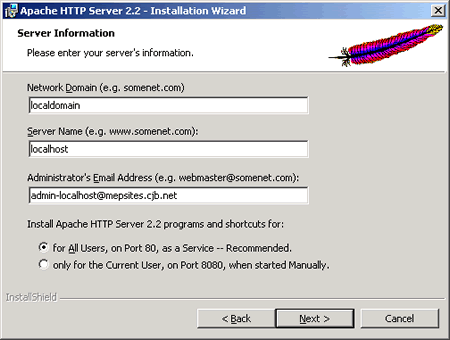
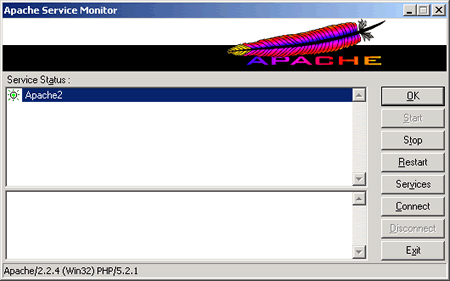
Muita gente usa o Windows como sistema principal, além do que, muitos não usam o Linux ainda. Seja por um motivo ou outro, mas é a realidade. E parte dessa gente desenvolve sites, páginas da web. Com o avanço da tecnologia não dá para ficar parado no HTML básico. Sites dinâmicos, que acessam banco de dados (como fóruns, cadastros, etc), precisam de uma solução diferente de mero HTML. Entra aí o PHP, normalmente rodando no servidor Apache, sob Linux/Unix.  
  
A hospedagem PHP é quase sempre mais barata (do que ASP, ColdFusion, entre outros concorrentes "closed source" do PHP) e, pela linguagem ser aberta, existem muitos módulos para PHP, permitindo uma diversidade muito maior do que com o ASP ou ASP.Net, da Microsoft. Mais barata mas de qualidade, muitas vezes superior. A quantidade de scripts prontos em PHP também é enorme, e o suporte comunitário é fácil de se obter em fóruns e grupos de discussão.  
  
Dá pra fazer praticamente de tudo com PHP, como lojas virtuais, sites de relacionamentos, interfaces para webmail, chats, serviços de help desk e fóruns em geral, gerenciadores de conteúdos para sites para agilizar a publicação, etc. Para fazer funcionar um site em PHP, normalmente usa-se a combinação LAMP: Linux + Apache + MySQL + PHP. Os servidores que oferecem suporte ao PHP quase sempre rodam Linux, mas o webmaster não precisa conhecer o sistema. Basta saber programar e carregar as páginas, usando uma interface web (gerenciador de arquivos) ou, mais profissionalmente, FTP.  
  
Pois bem, mas como testar os sites, localmente? Não basta dar um duplo clique nos arquivos .php, como se faz com os .htm ou .html. É necessário ter um servidor web configurado para isso. Isso quem desenvolve PHP já sabe, então vamos mostrar como rodar PHP no Windows (criando o "WAMP", Windows + Apache + MySQL + PHP ;).  
  
Assim você não precisa saber mexer no Linux, o que levaria uma boa dose de aprendizado (aprendizado para você, não desperdiçado, é claro, mas levaria um tempo a mais, afinal pode não ser fácil trocar de sistema). E também não precisaria ficar usando seu servidor web, usando sua conexão com a Internet nem gastando banda e espaço do seu plano de hospedagem. A idéia é então instalar e configurar um servidor web no seu computador, para acesso local. Vamos ver isso agora :)

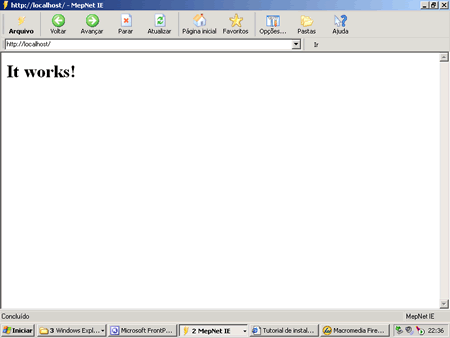
**O Apache**

O Apache é um servidor open source, muito estável e seguro, pois vem sendo desenvolvido há eras. Problemas de segurança são corrigidos num tempo menor, diferentemente do concorrente, IIS (Internet Information Services), da Microsoft. Há versão do Apache para Windows também, mas como o Linux é comprovadamente melhor para servidores web, a idéia de usar um servidor rodando no Windows é apenas para testar suas páginas, em tempo de desenvolvimento. Você cria o site, acessa-o localmente, como se estivesse usando a Internet. Seu próprio computador executa o servidor. Na hora do "vamos ver", você transfere seus arquivos para um servidor web e pronto.  
  
A questão mais difícil no Apache for Windows é a configuração inicial (no Linux também :P). Depois que o servidor está rodando a contento, é muito difícil dar problemas; os problemas podem ocorrer nos scripts usados, mas aí já é de responsabilidade do webmaster/programador. Muita gente se bate para colocar o Apache para funcionar no próprio micro, não encontra os arquivos compatíveis e trava numa mensagem de erro, ou seja, fica um tempão tentando colocar o servidor web para funcionar. Eu desenvolvo pequenos sites, e não avançava mais no PHP simplesmente porque não conseguia fazê-lo funcionar :(. Agora que consegui (ufa!), escrevi este tutorial para tentar ajudar muitas, muitas pessoas que também estejam tentando.  
  
O Apache hoje está na versão 2.x, e o PHP, na versão 5. Estou usando o Apache 2.2.4, e o PHP 5.2.1. Vamos instalar primeiro o Apache, depois o PHP (deixando o MySQL para uma outra oportunidade). Como este é um tutorial para webmasters que entendem da programação em PHP, deixaremos de lado características especiais dessa linguagem. A função é deixar o servidor pronto para rodar PHP localmente, depois é com você.

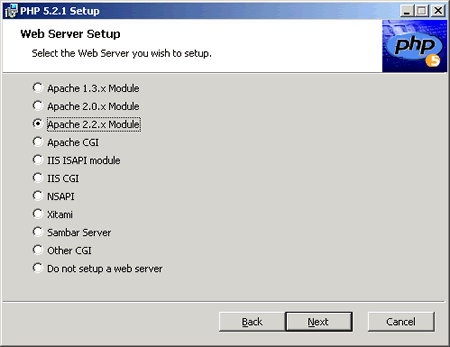
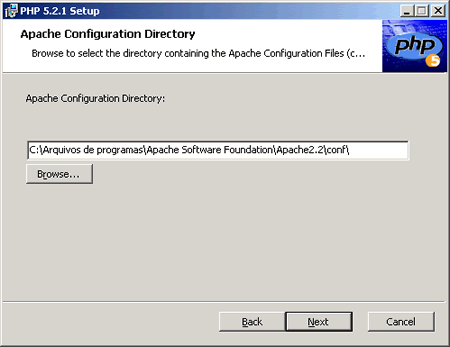
**Instalando o Apache**

Baixe o Apache para Windows em: <http://www.apache.org/dist/httpd/binaries/win32>  
  
Ele é fornecido compactado, num executável ou num pacote do Windows Installer. Este último é o mais recomendado. Desde que você use as versões recentes do Windows (2000, XP, etc), não encontrará problemas para instalá-lo. Prefira então o pacote "msi". O nome é algo como "apache\_x.x.xx-win32-x86.msi", onde x.x.x.x é o número da versão.  
  
**IMPORTANTE**: ao instalar servidores e/ou módulos para o Apache, execute o instalador usando uma conta com direitos administrativos no Windows.  
  
Ao baixá-lo, inicie a instalação, como você faria com qualquer programa para Windows com um bom instalador. Durante a instalação ele pedirá o domínio da rede (Network Domain), o nome do servidor (Server Name) e o e-mail do administrador do sistema. Como a instalação é para um servidor local, para desenvolvimento, preencha com "localdomain" no campo "Network Domain", "localhost" no campo "Server Name" e seu e-mail no último campo. Veja:  
  
Finalizada a instalação, o apache deverá estar funcionando. No Windows 2000/XP ele é instalado como um serviço que pode ser inicializado/desativado/reiniciado usando o console de serviços, o "services.msc". Por padrão, ele se configura para ser iniciado sempre junto com o Windows (veja na imagem de tela acima a opção na instalação para mudar isso, além de mudar a porta para 8080). Ao fazer alguma alteração num arquivo de configuração do Apache ou ao instalar um módulo, por exemplo, deve-se reiniciar o servidor (não o computador, mas sim o serviço do Apache). Isso pode ser feito com o "Apache Service Monitor", que ficará ativo na área de notificação (bandeja do sistema, próximo ao relógio):  


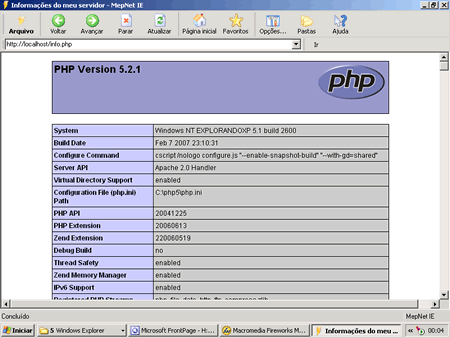
*Com esta interface pode-se parar, iniciar ou reiniciar o servidor do Apache no Windows*

Com ele instalado, abra qualquer navegador e digite o nome do seu computador, ou, de preferência, o nome "localhost", que sempre se refere ao computador local. Se preferir, acesse http://127.0.0.1, o IP da máquina local. Se tudo deu certo até aqui, você deverá ver uma mensagem "It works" ("Ele trabalha", "Ele funciona"). Veja:  
  
As versões mais antigas do Apache exibiam outras informações, mas a 2.2 mostra só isso. Ok, o servidor está funcionando. As páginas ficam por padrão na pasta "htdocs", dentro da pasta onde o Apache foi instalado. No meu caso, seria "C:\Arquivos de programas\Apache Software Foundation\Apache2.2\htdocs". Basta substituir o arquivo "index.html" pelo do seu site, e colocar suas páginas e subpastas aí. Assim você poderá acessar as subpastas da pasta "htdocs" digitando http://localhost/pasta, em qualquer navegador no micro local.  
  
Se o navegador ficar procurando o servidor, abrir uma página de busca ou dizer que não foi encontrado (exatamente como ocorre quando cai a conexão com a Internet), significa que o servidor web não foi iniciado corretamente. Nos itens básicos não têm o que dar errado, se for o caso, desinstale e instale novamente o Apache, tomando o cuidado de definir as opções da instalação como comentadas mais acima.  
  
Instalado o Apache, ele lê basicamente HTML puro. Porém, é diferente a forma de acesso, usando o endereço "http://localhost/index.html" de "C:\meu site\index.html". No primeiro caso, o navegador solicita a página ao computador "localhost" (que é o seu computador!), recebe-a do servidor, armazena-a no cache, para depois exibi-la. Ele trata como se fosse um site da Internet; como o servidor do site está na rede local (ou mesmo no seu próprio PC, um computador só), trata-se de uma Intranet. No segundo caso, você está acessando o arquivo diretamente no seu HD. O navegador pode até armazenar algum dado no cache, mas não há nenhuma comunicação com nenhum servidor. Ele lê o arquivo como qualquer programa leria qualquer arquivo.  
  
A configuração do Apache vai em alguns arquivos de texto puro, no estilo dos arquivos ".ini", e fica na pasta "conf", dentro da pasta do Apache. Ao editá-los e salvá-los, reinicie o servidor do apache para que as alterações entrem em vigor (usando o ícone do Apache na área de notificação, ou o services.msc, ou ainda o comando net stop apache2, e net start apache2, num prompt de comando). Os mais usados na versão Windows do Apache são o "httpd.conf" e o "mime.types".  
  
O "httpd.conf" é o principal, abordando diversas características do Apache. O "mime.types" define os tipos mime, tipos de arquivos e suas extensões, para que o Apache saiba como tratá-los. Por exemplo, ele deve saber que deve enviar arquivos HTML diretamente, sem processá-los, mas deve processar os arquivos PHP, antes de entregar o resultado ao browser. É importante que você faça backup destes arquivos, para restaurá-los, se você editar algo e der errado. Em ambos as linhas iniciadas com o caractere # são comentários, e são ignoradas. Para desativar um item ou colocar explicações, basta iniciar a linha com #.  
  
Um possível problema: se você mantiver outro servidor web ativo na máquina, poderá receber um erro do Apache ao tentar iniciá-lo. Isso pode ser por causa da porta 80: se você mantém o IIS ativo, ele normalmente é iniciado antes, e ocupa a porta 80 (praticamente qualquer servidor web se configura para usar a porta 80, que é dada como padrão). Para contornar isso e manter os dois servidores ativos, você deve trocar a porta de pelo menos um deles. No arquivo "httpd.conf" do Apache, localize o item "Listen 80", e troque o valor 80 por outro número que não esteja sendo usado por nenhuma outra porta. Por exemplo, 8080, 8081, etc. Localize mais para frente "ServerName localhost:80", e troque o 80 daí também.  
  
Se quiser, use o recurso de pesquisa do seu editor de textos para localizar as ocorrências do número 80. Salve o arquivo e reinicie o servidor. Agora, você deve acessar os sites digitando ":80" após o "domínio". Como é local, você deverá digitar: "http://localhost:8080", trocando é claro, 8080 pela porta escolhida por você. Se entrar sem definir a porta, o navegador muito provavelmente irá usar a 80, o que resultará que o site aberto será o do outro servidor (como o IIS, por exemplo). Sempre use o número da porta separado por dois pontos do nome do domínio. Assim, o certo é "http://localhost:8080/teste.htm", e não "http://localhost/teste.htm:8080". Preocupe-se com isso apenas se você manter mais de um servidor web na mesma máquina :).

PHP

O instalador do PHP 5 também pode ser encontrado no formato de pacote do Windows Installer (MSI). Prefira usar este, pois normalmente, é o mais fácil para instalar. Baixe-o em: (procure pela versão para win32):  
  
<http://www.php.net/downloads.php>  
  
Assim como com o nome do arquivo do instalador do Apache, o instalador do PHP possuirá um nome parecido com php-x.x.x-Win32.zip, onde o x.x.x corresponde à versão. Neste tutorial estou usando a versão 5.2.1; as telas de instalação poderão ser diferentes em outras versões, mas as idéias serão basicamente as mesmas. O PHP é um módulo para o Apache. Ao processar uma página PHP, o Apache envia a página ao módulo PHP, que a processa e devolve o resultado ao Apache, que por sua vez, envia ao browser que fez a solicitação.  
  
Para isso, a instalação do PHP consiste basicamente em colocar os arquivos numa pasta, e configurar o Apache para que "enxergue" os módulos PHP e chame-os sempre que encontrar um arquivo .php. Sem o PHP instalado, se você colocar um arquivo .php no site, ao acessá-lo pelo navegador ele será exibido como texto (com o código-fonte), ou o mais comum, o navegador pedirá para salvá-lo, como ao fazer o download de um arquivo qualquer. A definição do módulo do PHP no Apache vai no arquivo "httpd.conf", e o tipo de arquivo .php, no "mimes.types".  
  
O instalador do PHP 5 é muito sofisticado, e possui uma boa configuração automática. Se você baixar o arquivo .zip em vez do instalador, deverá extrair os arquivos para uma pasta e então configurar o Apache manualmente. Com o instalador, a instalação pode ser automática, para diversas versões do Apache, e inclusive para outros servidores, como o IIS, Xitami, etc. Mas para que ela se dê de forma automática, você deve especificar as opções corretas na instalação. Não se esqueça de instalar primeiro o Apache :)  
  
Na instalação do PHP, ele pedirá o tipo de servidor usado. No meu caso, escolhi Apache 2.2.x:  
  
Então ele pede o local dos arquivos de configuração do Apache. Localize a pasta "conf", dentro da pasta do Apache, senão a configuração não será automatizada (mas veremos também como fazê-la manualmente):  
  
Ao final da instalação o instalador do PHP pergunta se você quer que ele atualize os arquivos de configuração do Apache. Se quiser a configuração automática, confirme :)  
  
Se tudo ocorreu bem até aí, nenhuma mensagem de erro será exibida ao iniciar ou reiniciar o Apache, e as páginas .php serão processadas e entregues corretamente ao browser. Ainda não foi configurado um servidor de banco de dados, então apenas os recursos diretos do PHP funcionarão (e os dos outros módulos pré-instalados do Apache, é claro). Um ótimo meio de testar o PHP e ver diversas informações do servidor é criar uma página PHP, com o seguinte conteúdo:

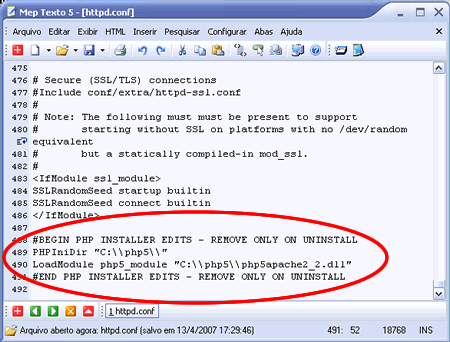
<html><head><title>Informações do meu servidor</title></head>  
<body>  
<? phpinfo(); ?>  
</body></html>

Salve-a na pasta "htdocs" do Apache, com a extensão .php, e abra no navegador:  
  
Se a página não puder ser acessada, pode haver um erro de configuração, ou de instalação do PHP. Se nem o "http://localhost" puder ser acessado, então o problema deve ser com o Apache. Se tiver erros nas configurações do Apache, ele poderá não ser iniciado, deixando o "servidor" fora do ar. Vamos ver agora como rever e alterar a configuração manualmente, editando os arquivos, inclusive possibilitando a correção de problemas.

### Configurando o PHP manualmente

A configuração do PHP envolve a edição do arquivo php.ini, na pasta onde o PHP for instalado, e a inclusão de referências a ele nos arquivos do Apache. Se você aceitar a configuração automática, não precisará alterar nada (a menos que ocorram erros). Mas se preferir configurar manualmente, arregace as mangas e vamos à obra!  
  
Para começar, instale o PHP usando somente os componentes padrões. Eu marquei todos os itens na instalação, e ele dava um erro no final, além do que eu não conseguia mais iniciar o Apache. Então desinstalei e instalei novamente, sem selecionar todos os componentes, até por desconhecer a função de vários deles. É melhor você fazer uma instalação enxuta, as possibilidades de erros são menores. Com mais componentes, fica mais difícil localizar o ponto do erro.  
  
Pense na montagem de um computador, vá colocando os recursos aos poucos, e testando. Se funcionar, avance, caso contrário, páre e veja a última coisa até onde funcionava. Se você simplesmente instalar tudo, ficará muito mais complicado encontrar as causas dos erros, especialmente se você não tiver prática com todos os recursos oferecidos no instalador do PHP (ou do que for).  
  
Para configurar manualmente o PHP, ou se a configuração automática falhar, são basicamente três passos. Vamos começar pela configuração do Apache. Abra o arquivo "httpd.conf", da pasta "conf", que está dentro da pasta do Apache. No final do arquivo, depois de tudo o que tiver nele, deverão existir estas linhas:

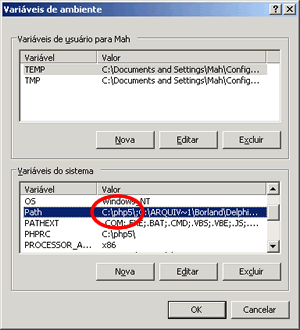
PHPIniDir "C:\\php5\\"  
LoadModule php5\_module "C:\\php5\\php5apache2\_2.dll

Veja num editor de textos, para ter uma idéia melhor:  


Neste screenshot a configuração destacada foi adicionada pelo instalador do PHP 5. Observe os comentários, pedindo para remover estas linhas apenas na desinstalação do PHP.

A linha **PHPIniDir** define a pasta do PHP, onde ele foi instalado (onde esta o php.ini original).  
  
A linha **LoadModule**... Carrega o módulo do PHP no Apache, que se chama "php5\_module", e está no arquivo "C:\php5\php5apache2\_2.dll".  
  
É recomendável instalar o PHP em uma pasta onde o nome do caminho dela não contenha espaços, mas isso não é obrigatório. Eu instalei em "C:\php5", seguindo a idéia das versões mais antigas, mas o instalador sugeria por padrão uma subpasta da "Arquivos de programas". Seguindo o exemplo, prefira usar duas barras invertidas (\\) em vez de uma só nos caminhos, e não se esqueça de colocar entre aspas se tiver espaços.  
  
Agora vamos configurar o "mime.types". Abra-o num editor de textos puro e adicione, também no final de tudo (tecnicamente, pode ser onde você quiser, mas não se confunda depois :), as linhas:

application/x-httpd-php php  
application/x-httpd-php-source phps

Elas identificam os tipos de arquivo .php e .phps, para que o Apache saiba lidar com eles e então enviá-los corretamente aos browsers.  
  
Agora falta configurar o arquivo "php.ini". Ele deve ficar numa pasta que esteja cadastrada na variável de ambiente "path", do Windows. Isso permite que qualquer programa chame este arquivo apenas pelo nome, sem precisar indicar o caminho completo.  
  
Antigamente se editava o php.ini, e depois ele era copiado para a pasta do Windows (ou system32). Nada impede que você faça isso, mas na instalação do PHP5, com a configuração automatizada, ele adiciona à variável PATH o caminho da pasta do PHP. Para isso é preciso que a instalação seja feita por um administrador. Você pode confirmar se a pasta do PHP está ou não na variável de ambiente PATH indo nas propriedades do "Meu computador", guia "Avançado", botão "Variáveis de ambiente":  


Veja que a pasta do PHP foi adicionada automaticamente à variável PATH

Estando com o php.ini em uma pasta definida na variável de ambiente Path (se não estiver, copie-o para a pasta do Windows), vamos editá-lo. Abra-o num editor de textos puro (por padrão ele fica na pasta do PHP).  
  
Uma configuração útil é ativar o uso dos delimitadores <? ... ?>, muito usados, por serem mais simples. No php.ini localize a linha que começa com "short\_open\_tag", e troque o Off por On (cuidado com as letras maiúsculas e minúsculas!). Ela deverá ficar:

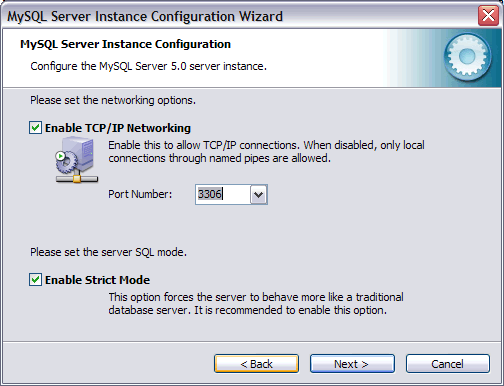
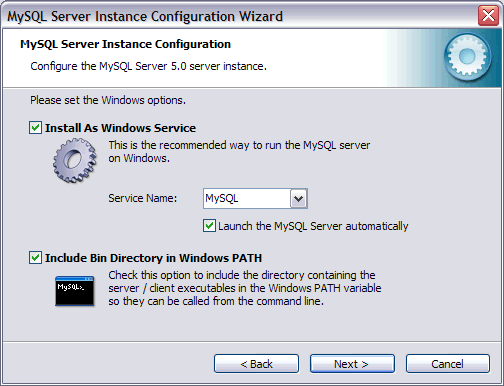
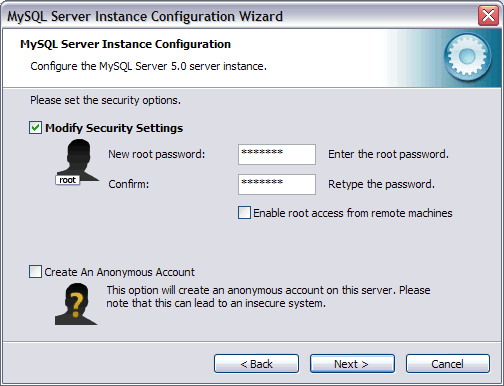
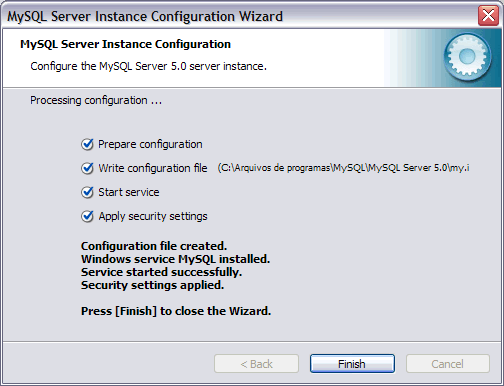
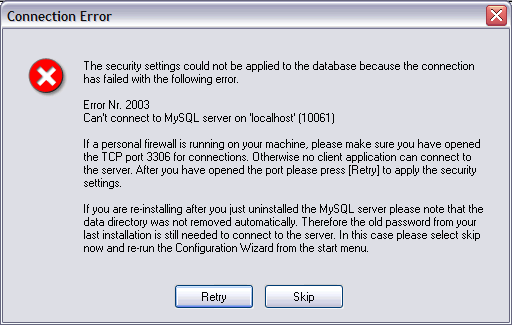
short\_open\_tag = On

Sem fazer isso (se deixar Off), os scripts iniciados com <? não serão identificados como scripts PHP, mas apenas os que iniciarem com <?php ou <script language="php">. Aproveitando, você pode definir "On" para a linha mais abaixo, "asp\_tags", para permitir que os scripts que usem os delimitadores <% e %> sejam executados, caso esteja acostumado com ASP e prefira utilizar esta forma.  
  
O php.ini vem bem comentado (em inglês), role a tela e leia-o com calma, para ver o que você pode alterar. É ideal ter conhecimentos avançados de PHP para alterá-lo por conta, e não se esqueça de manter um backup a cada alteração. No final da segunda parte deste tutorial comentarei sobre outras configurações necessárias a serem feitas no php.ini, para que os scripts PHP funcionem corretamente com os recursos mais usados.  
  
Cuidado: se você instalar o PHP por cima de uma instalação existente, o instalador poderá adicionar uma nova configuração nos arquivos "httpd.conf" e "mime.types", duplicando as configurações, causando um erro na inicialização do Apache. O ideal antes de instalar uma nova versão ou reinstalar, é remover manualmente as linhas adicionadas pelo PHP nos arquivos do Apache, e então usar o Adicionar ou remover programas, do painel de controle do Windows, para removê-lo. Após isso, instale normalmente a nova versão (ou a mesma :). Saiba ainda que, ao desinstalar o Apache e/ou o PHP, os arquivos de configuração deles não serão removidos por motivos de segurança, pois podem conter modificações suas. Se quiser "limpar" o sistema e iniciar a instalação do zero, desinstale o PHP e o Apache pelo painel de controle, e a seguir, apague as pastas deles. Se você copiou o php.ini para a pasta do Windows, remova-o de lá também.

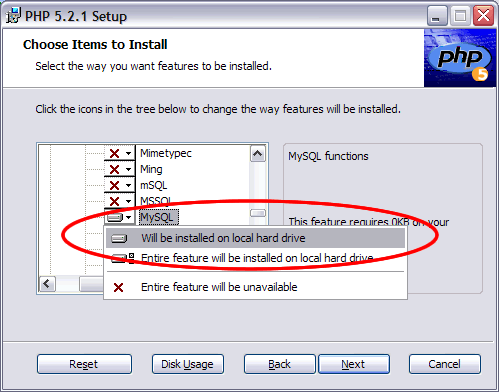
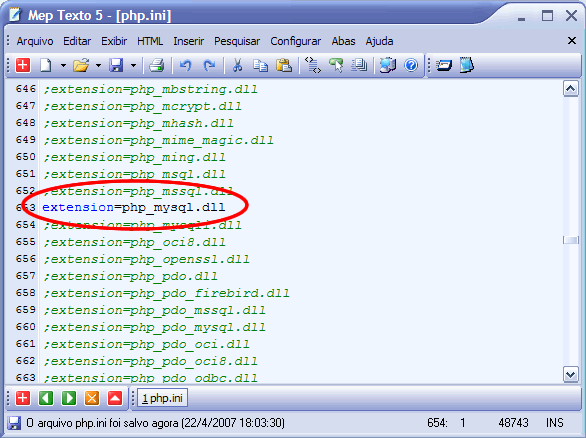
### Concluindo a primeira etapa

A configuração do Apache + PHP no Windows é basicamente esta. Agora é com você e suas páginas.  
  
Tenha em mente que este tutorial foi escrito para desenvolvedores de sites que querem desenvolver páginas dinâmicas em PHP, mas que usam o Windows no desktop ou workstation, e precisam criar e testar as páginas off line, e/ou ainda para quem está estudando PHP e não assina um plano de hospedagem. A hospedagem dos sites propriamente ditos deverá ser feita em servidores web, normalmente basta carregar os arquivos da pasta "htdocs" do Apache via FTP.  
  
A configuração de redes e servidores web foge ao objetivo deste tutorial, por isso não abordamos as outras configurações do Apache, como número máximo de sessões, alteração da pasta root do site, criação de "subsites", domínios e DNS, etc.

MySQL

O MySQL, como você já deve saber, é um gerenciador de banco de dados. O funcionamento dele no sistema não será abordado aqui, apenas as idéias básicas de como criar os bancos de dados a serem usados com o PHP, além da instalação.  
  
Procure a versão para Windows, na página do projeto:  
  
[www.mysql.com](http://www.mysql.com)  
  
Neste tutorial estou usando a 5.0.37. Na página de download há um pacote “mysql-essential-5.0.37-win32.msi” e outro maior, “mysql-5.0.37-win32.exe”. O segundo possui mais algumas coisas, documentação adicional, etc., mas na prática não fará tanta diferença, para os recursos básicos. Se você não quiser baixar o pacote maior, pode optar pelo “essential”, “essencial”.  
  
O executável de instalação apresenta a mesma interface da instalação do Apache e do PHP, ou seja, não tem muito segredo, basta ir clicando em Avançar. Deixe todas as opções que você não souber do que se trata, com os valores padrões. No final da instalação ele poderá pedir um cadastro no MySQL.com, normalmente pule essa parte (clicando em “Skip Sign-Up”).  
  
Instalado, na última tela de instalação ele exibirá uma caixinha de marcação para configurar logo ao concluir. Deixe-a marcada. Aparecerá um assistente, no mesmo estilo da instalação. Este assistente de configuração poderá ser executado a qualquer momento depois, bastando ir em “Iniciar > Programas > MySQL > MySQL Server 5.0 > MySQL Server Instance Config Wizard”.  
  
Esse assistente permite definir diversas coisas importantes, que poderão causar o funcionamento não esperado do servidor se alteradas indevidamente. Deixe o padrão para o que não souber do que se trata, apenas vá clicando em “Avançar” (“Next”).  
  
Num ponto ele pede a ativação de conexões TCP/IP, e permite escolher a porta onde ele ficará disponível, dando como padrão a 3306. Deixe esse valor, e se for o caso, libere essa porta para aceitar conexões de entrada no seu firewall. Mesmo fazendo a configuração num computador sem rede nem Internet, comigo não funcionou corretamente quando deixei desmarcada, o PHP não conseguia se comunicar com o MySQL:  
  
Mais algumas telas ele dará a opção de incluir o caminho do MySQL na variável de ambiente “Path” do Windows. É bom marcá-la, para que você possa chamar os executáveis do MySQL a partir de qualquer prompt, em qualquer pasta. Basta marcar a opção “Include Bin Directory in Windows PATH”:  
  
Na próxima tela deve-se definir a senha de root, que seria o administrador do MySQL. Se você mexe com PHP deve saber que para acessar um banco de dados é necessário fornecer um nome de usuário e senha válidos para aquele banco; você poderá usar o “root” e a senha definida aqui, ou outra conta. Justamente, para criar outras contas você precisará do root, e para alterar as configurações ou criar novos bancos de dados, também. Portanto, anote esta senha em um local seguro:  
  
Se você perdê-la, bastará desinstalar e instalar novamente o MySQL (e depois configurar as permissões dos bancos de dados novamente).  
  
Agora vem a última parte da configuração. Se algo der errado aqui, o MySQL poderá não funcionar corretamente. Apenas clique em “Execute” e cruze os dedos ;)  
  
Se tudo ocorreu bem, você verá uma tela como a exibida acima. Se ele não conseguir obter acesso à porta definida na instalação (por padrão, a 3306), ele poderá exibir esta mensagem:  
  
Clique em “Retry” quantas vezes forem necessárias, para tentar a configuração. Libere conexões de entrada na porta 3306, pelo seu firewall, se for o caso. Se clicar em “Skip” a configuração falhará, e você deverá executar o assistente de configuração novamente depois.  
  
Em instalações e reinstalações eu tive problemas, ele nunca conseguia atribuir as permissões corretamente, dizendo que a porta estava em uso, mesmo ela estando liberada no firewall e com todos os outros programas fechados. A melhor saída que encontrei, em caso de problemas desse tipo, foi ficar clicando no “Retry” várias, várias vezes.  
  
Se ainda assim não der, desinstale o MySQL, apague a pasta “MySQL” da pasta “Arquivos de programas”, reinicie o computador e inicie a instalação novamente. É algo tedioso, mas se falhar nesta tela o serviço não será iniciado corretamente e os scripts PHP não poderão usar os bancos de dados. Talvez você dê sorte e ele funcione de primeira; estou abordando esta parte pois não sei o que houve, comigo demorou muito até funcionar da primeira vez.  
  
Feito isso, o servidor de banco de dados MySQL será configurado para iniciar automaticamente com o Windows. Você pode alterar esse comportamento, para ele e para o Apache, no console de serviços do Windows (“Iniciar > Executar > services.msc”). Se seu computador tem pouca memória, você pode preferir iniciá-los apenas manualmente, quando necessário, para que não sejam iniciados sempre junto com o Windows. Com 256 MB e Windows XP não há muito o que reclamar não, dá até para deixar estes serviços ativos, mesmo que você não mexa com PHP em toda sessão. Uma questão é sobre a abertura da porta 3306 para o MySQL, se você mantiver acesso à Internet no computador diretamente (sem passar por um roteador, proxy ou outro firewall). Se for o caso você pode desativar o uso do TPC/IP na configuração do MySQL, e tentar acessá-lo nos scripts locais (comigo não funcionou), ou então deixar o serviço “MySQL” no services.msc desativado, e só iniciá-lo quando for testar.  
  
Dica: para evitar que um serviço no Windows seja iniciado automaticamente, abra o console de serviços, indo em “Iniciar > Executar” e digitando “services.msc”. Dê um duplo clique no serviço desejado e escolha o modo de inicialização “manual”. Para iniciá-lo depois, você pode ir nesse console de serviços, selecionar o item desejado e clicar em “Iniciar”, ou então via linha de comando. Digite num prompt ou no “Executar”: ***net start serviço*** para iniciar um serviço, e ***net stop serviço*** para pará-lo. Por exemplo, net start mysql, para iniciar o servidor MySQL quando ele não estiver sendo iniciado automaticamente.

### Ativando o MySQL no PHP

Instalado o MySQL, falta agora configurar o PHP. Na instalação típica no Windows, o PHP não instala nenhuma das extensões adicionais. Para que o MySQL possa ser usado em conjunto com o PHP, a extensão MySQL deverá ser instalada e carregada. Se você baixou o arquivo zipado do PHP e fez todo o processo manual, deverá agora incluir a DLL da extensão MySQL e configurar o php.ini para encontrá-la. Se você usou o instalador do pacote “.msi”, ele pode fazer isso automaticamente. Com o PHP instalado, abra o instalador dele novamente, e quando solicitado, clique em “Change”, para alterar os componentes instalados. Se preferir, acesse o painel de controle “Adicionar ou remover programas” do Windows, localize o PHP e clique em “Alterar” (e não em “Remover”). Na seção dos componentes, na categoria “Extensions”, localize o item “MySQL”, clique no ícone de um disco ao lado dele, e clique na opção “Will be installed on local hard drive”. Veja:  
  
Fazendo isso, o PHP instalará suporte ao MySQL. Mas você ainda deverá ativá-lo no php.ini, para que efetivamente, funcione. Localize a linha “;extension=php\_mysql.dll” e retire o comentário dela:  
  
Só para desencargo de consciência, localize também a seção [MySQL] (no arquivo php.ini), e veja se o item “mysql.allow\_persistent” está ativo, com o valor “On” (se não estiver, ative-o então):

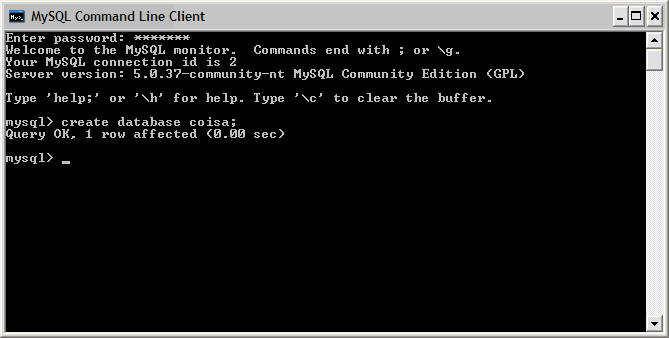
mysql.allow\_persistent = On

Se você fizer a instalação manual, deverá colocar a DLL “php\_mysql.dll” na pasta de extensões do PHP; normalmente, essa pasta chama-se “ext” e fica dentro da pasta do PHP. Ao usar o instalador dele, ele já copia o arquivo para essa pasta.  
  
Finais

Dica: se você pretende instalar suporte ao MySQL diretamente, pode selecionar a extensão “MySQL” já durante a instalação do PHP. Neste tutorial vimos dessa forma por questões didáticos.

Quando você usa um servidor de hospedagem compartilhado, normalmente cria os bancos de dados usados pelas suas aplicações PHP através de uma interface via navegador, provida pelo seu provedor de hospedagem. Localmente, você deverá criar os bancos de dados via linha de comando, ou criando uma página PHP com instruções query do SQL. Para criar bancos de dados, acesse o prompt do MySQL, clicando no item “MySQL Command Line Client”, no grupo de atalhos do MySQL no “Iniciar > Programas”. Ele pedirá a senha de root, definida na instalação. A partir daí, é com você.  
  
Para criar um banco de dados, digite:

create database coisa;

Veja:  
  
Fazendo isso, esse banco de dados poderá ser usado nos scripts em PHP. Muitos scripts prontos que você pode baixar em diversos sites de materiais para web masters, contém arquivos INSTALL, LEIAME, etc, com as instruções. Você deve criar o banco de dados usado no script antes de rodar os scripts. A criação das tabelas pode ser feita via páginas PHP, depois de selecionado o banco de dados, ou pelo prompt do MySQL. Para criar tabelas pelo prompt, você deverá saber digitar as instruções do MySQL. Uma dica é copiar as instruções e colar no prompt, teclando [enter] ao final da última linha colada.  
  
Antes de criar a tabela, deve-se selecionar o banco de dados a ser usado. Para isso, digite no prompt:

use coisa;

onde “coisa” é o nome do banco de dados, que já deverá existir. Para criar a tabela, use a sintaxe:

create table tabela1 (  
  
declaração do campo 1,  
  
declaração do campo 2,  
  
declaração do campo 3,  
  
);

A declaração da criação dos campos e tabelas não será abordada neste tutorial, por fugir do objetivo do mesmo. Você pode obter estas instruções em livros ou apostilas sobre MySQL, ou então instalar um script (como o phpmyadmin, disponível em [www.phpmyadmin.net](http://www.phpmyadmin.net)) que cria uma interface via navegador para a criação de bancos de dados e tabelas.  
  
Alguns scripts usam usuários específicos para o banco de dados. Se você está só estudando e testando, pode sempre usar o root e a senha definida na instalação do MySQL, será mais fácil. O root tem livre acesso a todos os bancos de dados. Na hora de publicar seu site você deverá editar o script, alterando o nome de usuário e a senha (e o nome do banco de dados, se for o caso). Usar só o root em servidores de produção não é recomendável, aliás é recomendável manter contas separadas para cada aplicação ou banco de dados; se uma brecha de segurança em uma aplicação for descoberta e explorada, ela não afetará os outros bancos de dados; fora que em servidores compartilhados você normalmente não tem acesso ao root.  
  
Se você quiser criar um usuário no MySQL e atribuir direitos a um banco de dados específico para ele, de forma mais semelhante ao que é usado em servidores de produção, pode dar este comando no prompt do MySQL:

grant all on coisa.\* to usuario identified by 'senha';

Onde coisa é o nome do banco de dados, usuário é o nome de usuário desejado (um nome sem espaços) e senha é a senha dele. O nome usado aqui poderá ser usado nos scripts, para acessar esse banco de dados.

### Outras configurações importantes do PHP

Se você mexe com PHP, mesmo que minimamente, já deve saber que é possível iniciar os scripts usando <?php ou <?, simplesmente. Há ainda a notação usada no ASP, da Microsoft, possível no PHP para facilitar para quem está acostumado com a sintaxe do ASP: <%. Tem as variáveis passadas nas páginas pelo endereço (método GET do HTTP), como em <a href=pagina.php?id=5>. E ainda, entre outras coisas, a recuperação de variáveis automaticamente, usando o método POST do HTTP, onde, por exemplo, um campo <input name=username type=text> numa página HTML gera automaticamente uma variável de nome “username” cujo conteúdo seria o texto preenchido pelo usuário, válida globalmente na página definida em “action” no formulário.  
  
Você programa em PHP usando estes recursos, e os scripts funcionarão na maioria dos servidores de hospedagem. Mas, ao rodá-los no micro local, com o PHP instalado da forma padrão, simplesmente eles não rodarão. O PHP vem com muita coisa desativada, seja por motivo de performance (desempenho) ou segurança. Como são técnicas usadas na programação em PHP na maioria dos scripts, não dá para ficar sem elas. Você deverá ativá-las no PHP. Vamos então ver algumas opções que você deve alterar no php.ini:

short\_open\_tag = On

Esse item vem desativado por padrão, localize-o e deixe-o com o valor On. Ele libera o uso dos delimitadores <? ... ?> para identificar scripts em PHP. Se você não ativá-lo, poderá usar apenas as formas <?php ...> e <script language=php> nas suas páginas.

asp\_tags = On

Ative este item apenas se você gostar de usar as tags no estilo do ASP, como em <% ... %>. Se não usar isso, pode deixar desativado.

display\_errors = On

Esse item vem desativado. Para um servidor de produção, é altamente recomendável deixá-lo desativado. Ele imprime erros de sintaxe ou funcionamento do PHP na página. Por questões de segurança, é bom deixá-lo em Off, pois entre as mensagens de erros muitas vezes vêm nomes de arquivos, caminhos locais, e se você bobear, até a senha do banco de dados, se tiver erro de sintaxe justamente na declaração da mesma. Em contrapartida, num servidor local, para aprender PHP, é recomendável deixá-lo em On, ativado.  
  
Os erros serão impressos na página, com a indicação do número da linha e o arquivo afetado, ou seja: é uma forma de “debugar” suas aplicações, identificando erros e indo facilmente ao ponto que os causa. Um caso comum, num erro em PHP, é a página ficar toda em branco ao ser acessada (vazia), especialmente se o erro estiver no começo do script. Se você está estudando ou treinando, deverá caçar no script onde está o erro.  
  
Com esta opção ativada, ele mostrará a linha e o nome do arquivo que causou o erro, quebrando um galhão para você. Aproveitando, um pouco mais acima no php.ini, deixe o item “error\_reporting = E\_ALL” com o valor E\_ALL, que mostra todos os tipos de erros.

register\_globals = On  
  
register\_long\_arrays = On  
  
register\_argc\_argv = On  
  
auto\_globals\_jit = On

Deixe estes quatro itens em On. Com eles ativados você não terá problemas ao passar variáveis da forma habitual para as páginas, seja usando o método GET ou POST do HTTP. Muitos scripts que usam formulários não funcionam na configuração padrão do PHP justamente por causa destes itens, que vêm desativados :)

file\_uploads = On  
  
upload\_max\_filesize = 2M

Se algum script fará upload de arquivos pelo HTTP (ou seja, através de uma página da web), como por exemplo, o carregamento de fotos para um site de relacionamentos ou fórum, você deve deixar ativada a opção file\_uploads. O upload\_max\_filesize determina o tamanho máximo dos arquivos que poderão ser carregados. Num servidor de produção não é bom deixar um valor muito grande, para não reduzir a performance. O valor deve ser escrito na forma XM, onde X é um número, em megabytes.

### Considerações finais

Particularmente eu me bati um tempão, sendo que a maior dose de problemas estava no arquivo php.ini, com as muitas opções comentadas desativadas.  
  
É basicamente isso, agora é com você! Não se esqueça de reiniciar o Apache, após fazer modificações no php.ini ou nos arquivos de configuração dele. Para reiniciá-lo, a forma mais “simples” é clicar com o botão normal do mouse no ícone do monitor do serviço do Apache, na área de notificação (próximo ao relógio), e escolher a opção “Restart”. Outro meio fácil é usar o atalho presente em “Iniciar > Programas > Apache HTTP Server > Control Apache Server > Restart”.  
  
Só para alertar uma última vez, não se esqueça de trocar o nome de usuário, senha e talvez o nome do banco de dados, ao carregar seus scripts para seu provedor de hospedagem. Bom trabalho!