدامها مع الأوامر البرمجية ك if و for و while و switch لإستخدامها بدلاً من الأقواس والصيغ العامة لها كالتالي :

```
<?php
if (/*الشرط*/):
```

```
    /*
        أي عدد من الأوامر البرمجية
    */
endif;

// الحالة المتعدده
if(/*الشرط*/):
    /*
        أي عدد من الأوامر البرمجية
    */
elseif(/*الشرط*/):
    /*
        أي عدد من الأوامر البرمجية
    */
elseif(/*الشرط*/):
    /*
        أي عدد من الأوامر البرمجية
    */
endif;

while(/*الشرط*/):
    /*
        الأوامر البرمجية المراد تكرارها
    */
endwhile;

for(/*أوامر الحلقة*/):
    /*
        الأوامر البرمجية المراد تكرارها
```

```
   */
endfor;

switch(/*القيمة*/):
   case "":
      // ...
      break;
   case "":
      // ...
      break;
   default:
      //...
endswitch;
?>
```

الفصل الثالث : المصفوفات والدوال

المصفوفات :

كما مر معنا في فصلٍ سابق , يمكن للمتغيرات ان تحوي قيمة واحدة فقط , فجاءت المصفوفات لتحل هذا القصور و تُمكن المبرمج من تخزين عدة قيم في متغير واحد يسمى بالمصفوفة , (المصفوفات في البرمجة تختلف اختلافا كليا عن المصفوفات الرياضية) , واذا كنت قد تعاملت مع المصفوفات بلغات برمجة غير php ستجد ان php لها طريقة خاصة ومرونة كبيرة جداً في التعامل مع المصفوفات كما سنرى في سياق هذا الفصل .

المصفوفات تتكون من ما يُعرف بمفتاح أو مُعرف العنصر داخل المصفوفة وهو الـ key أو الـ index للمصفوفة ويبدأ من 0 إلى أقل من عدد عناصر المصفوفة بمقدار واحد (إن العد يبدأ من الصفر) وكل عنصر من عناصر المصفوفة يحتوي على قيمة مرتبطة بهذا المفتاح , في php يمكن أن تكون هذه القيمة أي نوع من أنواع البيانات سواء عدد صحيح أو كسري أو قيمة منطقية أو القيمة الفارغة أو مصفوفة أو كائن .

لتخزين قيم ما على شكل مصفوفة عليك فقط أن تضع الأقواس المربعة [] بعد اسم المتغير وتقوم بإسناد القيم للمصفوفة كالتالي :

```
<?php
$myArr[] = 10;            //key = 0 , value = 10
$myArr[] = 12.16;         //key = 1 , value = 12.16
$myArr[] = true;          //key = 2 , value = true
$myArr[] = "username";    //key = 3 , value = "username"
$myArr[] = 'password';        //key = 4 , value = 'password'

for($i = 0; $i < 5; $i++)
{
   echo '<h3>'.$myArr[$i].'</h3>';
}
?>
```

واضح من الكود السابق أنه بإمكاننا تخزين أنواع مختلفة من البيانات داخل المصفوفات سواءً أكانت نص أم رقم أم رقم ذو فاصلة عشرية ...
ويمكن أيضاً تخزين القيم في المصفوفة بالشكل المعتاد كما في أغلب لغات البرمجة , وفي حال أردنا طباعة قيمة المصفوفة داخل علامتي الإقتباس يجب وضعها بين قوسين {} كالتالي :

```
<?php
$myArr[0] = 10;
$myArr[1] = 12.16;
$myArr[2] = true;
$myArr[3] = "username";
$myArr[4] = 'password';

for($i = 0; $i < 5; $i++)
{
   echo "<h3>{$myArr[$i]}</h3>";
}
?>
```

تم استخدام حلقة التكرار for لاخراج عناصر المصفوفة حيث i تتدرج من الصفر وحتى عدد عناصر المصفوفة ناقص واحد (حيث i<5 تكافئ i<=4) .
ولإعطاء قيم للمصفوفة عند تعريفها دفعة واحدة نستخدم الكلمة المحجوزة array وتوضع العناصر بين قوسين ويفصل بينها فاصلة ',' كالتالي:

```
<?php
$myArr = array(10, 12.16, true, "username", 'password');

for($i = 0; $i < 5; $i++)
{
  echo "<h3>{$myArr[$i]}</h3>";
```

```
}
?>
```

و لمعرفة عدد عناصر المصفوفة نستخدم الدالة count , التي تقبل وسيطا واحدا هو المصفوفة المُراد معرفة عدد عناصرها , وتُعيد عدد عناصر المصفوفة , كما في المثال التالي :

```
<?php
$myArr = array(10, 12.16, true, "username", 'password');

for($i = 0; $i < count($myArr); $i++)
{
   echo "<h3>{$myArr[$i]}</h3>";
}
?>
```

المصفوفات المتعددة الابعاد :

كما ذكرنا سابقا , يمكن ان يكون اي عنصر من عناصر مصفوفة من أي نوع من البيانات , فإذا كانت قيمة هذا العنصر مصفوفة حصلنا على مصفوفة متعددة الابعاد .

ويمكن تمثيل المصفوفات متعددة الابعاد على أنها مصفوفات أحادية متداخلة والتالي تمثيل مصفوفة 2X3 :

```
<?php
$myArr[0][] = "username";
$myArr[0][] = "password";
$myArr[0][] = 10;
$myArr[1][] = 12;
$myArr[1][] = 45.99;
$myArr[1][] = true;
for($i = 0; $i < count($myArr); $i++)
{
```

```
  for($j = 0; $j < count($myArr[$i]); $j++)
  {
    echo "<h3>{$myArr[$i][$j]}</h3>";
  }
}
?>
```

ولإسناد القيم دفعة واحدة عند تعريف المتغير يكون كالتالي :

```
<?php
$myArr = array(
       array('username', "password", 10),
       array(12, 45.99, true)
      );
for($i = 0; $i < count($myArr); $i++)
{
   for($j = 0; $j < count($myArr[$i]); $j++)
   {
      echo "<h3>{$myArr[$i][$j]}</h3>";
   }
}
?>
```

هكذا يمكن إسناد مصفوفات داخل مصفوفات بأي عدد من الابعاد تُريد , أي يُمكن انشاء مصفوفات ذات عشر أبعاد , لكن لا يُمكن التعامل معها بسهولة (هذا اذا امكن التعامل معها اساساً)

المصفوفات المترابطة :

وتكون باستخدام سلاسل نصية للـ key للمصفوفة بدلاً من الأرقام حيث كل عنصر في المصفوفة يتألف من قسمين : الأول هو المفتاح key والثاني هو القيمة value :

```
<?php
//وضعنا فراغات في بعض عناصر المصفوفة لكي لا تظهر الكلمات ملتصقة ببعضها البعض
```

```
$myArr['name'] = 'name ';
$myArr['age']  = 30;
$myArr['city'] = ' city ';
$myArr['phone']= 125668522;
echo $myArr['name'].$myArr['age'].$myArr['city'].$myArr['phone'];
?>
```

كما يمكن أن يكون مُعرف القيم سلاسل النصية وترقيم الرقمي معاً للمصفوفة في مصفوفة واحدة كما سنرى , وهناك دوال مهمة لعرض محتويات وبيانات المتغيرات و المصفوفات والكائنات وهي var_dump و print_r و var_export سنستخدمها لعرض لطباعة محتويات المصفوفة من القيم والـ key لكل قيمة , وتقبل - هذه الدوال - وسيطا واحداً هو المصفوفة المُراد طباعتها , كما في المثال التالي :

```
<?php
$myArr['name'] = 'username 1';
$myArr[]       = "username 2";
$myArr['age']  = 30;
$myArr[]       = 40;
$myArr['city'] = 'luxor';
$myArr[]       = 'Cairo';
$myArr['phone']= 125668522;
$myArr[]       = 124559587;

echo var_export($myArr);
?>
```

ولإسناد القيم من هذا النوع من المصفوفات عند التعريف دفعة واحدة يكون كالتالي :

```
<?php
$myArr = array('name' => 'username 1', 'city' => 'luxor', 'phone'
=> 125668522);
```

```
echo var_export($myArr);
?>
```

دالة foreach للدوران على عناصر المصفوفة :

من أفضل الطرق للدوران على عناصر المصفوفة وبالأخص المصفوفات المترابطة هو إستخدام دالة foreach, ويمكن من خلالها إستخراج القيمة أو القيمة و المُعرف (المفتاح) والشكل العام لها هو :

```
foreach ($array as $key => $value)
{
  //$key هو مفتاح المصفوفة
  //$value هي القيمة المرتبطة بالمفتاح
}
```

والمثال التالي يوضح فكرة عملها :

```
<?php
$myArr = array('name' => 'username 1', 'city' => 'luxor', 'phone'
=> 125668522);
foreach($myArr as $value)
{
  //استخدام الدالة foreach للحصول على القيمة فقط
  echo "<h3>$value</h3>";
}

foreach($myArr as $key=>$value)
{
  //الحصول على المفتاح (المُعرف) و القيمة
  echo "<h3>$key : $value</h3>";
}
?>
```

مثال آخر :

```
<?php
$myArr = array('name' => 'username 1', 'username 2', 'city' =>
'luxor', 'phone' => 125668522, 'Ciro', 125885465);
foreach($myArr as $key => $value)
{
   echo "<h3>$key : $value</h3>";
}
?>
```

دوال التحكم بالمصفوفات

يوجد عدة دوال لاجراء العمليات المختلفة على المصفوفات (تقسيم مصفوفة لعدة اجزاء, ترتيب مصفوفة, عكس مصفوفة ....الخ) وسيتم شرح اشهر تلك الدوال :

الدالة explode :

تقوم هذه الدالة بتقطيع نص وتحويله الى مصفوفة حيث تقبل وسيطين اجباريين الوسيط الاول هو "الفاصل" الذي عنده يتم اقتطاع الجملة و الوسيط الثاني هو النص , لازالة الغموض سوف نأخذ مثالا بسيطا : بفرض اننا نريد ان نجعل كل كلمة في جملة معينة عنصرا من عناصر مصفوفة وبالتالي يكون الفاصل هو "الفراغ" كما في الكود التالي :

```
<?php
$string = 'this is a sting';
$array = explode(' ', $string);
print_r($array);
?>
```

تُستخدم هذه الدالة بكثرة عند القراءة من الملفات النصية كما سنجد في الفصل التاسع

الدالة implode :

تقوم هذه الدالة - تقريباً- بعكس عمل الدالة explode , أي انها تقوم بتحويل عناصر مصفوفة الى الى نص يفصل بينها "فاصل" :

```
<?php
```

```
$string = implode ($glue, $pieces);
?>
```

حيث الوسيط الاول هو الفاصل و الوسيط الثاني هو المصفوفة المُراد تحويل جميع عناصرها الى سلسلة نصية , جرب المثال التالي لتعرف مزيداً عن عمل هذه الدالة :

```
<?php
$array = array(10, 12.16, true, "username", 'password');
$string = implode(' -- ', $array);
echo $string;
#outputs : 10 -- 12.16 -- 1 -- username --  password
?>
```

الدالة is_array :

تقوم هذه الدالة بالتحقق من ان الوسيط المُمرر لها هو مصفوفة وذلك باعادة القيمة true او false :

```
<?php
$string = 'this is a sting';
$array = explode(' ', $string);
echo is_array($array);
//this will output '1'
?>
```

إضافة قيمة الى المصفوفة :

كما مر معنا سابقا يمكن اضافة عنصر جديد بواسطة القوسين [] كالتالي :

```
<?php
$array = array('sy', 'eg', 'lb');
echo 'the array is : <br>';
print_r($array);
$array[] = 'sa';
echo '<br>the array after adding sa is :<br>';
```

```
print_r($array);
?>
```

او باستخدام الدالة array_push حيث تقبل وسيطين الاول هو المصفوفة الهدف والثاني هو القيمة المراد اضافتها ,نعدل الملف السابق كي يستخدم الدالة array_push

```
<?php
$array = array('sy', 'eg', 'lb');
echo 'the array is : <br>';
print_r($array);

#$array[] = 'sa';        this line is repalced by :
array_push($array, 'sa');
echo '<br>the array after adding sa is :<br>';
print_r($array);
?>
```

البحث داخل المصفوفات :

نستخدم الدالة in_array للبحث داخل المصفوفة عن قيمة معينة, هذه الدالة تعيد true في حال نجاحها:

```
<?php
$array = array('sy', 'eg', 'lb', 'sa');
if(in_array('sa', $array) == true)
{
  echo ' sa is found in $array array <br>';
}

if(in_array('fr', $array) == false)
{
```

```
  echo ' fr is NOT found in $array array <br>';
}
?>
```

حيث الوسيط الاول هو القيمة المُراد البحث عنها والوسيط الثاني هو المصفوفة الهدف .

قلب مصفوفة :

حيث تستخدم الدالة array_reverse لقلب ترتيب مصفوفة اي جعل اول عنصر اخر عنصر و هكذا , المثال التالي يوضح الفكرة :

```
<?php
$array = array('1', '2', '3', '4');
$new_array = array_reverse($array);
print_r($new_array); #outputs : Array ( [0] => 4 [1] => 3 [2] => 2
[3] => 1 )
?>
```

الدالة array_unique :

تقوم الدالة array_unique بإزالة أي قيمة تتكرر في المصفوفة , حيث تعيد مصفوفة جديدة بدن أي عناصر مكررة:

```
<?php
$array = array('sy', 'eg', 'lb', 'sy', 'lb', 'sa');
$new_array = array_unique($array);
echo 'the first array is : ';
print_r($array);    # Array ( [0] => sy [1] => eg [2] => lb [3] =>
sy [4] => lb [5] => sa )
echo '<br> the "unique" one : ';
print_r($new_array);    # Array ( [0] => sy [1] => eg [2] => lb
[5] => sa )
```

```
?>
```

لاحظ مفاتيح المصفوفة الثانية .

ترتيب عناصر المصفوفة :

يتم ذلك بواسطة الدالتين sort و asort , حيث تقوم الدالة sort بترتيب عناصر مصفوفة تصاعدياً , شكلها العام كالتالي :

```
sort($array);
```

حيث لا تعيد هذه الدالة أي قيمة , أي تقوم بتعديل المصفوفة مباشرة . الوسيط الأول هو المصفوفة المُراد ترتيب عناصرها .

```
<?php
$array = array(123, 1, 12, 'name' => 'sy', 'eg');
print_r($array);
sort($array);
echo '<br>';
print_r($array);
?>
```

لاحظ أن المصفوفة المرتبة لا تحتفظ بمفاتيح المصفوفة الاصلية , وللاحتفاظ بها نستخدم الدالة asort التي تقوم بنفس عمل sort لكنها تحتفظ بقيم المفاتيح أو المُعرفات :

```
<?php
$array = array(123, 1, 12, 'name' => 'sy', 'eg');
print_r($array);
asort($array);
echo '<br>';
print_r($array);
?>
```

الدوال :

تُوفر php عددا كبيرا من الدوال يتجاوز عددها الالف دالة , ناهيك عن العدد الضخم من المكتبات الاخرى التي تقوم بعدد لا باس به من العمليات , لكن بشكل أو باخر ستحتاج الى دالة جديدة تقوم بمهمة معينة لبرنامجك . الشكل العام لتعريف الدالة هو :

```
function functionName(/*وسائط الدالة*/)
{
   /*
       جسم الدالة
   */
}
```

وسائط الدالة

يمكنها أن تقبل أي نوع من البيانات , وكذلك يمكنها اعادة بأي نوع من البيانات أو عدم الرجوع بأي قيمة , الدالة التالية دالة لا تأخذ أي وسائط ولا تُعيد أي قيمة فقط تطبع جملة على المتصفح , ولتشغيل هذه الدالة علينا استدعائها بكتابة اسمها و من ثم قوسين () كما يلي:

```
<?php
function f_echo()
{
   echo "<h1>PHP:hypertext processor</h1>";
}
f_echo();
?>
```

أما الدالة التالية فهي تأخذ وسيطا لتقوم بطباعته ضمن وسمي h1 , لاحظ ان المتغير $in هو متغير محلي مُعرف داخل الدالة فقط ولا علاقة له مع المتغير $in خارج الدالة

```
<?php
$in = 'username1';
function f_echo($in)
```

```
{
   echo "<h1>$in</h1>";
}

f_echo(10);
f_echo(12.332);
f_echo('username2');
f_echo(true);
//f_echo(array(10,20,30));
?>
```

والكود الأخير الموجود في التعليق هو محاولة تمرير مصفوفة لطباعتها , لكن لو نفذت هذا السطر سيتم توليد خطأ , لأن الدالة تحتوي على بيانات داخلها ولا نستطيع طباعتها مباشرة .
الدالة التالية تأخذ وسيطين وتُعيد حاصل الجمع :

```
<?php
function sum($var1, $var2)
{
   return $var1 + $var2;
}
echo sum(10, 20);
?>
```

كما يمكن تمرير الدوال لبعضها البعض كالتالي :

```
<?php
f_echo(sum(10,20));
function f_echo($in)
{
   echo "<h1>$in</h1>";
```

```
}
function sum($var1,$var2)
{
   return $var1+$var2;
}
?>
```

القيم الافتراضية للوسائط :

في بعض الاحيان يكون للدوال وسائط اختيارية حيث يتم وضع قيمة افتراضية لها , فإذا لم يتم تحديد قيمة الوسيط , فسيتم اخذ القيمة الافتراضية بدلا عنه , ويجب ان تكون جميع الوسائط بعد الوسيط الافتراضي افتراضية , اي لا يجوز ان تعريف الدالة بالشكل التالي :

```
<?php
function function_name ($var1 = 'value', $var2)
{

}
?>
```

مثال عن الاستخدام الصحيح :

```
<?php
function f_echo($in = "text")
{
   echo "<h1>$in</h1>";
}
f_echo();
?>
```

سيتم طباعة text بسبب عدم اعطاء اي وسائط للدالة .

اعادة اكثر من قيمة من الدالة :

كما تلاحظ لا يجوز ان تعيد الدالة الواحدة اكثر من قيمة , لكن يمكن تجاوز هذه المشكلة

باستخدام المصفوفات حيث يتم اعادة مصفوفة تكون عناصرها هي القيم المطلوبة :

```
<?php
function math($x)
{
   return array($x * $x, log($x));
}
print_r(math(23));
?>
```

تمرير الوسائط بمرجعياتها :

في بعض الاحيان , نحتاج الى تعديل قيمة الوسيط مباشرة في الدالة عوضا عن ارجاع قيمة منها واسنادها الى متغير , لجعل الوسائط تُمرر الى دالة بمرجعيتها (By Reference) يجب استخدام الرمز & قبل اسم الوسيط عند تعريف الدالة :

```
<?php
function sum(&$var1, $var2)
{
  $value1 = $var1 + $var2;
  //the same as $var1+= $var2;
}

$num1 = 10;
$num2 = 15;
echo $num1;
//outputs 10
echo '<br>';
sum($num1, $num2);
echo $num1;
```

```
//outputs 25
?>
```

الفصل التاسع : التعامل مع الملفات و المجلدات

من الصعب برمجة تطبيق ويب دون التفاعل مع أي مصدر خارجي كقواعد البيانات أو الملفات و خصوصاً انشاء الملفات و المجلدات و حذفها و تعديلها ...

أولاً : التعامل مع الملفات

المسار هو طريقة للتعبير عن عنوان ملف أو مجلد في نظام التشغيل ,و المسارات نوعان : مسارات نسبية ومسارات مطلقة , المسارات النسبية تبدأ من المسار الحالي حتى نصل الى القيد المطلوب (القيد = مجلد أو ملف) مثلا مسار الملف file1.txt الموجود في المجلد folder الموجود في مجلد البرنامج الذي نقوم بتنفيذه يكون كالتالي :

```
folder/file1.txt
```

اما إذا كان الملف file1.txt موجوداً في المجلد الأب للمجلد التالي (أي المجلد الذي يسبقه) يكون المسار كالتالي :

```
../file1.txt
```

أي ان النقطتين تشيران الى أن الملف المطلوب في المجلد الأب للمجلد الحالي , ويوجد أيضا النقطة الواحدة "." التي تشير الى المجلد الحالي حيث يمكن استبدال المسار الأول كما يلي :

```
./folder/file1.txt
```

اما الروابط المطلقة فهي تُشير الى مسار الملف أياً كان المجلد الذي يوجد فيه البرنامج .

للحصول على المسار كاملاً نستخدم الدالة realpath التي تقبل وسيطاً وحيداً هو المسار النسبي للملف :

```
<?php
echo realpath('file1.txt');
?>
```

وفي حال لم يُحدد الوسيط فيستم اعادة المسار المطلق للمجلد الحالي .

التأكد من وجود ملف :

في بعض الأحيان يلزم معرفة إذا كان ملف مُعين بمساره موجود أم لا , ولمعرفة ذلك نقوم باستدعاء الدالة file_exists التي تقبل وسيطاً وحيداً هو مسار الملف و تُعيد القيمة true في حال وجوده :

```
<?php
if(file_exists('file1.txt') === true)
{
   echo 'file "file.txt" exists';
}
echo '<br>';
if(file_exists('file2.txt') === false)
{
   echo 'file "file2.txt" does not exists';
}
?>
```

الحصول على حجم تخزين ملف :

في حال اردنا معرفة حجم ملف , نستخدم الدالة filesize التي تقبل وسيطاً واحداً هو مسار الملف , وتُعيد هذه الدالة حجم الملف مقدراً بالبايت , وللحصول على الحجم مقدراً بالكيلوبايت أو الميغا بايت , نقسم على 1024 أو (1024*1024) على التوالي وبالترتيب :

```
<?php
$size = filesize('file1.txt');
echo 'The size of file1.txt is : '. floor($size / 1024) .' KB';
?>
```

استخراج امتداد ملف :

كما تعلم لكل نوع من الملفات امتداد معين خاص بها , حيث يكون الامتداد مسبوقاً بنقطة , فلذلك نقوم باستخراج الامتداد عن طريق الدالة explode - التي سبق شرحها في فصل مصفوفات حيث يكون امتداد الملف هو اخر سلسلة نصية تكون مسبوقة بنقطة "." كما في المثال التالي :

```
<?php
$file = 'file.example.txt';
```

```
$ext = explode('.', $file);
echo 'The file extension is : ' . $ext[count($ext) - 1];
//ومن ثم انقصنا منها 1 للحصول $ext للحصول على عدد عناصر المصفوفة count استخدمنا الدالة
. على مفتاح اخر عنصر
?>
```

الحصول على وقت تعديل أو تغيير أو الوصول لملف :

للحصول على بصمة الوقت التي تمثل آخر وقت لتغيير ملف ما , نستخدم الدالة filectime , حيث تقبل هذه الدالة وسيطاً واحداً هو مسار الملف .

أما للحصول على بصمة الوقت لآخر تعديل على الملف , نستخدم الدالة filemtime , وتقبل هذه الدالة - كما في الدالة السابقة - وسيطاً وحيداً هو مسار الملف .

الفرق التقني بين الدالة filectime و الدالة filemtime هو أن الدالة filectime تُعيد جميع التغيرات على ملف سواءً على محتوياته أم على صلاحيات الوصول إليه أم تغيير المستخدم المالك له . أما الدالة filemtime فهي تشير الى آخر تعديل في محتويات الملف فقط .

> **ملاحظة** : الحرف c في الدالة filectime يدل على كلمة change , أما الحرف m في الدالة الثانية فهو يدل على الكلمة modification .

```
<?php
echo date("m/d/Y H:i:s", filemtime('file1.txt'));
?>
```

والدالة fileatime تُعيد بصمة وقت آخر وصول للملف أو false في حال فشلها , وكما في الدوال السابقة فهي تقبل مسار ملف ما كوسيط .

الحصول على صلاحيات ملف :

بعد معرفة وجود قيد ما سواءً أكان ملفاً أم مجلداً , علينا أن نعلم ما هي الأفعال التي يمكننا القيام بها على القيد , أهل لدينا الصلاحيات للقراءة و الكتابة و التنفيذ .

في php نستخدم الدوال is_executable , is_writable , is_readable لمعرفة امكانية القراءة أو الكتابة أو التنفيذ على التوالي وبالترتيب .

تُعيد هذه الدوال true في حال نجاحها أو false ماعدا ذلك , وتقبل وسيطاً وحيداً هو مسار القيد .

المثال التالي يختبر إمكانية القراءة والكتابة و التنفيذ والحصول على حجم الملف وغيرها من المعلومات المتعلقة بالملف file1.txt :

```
<?php
$file = 'file1.txt';
echo '<pre>';
if(file_exists($file) === true)
{
   echo "Displaying file information for file $file ...<br>";
   echo 'File path :' .realpath($file). '<br>';
   echo 'File size :'. floor(filesize($file) / 1024). 'KB <br>';
   echo 'Last File changing time : ' .date("m/d/Y H:i:s",
filectime('file1.txt')). '<br>';
   echo 'Last File modification time : '. date("m/d/Y H:i:s",
filemtime('file1.txt')). '<br>';
   echo 'Last File access : ' .date("m/d/Y H:i:s",
fileatime('file1.txt')). '<br>';
   echo 'Is readable? : ';
   echo is_readable($file) == true ? 'true' : 'false';
   echo '<br>';
   echo 'Is writable? : ';
   echo is_writable($file) == true ? 'true' : 'false';
   echo '<br>';
   echo 'Is executable? : ';
   echo is_executable($file) == true ? 'true' : 'false';
```

```
   echo '<br>';
}
else
{
   echo "File $file is not exists ...<br>";
}
echo '</pre>';
?>
```

مثال على إخراج الكود السابق للملف file1.txt الموجود في مجلد /opt/lampp/htdocs/ :

```
Displaying file information for file file1.txt ...
File path :/opt/lampp/htdocs/image/file1.txt
File size :8KB
Last File changing time : 01/25/2013 21:57:47
Laast File modification time : 01/25/2013 20:40:03
Last File access : 01/25/2013 20:40:05
Is readable? : true
Is writable? : true
Is executable? : false
```

حذف ملف :

كما تتيح لك php إنشاء الملفات تُتيح لك حذفها , للقيام بهذه العملية استخدم الدالة unlink , تقبل هذه الدالة وسيطاً واحداً هو مسار الملف المُراد حذفه , وبالتأكيد يجب أن تكون لديك صلاحيات كتابة على الملف المُحدد حتى تستطيع حذفه عدا ذلك سيتم اظهار خطأ E_WARNING .

```
<?php
unlink('file1.txt');
?>
```

تغيير صلاحيات قيد :

كما في نظام linux والانظمة الشبيهة باليونكس , حيث نستخدم الأمر chmod لتغيير صلاحيات

قيد ما , نستخدم الدالة chmod في لغة php للقيام بالمهمة ذاتها .
لكن php لا تقبل الاعلان عن الصلاحيات كسلسلة نصية مثلاً "a-wx", وإنما تحصرها فقط باستخدام الصلاحيات بالارقام في النظام الثماني , أي تكون الصلاحية مكونة من اربعة ارقام الرقم الاول هو صفر , أما الارقام الثلاث الباقية هي عبارة عن الصلاحيات للمستخدم و لمجوعة المستخدم و لبقية المستخدمين على التوالي و بالترتيب , الجدول التالي يوضح الارقام والمصلاحيات المقابلة لها

الرقم 0	يشير الى عدم اعطاء أي صلاحية
الرقم 1	يشير الى اعطاء صلاحية التنفيذ فقط .
الرقم 2	يشير الى اعطاء صلاحية الكتابة فقط .
الرقم 3	يشير الى اعطاء صلاحية الكتابة والتنفيذ .
الرقم 4	يشير الى اعطاء صلاحية القراءة فقط .
الرقم 5	يشير الى اعطاء صلاحية القراءة والتنفيذ .
الرقم 6	يشير الى اعطاء صلاحية القراءة و الكتابة .
الرقم 7	يشير الى اعطاء صلاحية القراءة والكتابة والتنفيذ .

نقبل هذه دالة chmod وسيطين , الاول هو مسار القيد المُراد تغيير صلاحياته,والثاني هو الصلاحية مثال :

```
<?php
chmod('folder/file1.txt', 0600); # القراءة والكتابة للمستخدم , لاشيئ لبقية المستخدمين
chmod('folder/file1.txt', 0755); # القراءة والكتابة والتنفيذ للمالك , القراءة و التنفيذ لبقية المستخدمين
```

```
?>
```

نسخ أو نقل ملف :

نستعمل الدالة copy لنسخ الملفات , تقبل هذا الدالة وسيطين الاول هو مسار الملف المُراد نسخة و الثاني هو المسار الجديد . في حال وجود ملف في المسار الجديد فسيتم استبداله تلقائياً .

```
copy($source, $dest);
```

اما لنقل ملف فنستخدم الدالة rename التي تقوم اساساً بتغيير اسم الملف لكن يمكن استخدامها لنقله , تقبل هذه الدالة وسيطين الاول هو مسار الملف و الثاني هو مسار الملف الجديد :

```
rename($oldname, $newname);
```

مثال :

```
<?php
copy ('file1.txt', 'file2.txt');
rename('file2.txt', '../file.txt');
?>
```

قراءة الملفات والكتابة عليها :

قبل اجراء أي عمليات على الملف , علينا تهيئته وذلك بانشاء مقبض للملف عن طريق الدالة fopen التي تقبل وسيطين الوسيط الأول هو مسار الملف , أما الوسيط الثاني هو الوضع المراد فتح الملف به , الجدول التالي يبين الاوضاع المختلفة لفتح ملف :

الوضعr	يقوم بفتح الملف للقراءة فقط مع وضع مؤشر الملف في بدايته (سنتحدث لاحقاً عن مؤشر الملف وكيفية تحريكه) .
الوضع r+	يقوم بفتح الملف للقراءة والكتابة مع وضع مؤشر الملف في بدايته .
الوضع w	يقوم بفتح الملف للكتابة فقط ويقوم بمسح جميع محتوياته , وإذا لم يكن

	الملف موجوداً سوف يقوم بانشاءه .
الوضع w+	كما في الوضع w , لكنه يقوم بفتح الملف للقراءة والكتابة .
الوضع a	يقوم بفتح الملف للكتابة فقط ويضع مؤشر الملف عند نهايته , إذا لم يكن الملف موجوداً يقوم بإنشاءه .
الوضع a+	يقوم بفتح الملف للقراءة و الكتابة ويضع مؤشر الملف عند نهايته , إذا لم يكن الملف موجوداً يقوم بإنشاءه .

الفرق بين a و w يكمن في أن w يقوم بحذف محتويات الملف بينما الوضع a يحافظ على محتويات الملف و يضع المؤشر عند نهايته .
يمكن اضافة الحرف b الى الاحرف السابقة لفتح الملف بالنظام الثنائي, ويفيد هذا الوضع عند القراءة أو الكتابة على ملفات غير نصية .
يجب أن تكون لديك الصلاحية للقراءة أو الكتابة (حسب الوضع الذي تقوم باستخدامه) على الملف , ما عدا ذلك سيتم توليد رسالة خطأ مفادها أنك لا تملك الصلاحيات الكافية للقيام بتلك المهمة .
و يمكنك باستخدام الدالة fopen للإشارة الى ملفات خارج الخادم المُنفذ عليه البرنامج , عن طريق بروتوكول HTTP أو ftp أو غيرهم ... لكن للقراءة فقط :

```
<?php
$handle = fopen("file.txt", "r");
$handle = fopen("./folder/file.zip", "wb");
$handle = fopen("http://www.example.com/", "r");
$handle = fopen("ftp://user:password@example.com/somefile.txt",
"w");
?>
```

يتم تحرير الذاكرة باغلاق مقبض الملف عن طريق الدالة fclose التي تقبل وسيطا وحيداً هو مقبض الملف المُنشئ بواسطة الدالة السابقة .

ملاحظة : انشاء مقبض لملف لا يقتصر على الدالة fopen حيث يوجد دوال اخرى مثل الدالة fsockopen التي تقوم بانشاء مقبض عن طريق اتصال socket , وهذا الموضوع خارج عن نطاق هذا الفصل .

قراءة البيانات من ملف :

بعد انشاء مقبض الملف باستخدام الدالة fopen , نقوم باستخدام الدالة fread للقراءة من الملف وتقبل وسيطين : الاول هو مقبض الملف , والثاني هو عدد البايتات التي سيتم قراءتها من الملف بدءاً من مكان وجود مؤشر القراءة :

```
fread($handle, $length);
```

ولقراءة الملف باكمله نقوم بتحديد قيمة الوسيط length بحجم الملف عن طريق الدالة filesize كما في المثال التالي الذي يقوم بطباعة محتويات الملف file1.txt :

```
<?php
$filename = 'file1.txt';
$handle = fopen($filename, 'r');
$contents = fread($handle, filesize($filename));
echo $contents;
fclose($handle);
?>
```

الدالة fgets شبيهة جداً بالدالة fread وتقوم بنفس العمل تقريباً حيث في معظم الاحيان يمكن استخدام fgets عوضا عن fread , حيث الفرق الاساسي بينهما هو ان الدالة fgets لا تتطلب تحديد عدد البايتات التي يجب قراءتها من الملف حيث تكون القيمة الافتراضية للوسيط length هي 1024 بايت . وكلا الدالتين تقومان بالتوقف عن القراءة عندما تصلان الى نهاية الملف EOF End Of File)

الكتابة على ملف :

من اهم العمليات التي يمكن اجراءها على ملف هو اضافة و تعديل محتواه , ويتم ذلك في لغة php عن طريقة الدالة fwrite التي تقوم بكتابة البيانات المُمررة اليها الى ملف , طبعا يجب فتح الملف بوضع يسمح بالكتابة عليه كما في الوضعين (w , a) , الدالة fwrite تقبل ثلاثة وسطاء , الوسيط الاول هو مقبض الملف والثاني هو البيانات المُراد كتابتها و الثالث اختياري يمثل عدد البايتات التي سيتم كتابتها , فاذا تم تحديد الوسيط الثالث فان الكتابة على الملف سوف تتوقف عندما يصبح عدد البايتات المكتوبة مساويا لقيمة هذا الوسيط , الشكل العام للدالة fwrite هو :

```
fwrite($handle, $string, $length);
```

لا تنسى ان الدالة fwrite تقوم بطباعة رسالة خطأ عند عدم توفر صلاحيات للكتابة .

المثال التالي يقوم يكتابة الجملة "Hello World !!" على الملف file1.txt :

```
<?php
$filename = 'file1.txt';
$handle = fopen($filename, 'w+');
fwrite($handle, 'Hello World !!');
fclose($handle);
?>
```

الدالة fputs هي دالة مكافئة Alias للدالة fwrite , اي انها تقوم بنفس العمل تماماً وتأخذ الوسائط ذاتها

الدالة feof :

تقوم هذه الدالة باعادة true في حال وصل مؤشر القراءة الى نهاية الملف و false عدا ذلك و تُفيد عندما نقوم بالدوران على محتويات ملف لقراءته , وتقبل هذه الدالة وسيطا وحيداً هو مقبض الملف.

```
<?php
$file = fopen('file1.txt', 'r');
while(!feof($file))
{
```

```
echo fgets($file). '<br>';
}
fclose($file);
?>
```

تغيير مكان المؤشر :

لتغيير مكان المؤشر سواء عند القراءة أو الكتابة نستخدم الدالة fseek التي تقبل وسيطين اجباريين , الاول هو مقبض الملف والثاني هو offset الذي سوف يتم وضع المؤشر عنده .

```
fseek($handle, $offset);
```

> **ملاحظة** : عند فتح الملف بوضع a أو a+ فإن الكتابة سوف تكون في اخر الملف حتى لو قمت بتغيير مكان المؤشر .

الدالتين file_get_contents و file_put_contents :

تقوم الدالة file_get_contents بقراءة ملف بأكمله على شكل سلسلة نصية و يُمرر لها مسار الملف كوسيط , شكلها العام :

```
file_get_contents($filename);
```

اما الدالة file_put_contents فتقوم بكتابة البيانات المُمررة اليها بالوسيط الثاني على الملف الذي يتم تحديده بمساره والذي يشكل الوسيط الاول :

```
file_put_contents($filename, $data);
```

وتقوم هذه الدالة بانشاء الملف اذا لم يكن موجوداً , وفي حال وجوده تقوم بمسح جميع محتوياته !

ايهما استخدم fopen ومن ثم اقرأ الملف عن طريق fread ام استخدم file_get_contents ؟

بشكل بسيط اذا كنت تريد قراءة جميع محتويات ملف ما فاستخدم file_get_contents اما اذا كنت تريد قراءة عدد محدد من البايتات فاستخدم fread وذلك لتوفير اكبر قدر ممكن من الذاكرة .

ثانياً : التعامل مع المجلدات

يمكن باستخدام php القيام بمختلف العمليات على المجلدات كانشاءها و حذفها وتغيير صلاحيات الوصول إليها .

القراءة من مجلد :

كما في دالة fopen عند التعامل مع الملفات , تُستخدم الدالة opendir للحصول على مقبض للمجلد , حيث تقبل الدالة opendir وسيطاً واحداً هو مسار المجلد , الشكل العام لتعريف هذه الدالة هو :

```
$resorce = opendir($path);
```

وايضاً لتحرير الذاكرة وإغلاق مقبض الملف , نستخدم الدالة closedir التي تقوم بعمل مشابه للدالة fclose , تقبل هذه الدالة وسيطاً واحداً هو مقبض المجلد الذي قُمنا بإنشاءه باستخدام الدالة opendir :

```
closedir($handle);
```

قراءة محتويات مجلد :

تستخدم الدالة readdir لقراءة القيد التالي من مجلد تم انشاء مقبضه بواسطة الدالة opendir , حيث تقوم هذه الدالة بقراءة قيود الملفات على التتالي وحسب ترتيب نظام الملفات المُستخدم . تقبل هذه الدالة وسيطاً واحداً هو مقبض المجلد و تُعيد القيد (اسم الملف أو المجلد) , وللمرور على جميع قيود المجلد نستخدم حلقة التكرار while . ولتطبيق الدوال الثلاث السابقة نجرب المثال التالي:

```
<?php
$dir = opendir('folder');
while (($file = readdir($dir) )!== false)
{
   echo $file.'<br>';
}
```

```
closedir($dir);
?>
```

في البداية قمنا بانشاء مقبض للمجلد ومن ثم حلقة تكرار يتم فيها طباعة اسم الملف أو المجلد ومن ثم قمنا بتحرير الذاكرة واغلاق المقبض .

لاحظ وجود قيدين ممثلين بـ "." و ".." , و للتخلص منهم نعدل بالكود السابق لكي يتأكد من أن القيد لا يساوي احدى هاتين القيمتين :

```
<?php
$dir = opendir('folder');
while (($file = readdir($dir) )!== false)
{
   if($file == '.' OR $file == '..')
   {
      continue;
   }
   echo $file.'<br>';
}
closedir($dir);
?>
```

حذف المجلدات :

لحذف مجلد نستخدم الدالة rmdir التي تقبل وسيطاً واحداً هو مسار المجلد المُراد حذفه , لكن يجب أن يكون هذا المجلد فارغاً أما اذا كان المجلد يحوي أي ملف او مجلد فرعي , فلن يتم تنفيذ هذه التعليمة و سيتم توليد رسالة خطأ .

لكن اذا اردنا حذف مجلد يحوي ملفات ومجلدات فرعية , فيجب علينا أولاً أن نقوم بحذف جميع محتوياته قبل محاولة استدعاء الدالة السابقة :

```
<?php
function remove_dir($path)
{
```

```
   if(is_dir($path) === false)
   {
      return false;
   }
   $dir = opendir($path);
   while (($file = readdir($dir) )!== false)
   {
      if($file == '.' OR $file == '..')
      {
         continue;
      }
      if(is_file($path.'/'.$file))
      {
         unlink($path.'/'.$file);
      }
      elseif(is_dir($path.'/'.$file))
      {
         remove_dir($path.'/'.$file);
      }
   }
   rmdir($path);
   closedir($dir);
}
remove_dir('folder');
?>
```

إنشاء مجلد : بالطبع يمكنك باستخدام php انشاء المجلدات وتعيين صلاحيات الوصول إليها , ويتم ذلك بواسطة الدالة mkdir التي تقبل وسيطين , الأول اجباري هو اسم المجلد والثاني إختياري هو صلاحيات الوصول للمجلد , ويكون شكلها العام كالتالي :

```
mkdir($pathname, $mode);
```

وبشكل افتراضي يكون mode مساوياً للقيمة 0777 , أي صلاحيات القراءة والكتابة والتنفيذ لجميع المستخدمين .