# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# CURSO DE LINGUAGEM PHP





Autor: Maurício Vivas de Souza Barreto
mauricio@cipsga.org.br

**Abril de 2000**

Maurício Vivas de Souza Barreto

mauricio@cipsga.org.br

*vivas@usa.net*

Abril de 2000



Projeto Supervisionado de Final de Curso

Este apostila de PHP e fruto do Projeto Supervisionado de Final de Curso de Maurício Vivas de Souza Barreto, tendo o mesmo sido submetido a uma banca examinadora composta pelo Professor Giovanny Lucero, Professora Ana Rosimeri e Professor Leonardo Nogueira Matos, da Universidade Federal de Sergipe, Centro de Ciências Exatas e Tecnologia do Departamento de Estatistica e Informática.

# *Indice*

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# 1. Introdução

# O que é PHP?

PHP é uma linguagem que permite criar sites WEB dinâmicos, possibilitando uma interação com o usuário através de formulários, parâmetros da URL e links. A diferença de PHP com relação a linguagens semelhantes a Javascript é que o código PHP é executado no servidor, sendo enviado para o cliente apenas html puro. Desta maneira é possível interagir com bancos de dados e aplicações existentes no servidor, com a vantagem de não expor o código fonte para o cliente. Isso pode ser útil quando o programa está lidando com senhas ou qualquer tipo de informação confidencial.

O que diferencia PHP de um script CGI escrito em C ou Perl é que o código PHP fica embutido no próprio HTML, enquanto no outro caso é necessário que o script CGI gere todo o código HTML, ou leia de um outro arquivo.



# O que pode ser feito com PHP?

Basicamente, qualquer coisa que pode ser feita por algum programa CGI pode ser feita também com PHP, como coletar dados de um formulário, gerar páginas dinamicamente ou enviar e receber *cookies*.

PHP também tem como uma das características mais importantes o suporte a um grande número de bancos de dados, como dBase, Interbase, mSQL, mySQL, Oracle, Sybase, PostgreSQL e vários outros. Construir uma página baseada em um banco de dados torna-se uma tarefa extremamente simples com PHP.



Além disso, PHP tem suporte a outros serviços através de protocolos como IMAP, SNMP, NNTP, POP3 e, logicamente, HTTP. Ainda é possível abrir *sockets* e interagir com outros protocolos.

# Como surgiu a linguagem PHP?

A linguagem PHP foi concebida durante o outono de 1994 por **Rasmus Lerdorf**. As primeiras versões não foram disponibilizadas, tendo sido utilizadas em sua *home-page* apenas para que ele pudesse ter informações sobre as visitas que estavam sendo feitas. A primeira versão utilizada por outras pessoas foi disponibilizada em 1995, e ficou conhecida como "**P**ersonal **H**ome **P**age Tools" (ferramentas para página pessoal). Era composta por um sistema bastante simples que interpretava algumas *macros* e alguns utilitários que rodavam "por trás" das *home-pages*: um livro de visitas, um contador e algumas outras coisas.

Em meados de 1995 o interpretador foi reescrito, e ganhou o nome de **PHP/FI**, o "FI" veio de um outro pacote escrito por Rasmus que interpretava dados de formulários HTML (**F**orm **I**nterpreter). Ele combinou os scripts do pacote *Personal Home Page Tools* com o FI e adicionou suporte a mSQL, nascendo assim o PHP/FI, que cresceu bastante, e as pessoas passaram a contribuir com o projeto.

Estima-se que em 1996 PHP/FI estava sendo usado por cerca de 15.000 *sites* pelo mundo, e em meados de 1997 esse número subiu para mais de 50.000. Nessa época houve uma mudança no desenvolvimento do PHP. Ele

deixou de ser um projeto de Rasmus com contribuições de outras pessoas para ter uma equipe de desenvolvimento mais organizada. O interpretador foi reescrito por **Zeev Suraski** e **Andi Gutmans**, e esse novo interpretador foi a base para a versão 3.

Atualmente o uso do PHP3 vem crescendo numa velocidade incrível, e já está sendo desenvolvida a versão 4 do PHP.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# 2. Sintaxe Básica

## Delimitando o código PHP

O código PHP fica embutido no próprio HTML. O interpretador identifica quando um código é PHP pelas seguintes tags:

```
<?php
comandos
?>

<script language="php">
comandos
</script>

<?
comandos
?>

<%
comandos
%>
```

O tipo de *tags* mais utilizado é o terceiro, que consiste em uma "abreviação" do primeiro. Para utilizá-lo, é necessário habilitar a opção *short-tags* na configuração do PHP. O último tipo serve para facilitar o uso por programadores acostumados à sintaxe de ASP. Para utilizá-lo também é necessário habilitá-lo no PHP, através do arquivo de configuração *php.ini*.

## Separador de instruções

Entre cada instrução em PHP é preciso utilizar o ponto-e-vírgula, assim como em C, Perl e outras linguagens mais conhecidas. Na última instrução do bloco de script não é necessário o uso do ponto-e-vírgula, mas por questões estéticas recomenda-se o uso sempre.

## Nomes de variáveis

Toda variável em PHP tem seu nome composto pelo caracter $ e uma string, que deve iniciar por uma letra ou o caracter "_". **PHP é case sensitive**, ou seja, as variáveis $vivas e $VIVAS são diferentes. Por isso é preciso ter muito cuidado ao definir os nomes das variáveis. É bom evitar os nomes em maiúsculas, pois como veremos mais adiante, o PHP já possui alguma variáveis pré-definidas cujos nomes são formados por letras maiúsculas.

# Comentários

Há dois tipos de comentários em código PHP:

## Comentários de uma linha:

Marca como comentário até o final da linha ou até o final do bloco de código PHP – o que vier antes. Pode ser delimitado pelo caracter "#" ou por duas barras ( // ).



```
Exemplo:
<?  echo "teste"; #isto é um teste  ?>
<?  echo "teste"; //este teste é similar ao anterior ?>
```

## Comentários de mais de uma linha:

Tem como delimitadores os caracteres "/*" para o início do bloco e "*/" para o final do comentário. Se o delimitador de final de código PHP ( ?> ) estiver dentro de um comentário, não será reconhecido pelo interpretador.



Exemplos:

```
<?
  echo "teste"; /* Isto é um comentário com mais
de uma linha, mas não funciona corretamente ?>
*/

<?
  echo "teste"; /* Isto é um comentário com mais
de uma linha que funciona corretamente
*/
?>
```

# 3. Criando os primeiros scripts

## Primeiro Exemplo

Neste exemplo, criaremos um script com uma saída simples, que servirá para testar se a instalação foi feita corretamente:

```
<html>
<head><title>Aprendendo PHP</title></head>
<body>

<?php
echo "Primeiro Script";
?>

</body>
</html>
```



Salve o arquivo como “primeiro.php3” no diretorio de documentos do Apache (ou o Web Server escolhido). Abra uma janela do navegador e digite o endereço “http://localhost/primeiro.php3”. Verificando o código fonte da página exibida, temos o seguinte:



```
<html>
<head><title>Aprendendo PHP</title></head>
<body>

Primeiro Script

</body>
</html>
```

Isso mostra como o PHP funciona. O script é executado no servidor, ficando disponível para o usuário apenas o resultado. Agora vamos escrever um script que produza exatamente o mesmo resultado utilizando uma variável:

```
<html>
<head><title>Aprendendo PHP</title></head>
<body>

<?php
$texto = "Primeiro Script";
echo $texto;
?>

</body>
</html>
```

# Utilizando formulários HTML

Ao clicar num botão “Submit” em um formulário HTML as informações dos campos serão enviadas ao servidor especificado para que possa ser produzida uma resposta. O PHP trata esses valores como variáveis, cujo nome é o nome do campo definido no formulário. O exemplo a seguir mostra isso, e mostra também como o código PHP pode ser inserido em **qualquer** parte do código HTML:

```
<html>
<head><title>Aprendendo PHP</title></head>
<body>

<?php
if ($texto != "")
  echo "Você digitou \"$texto\"<br><br>";
?>

<form method=post action="<? echo $PATH_INFO; ?>">
<input type="text" name="texto" value="" size=10>
<br>
<input type="submit" name="sub" value="Enviar!">
</form>

</body>
</html>
```





Ao salvar o arquivo acima e carregá-lo no browser, o usuário verá apenas um formulário que contém um espaço para digitar o texto, como visto na figura 01. Ao digitar um texto qualquer e submeter o formulário, a resposta, que é o mesmo arquivo PHP (indicado pela constante $PATH_INFO, que retorna o nome do arquivo) será como na figura 02:

| | |
|---|---|
| [Imagem16] | [Imagem17] |
| *figura 01* | *figura 02* |

Isso ocorre porque o código PHP testa o conteúdo da variável $texto. Inicialmente ele é uma string vazia, e por isso nada é impresso na primeira parte. Quando algum texto é digitado no formulário e submetido, o PHP passa a tratá-lo como uma variável. Como no formulário o campo possui o nome “texto”, a variável com seu conteúdo será $texto. Assim, no próximo teste o valor da variável será diferente de uma string vazia, e o PHP imprime um texto antes do formulário.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# Interagindo com o browser

PHP também permite interagir com informações do browser automaticamente. Por exemplo, o script a seguir mostra informações sobre o browser do usuário. As figuras 03 e 04 mostram o resultado visto no Netscape Communicator e o Microsoft Internet Explorer, respectivamente.

```
<html>
<head><title>Aprendendo PHP</title></head>
<body>

<? echo $HTTP_USER_AGENT; ?>

</body>
</html>
```



[Imagem18]

*figura 03*

[Imagem19]

*figura 04*

Observe que o resultado mostra características de cada browser, como a versão, e no caso do Communicator até o idioma (“en”). Com isso, se você criar uma página com recursos disponíveis somente no Internet Explorer, por exemplo, pode esconder o código dos outros browsers, com um código semelhante ao seguinte:



```
<html>
<head><title>Aprendendo PHP</title></head>
<body>

<?
if  (strpos($HTTP_USER_AGENT,"MSIE 5") != 0) {
  echo "Você usa Internet Explorer";
} else {
  echo "Você não usa Internet Explorer";
}
?>

</body>
</html>
```

Neste exemplo, será apenas exibido um texto informando se está sendo utilizado o Microsoft Internet Explorer ou não, mas para outras funções poderia ser utilizado algo semelhante.

É bom notar o surgimento de mais uma função no código anterior: `strpos(string1,string2)`. Essa função retorna a posição da primeira aparição de string2 em string1, contando a partir de zero, e não retorna valor algum se não ocorrer. Assim, para testar se a string `$HTTP_USER_AGENT` contém a string `“MSIE”`, basta testar se `strpos` devolve algum valor.

# Acessando Bancos de Dados

Neste documento todos os exemplos referentes a acesso de bancos de dados utilizarão o gerenciador de banco de dados MySQL, que pode ser copiado gratuitamente no site *http://www.mysql.org*.

Para interagir com uma base de dados SQL existem três comandos básicos que devem ser utilizados: um que faz a conexão com o servidor de banco de dados, um que seleciona a base de dados a ser utilizada e um terceiro que executa uma *"query"* SQL.

## Conexão com o servidor

A conexão com o servidor de banco de dados mySQL em PHP é feita através do comando `mysql_connect`, que tem a seguinte sintaxe:



```
int mysql_connect(string /*host [:porta]*/ , string /*login*/ , string /*senha*/ );
```

Os parâmetros são bastante simples: o endereço do servidor(host), o nome do usuário (login) e a senha para a conexão. A função retorna um valor inteiro, que é o identificador da conexão estabelecida e deverá ser armazenado numa variável para ser utilizado depois. No nosso exemplo, temos como servidor de banco de dados a mesma máquina que roda o servidor http, como login o usuário "root" e senha "phppwd":



```
$conexao = mysql_connect("localhost", "root", "phppwd");
```

Assim, se a conexão for bem sucedida (existir um servidor no endereço especificado que possua o usuário com a senha fornecida), o identificador da conexão fica armazenado na variável `$conexão`.

## Seleção do banco de dados

Uma vez conectado, é preciso selecionar o banco de dados existente no servidor com o qual desejamos trabalhar. Isso é feito através da função int `mysql_select_db`, que possui a seguinte sintaxe:

```
int mysql_select_db(string /*nome_base*/, int /*conexao*/ );
```

O valor de retorno é 0 se o comando falhar, e 1 em caso de sucesso. O nome da base de dados a selecionar é o primeiro parâmetro fornecido, seguido pelo identificador da conexão. Se este for omitido, o interpretador PHP tentará utilizar a última conexão estabelecida. Recomenda-se sempre explicitar esse valor, para facilitar a legibilidade do código. No nosso exemplo, a base de dados a ser selecionada possui o nome "ged":

```
mysql_select_db("ged", $conexao);
```

Após a execução desse comando qualquer consulta executada para aquela conexão utilizará a base de dados selecionada.

## Execução de queries SQL

Após estabelecida a conexão e selecionada a base de dados a ser utilizada, quase toda a interação com o servidor mySQL pode ser feita através de consultas escritas em SQL (Structured Query Language), com o comando `mysql_query`, que utiliza a seguinte sintaxe:

```
int mysql_query(string consulta, int [conexao] );
```

O valor de retorno é 0 se falhar ou 1 em caso de sucesso. Sucesso aqui significa que a consulta está sintaticamente correta e foi executada no servidor. Nenhuma informação sobre o resultado é retornada deste comando, ou até mesmo se o resultado é o esperado. No caso da consulta ser um comando SELECT, o valor de retorno é um valor interno que identifica o resultado, que poderá ser tratado com a função `mysql_result()` e outras. A string query não deve conter ponto-e-vírgula no final do comando, e o identificador da conexão é opcional. Vamos criar uma tabela como exemplo:

```
$cria = "CREATE TABLE exemplo (codigo INT AUTO_INCREMENT PRIMARY KEY, nome CHAR(40), email CHAR(50))";

mysql_query($cria, $conexao);
```

Agora vejamos como ficou o código completo para executar uma query SQL numa base de dados mySQL, com um exemplo que cria uma tabela chamada exemplo e adiciona alguns dados:

```
$conexao = mysql_connect("localhost", "root", "phppwd");
mysql_select_db("ged", $conexao);

$cria = "CREATE TABLE exemplo (codigo INT AUTO_INCREMENT PRIMARY KEY, nome CHAR(40), email CHAR(50))";

$insere1 = "INSERT INTO exemplo (nome,email) VALUES ("Mauricio Vivas","vivas@usa.net");

$insere2 = "INSERT INTO exemplo (nome,email) VALUES ("Jose da Silva","jose@teste.com");

$insere3 = "INSERT INTO exemplo (nome,email) VALUES ("Fernando Henrique Cardoso","fhc@planalto.gov.br");

$insere4 = "INSERT INTO exemplo (nome,email) VALUES ("Bill Clinton","president@whitehouse.gov");

mysql_query($cria, $conexao);
mysql_query($insere1, $conexao);
mysql_query($insere2, $conexao);
mysql_query($insere3, $conexao);
mysql_query($insere4, $conexao);
```

# Tratamento de resultados de query SELECT

Ao executar uma query SQL SELECT através do comando `mysql_query`, o identificador do resultado deve ser armazenado numa variável que pode ser tratada de diversas formas. Duas maneiras interessantes de fazê-lo usam o comando `mysql_result` e o comando `mysql_fetch_row`, respectivamente.

O comando `mysql_result` tem a seguinte sintaxe:

```
int mysql_result(int resultado, int linha, mixed [campo]);
```

Onde `resultado` é o identificador do resultado, obtido com o retorno da função `mysql_query`, linha especifica a tupla a ser exibida, já que uma query SELECT pode retornar diversas tuplas, e `campo` é o identificador do campo a ser exibido, sendo o tipo descrito como `mixed` pela possibilidade de ser de diversos tipos (neste caso, inteiro ou string). Vejamos um exemplo utilizando a tabela criada anteriormente:



```
$consulta = “SELECT nome, email FROM exemplo WHERE email LIKE ‘vivas’”;

$resultado = mysql_query($consulta, $conexao);

printf("Nome: ", mysql_result($resultado,0,"nome"), “<br>\n”);
printf("e-mail: ", mysql_result($resultado,0,"email"),“<br>”);
```



Com o exemplo acima, o resultado será:

```
Nome: Mauricio Vivas<br>
e-mail: vivas@usa.net<br>
```

É importante notar que a utilização desta função é um pouco trabalhosa, já que no caso de um resultado com várias linhas é preciso controlar o número de linhas para tratá-las (pode-se utilizar a função `mysql_num_rows(int resultado)`, que retorna o número de linhas de um resultado), e no caso de uma alteração no nome do campo é preciso alterar também a maneira de tratá-lo. Por isso é mais aconselhável que se use uma outra função, como por exemplo `mysql_fetch_row`, que possui a seguinte sintaxe:

```
array mysql_fetch_row(int result);
```

A variável `resultado` é o identificador da memória de resultados, obtido como retorno da função `mysql_query`. O resultado produzido por esta função é de retirar a primeira linha da memória de resultados, se houver, e colocá-la num array. Assim torna-se mais fácil tratar um resultado com várias linhas, e sem utilizar os nomes dos campos na rotina de tratamento do resultado:

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
$consulta = “SELECT nome, email FROM exemplo”;

$resultado = mysql_query($consulta, $conexao);

echo "<table border=1>\n";
echo "<tr><td>Nome</td><td>e-mail</tr>\n";
while ($linha = mysql_fetch_row($resultado)) {
  printf("<tr><td>$linha[0]</td>”);
  printf("<td>$linha[1]</td></tr>”);
}
echo "</table>\n";
```

O código acima irá imprimir todos os registros da tabela exemplo numa tabela html. Se o programador desejar “pular” alguma(s) linha(s) do resultado, poderá utilizar a função `mysql_data_seek`, que tem por objetivo definir qual será a próxima linha da memória de resultados a ser impressa. Sua sintaxe é:

```
int mysql_data_seek(int resultado, int linha);
```

Sendo `resultado` o identificador do resultado e `linha` o numero da linha. Retorna 0 em caso de falha, e um valor diferente de zero em caso de sucesso.

Existem diversas outras funções para o tratamento de resultados, que armazenam as linhas em arrays e objetos, assim como outras funções para administrar o banco de dados, mas como este documento trata-se de uma introdução, inicialmente não tratará tópicos mais avançados.

# 4. Tipos

## Tipos Suportados

PHP suporta os seguintes tipos de dados:

- Inteiro
- Ponto flutuante
- String
- Array
- Objeto



PHP utiliza checagem de tipos dinâmica, ou seja, uma variável pode conter valores de diferentes tipos em diferentes momentos da execução do script. Por este motivo não é necessário declarar o tipo de uma variável para usá-la. O interpretador PHP decidirá qual o tipo daquela variável, verificando o conteúdo em tempo de execução.



Ainda assim, é permitido converter os valores de um tipo para outro desejado, utilizando o *typecasting* ou a função `settype` (ver adiante).

### Inteiros (integer ou long)

Uma variável pode conter um valor inteiro com atribuições que sigam as seguintes sintaxes:

```
$vivas = 1234; # inteiro positivo na base decimal
$vivas = -234; # inteiro negativo na base decimal
$vivas = 0234; # inteiro na base octal-simbolizado pelo 0
               # equivale a 156 decimal
$vivas = 0x34; # inteiro na base hexadecimal(simbolizado
               # pelo 0x) – equivale a 52 decimal.
```

A diferença entre inteiros simples e `long` está no número de bytes utilizados para armazenar a variável. Como a escolha é feita pelo interpretador PHP de maneira transparente para o usuário, podemos afirmar que os tipos são iguais.

#### *Números em Ponto Flutuante (double ou float)*

Uma variável pode ter um valor em ponto flutuante com atribuições que sigam as seguintes sintaxes:

```
$vivas = 1.234;
$vivas = 23e4; # equivale a 230.000
```

## Strings

Strings podem ser atribuídas de duas maneiras:

a) utilizando aspas simples ( ' ) – Desta maneira, o valor da variável será exatamente o texto contido entre as aspas (com exceção de \\ e \' – ver tabela abaixo)

b) utilizando aspas duplas ( " ) – Desta maneira, qualquer variável ou caracter de escape será expandido antes de ser atribuído.



Exemplo:

```
<?
$teste = "Mauricio";
$vivas = '---$teste--\n';
echo "$vivas";
?>
```

A saída desse script será "---$teste--\n".



```
<?
$teste = "Mauricio";
$vivas = "---$teste---\n";
echo "$vivas";
?>
```

A saída desse script será "---Mauricio--" (com uma quebra de linha no final).

A tabela seguinte lista os caracteres de escape:

| **Sintaxe** | **Significado** |
|---|---|
| \n | Nova linha |
| \r | Retorno de carro (semelhante a \n) |
| \t | Tabulação horizontal |
| \\ | A própria barra ( \ ) |
| \$ | O símbolo $ |
| \’ | Aspa simples |
| \” | Aspa dupla |

No apêndice 01 está disponível uma lista das funções utilizadas no tratamento de strings.

## Arrays

Arrays em PHP podem ser observados como mapeamentos ou como vetores indexados. Mais precisamente, um valor do tipo array é um dicionário onde os índices são as chaves de acesso. Vale ressaltar que os índices podem ser valores de qualquer tipo e não somente inteiros. Inclusive, se os índices forem todos inteiros, estes não precisam formar um intervalo contínuo

Como a checagem de tipos em PHP é dinâmica, valores de tipos diferentes podem ser usados como índices de array, assim como os valores mapeados também podem ser de diversos tipos.

```
Exemplo:
<?
$cor[1] = “vermelho”;
$cor[2] = “verde”;
$cor[3] = “azul”;
$cor[“teste”] = 1;
?>

Equivalentemente, pode-se escrever:

<?
$cor = array(1 => “vermelho, 2 => “verde, 3 => “azul”, “teste => 1);
?>
```



# Listas



As listas são utilizadas em PHP para realizar atribuições múltiplas. Através de listas é possível atribuir valores que estão num array para variáveis. Vejamos o exemplo:

```
Exemplo:

 list($a, $b, $c) = array(“a”, “b”, “c”);
```

O comando acima atribui valores às três variáveis simultaneamente. É bom notar que só são atribuídos às variáveis da lista os elementos do array que possuem índices inteiros e não negativos. No exemplo acima as três atribuições foram bem sucedidas porque ao inicializar um array sem especificar os índices eles passam a ser inteiros, a partir do zero. Um fator importante é que cada variável da lista possui um índice inteiro e ordinal, iniciando com zero, que serve para determinar qual valor será atribuído. No exemplo anterior temos $a com índice 0, $b com índice 1 e $c com índice 2. Vejamos um outro exemplo:

```
$arr = array(1=>”um”,3=>”tres”,”a”=>”letraA”,2=>”dois);
list($a,$b,$c,$d) = $arr;
```

Após a execução do código acima temos os seguintes valores:

```
$a == null
$b == “um”
$c == “dois”
$d == “tres”
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

Devemos observar que à variável $a não foi atribuído valor, pois no array não existe elemento com índice 0 (zero). Outro detalhe importante é que o valor “tres” foi atribuído à variável $d, e não a $b, pois seu índice é 3, o mesmo que $d na lista. Por fim, vemos que o valor “letraA” não foi atribuído a elemento algum da lista pois seu índice não é inteiro.

Os índices da lista servem apenas como referência ao interpretador PHP para realizar as atribuições, não podendo ser acessados de maneira alguma pelo programador. De maneira diferente do array, uma lista não pode ser atribuída a uma variável, servindo apenas para fazer múltiplas atribuições através de um array.

No apêndice 02 está disponível uma lista das funções mais comuns para o tratamento de arrays.

## Objetos

Um objeto pode ser inicializado utilizando o comando *new* para instanciar uma classe para uma variável.

```
Exemplo:
class teste {
          function nada() {
               echo “nada”;
          }
}

$vivas = new teste;
$vivas -> nada();
```

A utilização de objetos será mais detalhada mais à frente.

## Booleanos

PHP não possui um tipo booleano, mas é capaz de avaliar expressões e retornar *true* ou *false*, através do tipo `integer`: é usado o valor 0 (zero) para representar o estado *false*, e qualquer valor diferente de zero (geralmente 1) para representar o estado *true*.

# Transformação de tipos

A transformação de tipos em PHP pode ser feita das seguintes maneiras:

## Coerções

Quando ocorrem determinadas operações (“+”, por exemplo) entre dois valores de tipos diferentes, o PHP converte o valor de um deles automaticamente (coerção). É interessante notar que se o operando for uma variável, seu valor não será alterado.

O tipo para o qual os valores dos operandos serão convertidos é determinado da seguinte forma: Se um dos operandos for `float`, o outro será convertido para `float`, senão, se um deles for `integer`, o outro será convertido para `integer`.

```
Exemplo:
$vivas = “1”;       // $vivas é a string “1”
$vivas = $vivas + 1;// $vivas é o integer 2
$vivas = $vivas + 3.7;// $vivas é o double 5.7
$vivas = 1 + 1.5          // $vivas é o double 2.5
```

Como podemos notar, o PHP converte `string` para `integer` ou `double` mantendo o valor. O sistema utilizado pelo PHP para converter de *strings* para números é o seguinte:

- É analisado o início da `string`. Se contiver um número, ele será avaliado. Senão, o valor será 0 (zero);
- O número pode conter um sinal no início (“+” ou “-“);
- Se a `string` contiver um ponto em sua parte numérica a ser analisada, ele será considerado, e o valor obtido será `double`;
- Se a `string` contiver um “e” ou “E” em sua parte numérica a ser analisada, o valor seguinte será considerado como expoente da base 10, e o valor obtido será `double`;

```
Exemplos:
$vivas = 1 + “10.5”;      // $vivas == 11.5
$vivas = 1 + “-1.3e3”;    // $vivas == -1299
$vivas = 1 + “teste10.5”; // $vivas == 1
$vivas = 1 + “10testes”;  // $vivas == 11
$vivas = 1 + "  10testes";     // $vivas == 11
$vivas = 1 + "+  10testes";    // $vivas == 1
```





## Transformação explícita de tipos

A sintaxe do *typecast* de PHP é semelhante ao C: basta escrever o tipo entre parenteses antes do valor

```
Exemplo:
$vivas = 15;              // $vivas é integer (15)
$vivas = (double) $vivas  // $vivas é double (15.0)
$vivas = 3.9              // $vivas é double (3.9)
$vivas = (int) $vivas     // $vivas é integer (3)
                          // o valor decimal é truncado
```

Os tipos de *cast* permitidos são:

| | |
|---|---|
| (int), (integer) | ⇒ muda para integer; |
| (real), (double), (float) | ⇒ muda para float; |
| (string) | ⇒ muda para string; |
| (array) | ⇒ muda para array; |
| (object) | ⇒ muda para objeto. |

## Com a função settype

A função `settype` converte uma variável para o tipo especificado, que pode ser “integer”, “double”, “string”, “array” ou “object”.

```
Exemplo:
$vivas = 15;              // $vivas é integer
settype($vivas,double)    // $vivas é double
```

# 5. Constantes

## Constantes pré-definidas

O PHP possui algumas constantes pré-definidas, indicando a versão do PHP, o Sistema Operacional do servidor, o arquivo em execução, e diversas outras informações. Para ter acesso a todas as constantes pré-definidas, pode-se utilizar a função `phpinfo()`, que exibe uma tabela contendo todas as constantes pré-definidas, assim como configurações da máquina, sistema operacional, servidor http e versão do PHP instalada.



## Definindo constantes

Para definir constantes utiliza-se a função `define`. Uma vez definido, o valor de uma constante não poderá mais ser alterado. Uma constante só pode conter valores escalares, ou seja, não pode conter nem um array nem um objeto. A assinatura da função `define` é a seguinte:



```
int define(string nome_da_constante, mixed valor);
```

A função retorna `true` se for bem-sucedida. Veja um exemplo de sua utilização a seguir:

```
define ("pi", 3.1415926536);
$circunf                    =                    2*pi*$raio;
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# 6. Operadores

## Aritméticos

Só podem ser utilizados quando os operandos são números (integer ou float). Se forem de outro tipo, terão seus valores convertidos antes da realização da operação.

| | |
|---|---|
| + | adição |
| - | subtração |
| * | multiplicação |
| / | divisão |
| % | módulo |



## de strings



Só há um operador exclusivo para strings:

| | |
|---|---|
| . | concatenação |

## de atribuição

Existe um operador básico de atribuição e diversos derivados. Sempre retornam o valor atribuído. No caso dos operadores derivados de atribuição, a operação é feita entre os dois operandos, sendo atribuído o resultado para o primeiro. A atribuição é sempre por valor, e não por referência.

| | |
|---|---|
| = | atribuição simples |
| += | atribuição com adição |
| -= | atribuição com subtração |
| *= | atribuição com multiplicação |
| /= | atribuição com divisão |
| %= | atribuição com módulo |
| .= | atribuição com concatenação |

```
Exemplo:

$a = 7;
$a += 2; // $a passa a conter o valor 9
```

# bit a bit

Comparam dois números bit a bit.

| | |
|---|---|
| & | “e” lógico |
| \| | “ou” lógico |
| ^ | ou exclusivo |
| ~ | não (inversão) |
| << | shift left |
| >> | shift right |

# Lógicos

Utilizados para inteiros representando valores booleanos

| | |
|---|---|
| and | “e” lógico |
| or | “ou” lógico |
| xor | ou exclusivo |
| ! | não (inversão) |
| && | “e” lógico |
| \|\| | “ou” lógico |

Existem dois operadores para “e” e para “ou porque eles têm diferentes posições na ordem de precedência.

# Comparação

As comparações são feitas entre os valores contidos nas variáveis, e não as referências. Sempre retornam um valor booleano.

| | |
|---|---|
| == | igual a |
| != | diferente de |
| < | menor que |

| | |
|---|---|
| > | maior que |
| <= | menor ou igual a |
| >= | maior ou igual a |

# Expressão condicional

Existe um operador de seleção que é ternário. Funciona assim:

```
(expressao1)?(expressao2):( expressao3)
```

o interpretador PHP avalia a primeira expressão. Se ela for verdadeira, a expressão retorna o valor de expressão2. Senão, retorna o valor de expressão3.



# de incremento e decremento

| | |
|---|---|
| ++ | incremento |
| -- | decremento |



Podem ser utilizados de duas formas: antes ou depois da variável. Quando utilizado antes, retorna o valor da variável antes de incrementá-la ou decrementá-la. Quando utilizado depois, retorna o valor da variável já incrementado ou decrementado.

Exemplos:

```
$a = $b = 10; // $a e $b recebem o valor 10
$c = $a++; // $c recebe 10 e $a passa a ter 11
$d = ++$b; // $d recebe 11, valor de $b já incrementado
```

# Ordem de precedência dos operadores

A tabela a seguir mostra a ordem de precedência dos operadores no momento de avaliar as expressões;

| Precedência | Associatividade | Operadores |
|---|---|---|
| 1. | esquerda | , |
| 2. | esquerda | or |
| 3. | esquerda | xor |
| 4. | esquerda | and |
| 5. | direita | print |
| 6. | esquerda | = += -= *= /= .= %= &= != ~= <<= >>= |
| 7. | esquerda | ? : |
| 8. | esquerda | \|\| |
| 9. | esquerda | && |
| 10. | esquerda | \| |
| 11. | esquerda | ^ |
| 12. | esquerda | & |
| 13. | não associa | == != |
| 14. | não associa | < <= > >= |
| 15. | esquerda | << >> |
| 16. | esquerda | + - . |
| 17. | esquerda | * / % |
| 18. | direita | ! ~ ++ -- (int) (double) (string) (array) (object) @ |
| 19. | direita | [ |
| 20. | não associa | new |

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# 7. Estruturas de Controle

As estruturas que veremos a seguir são comuns para as linguagens de programação imperativas, bastando, portanto, descrever a sintaxe de cada uma delas, resumindo o funcionamento.

## Blocos

Um bloco consiste de vários comandos agrupados com o objetivo de relacioná-los com determinado comando ou função. Em comandos como `if`, `for`, `while`, `switch` e em declarações de funções blocos podem ser utilizados para permitir que um comando faça parte do contexto desejado. Blocos em PHP são delimitados pelos caracteres "{" e "}". A utilização dos delimitadores de bloco em uma parte qualquer do código não relacionada com os comandos citados ou funções não produzirá efeito algum, e será tratada normalmente pelo interpretador.

```
Exemplo:
if ($x == $y)
  comando1;
  comando2;
```

Para que comando2 esteja relacionado ao if é preciso utilizar um bloco:

```
if ($x == $y){
  comando1;
  comando2;
}
```

## Comandos de seleção

Também chamados de condicionais, os comandos de seleção permitem executar comandos ou blocos de comandos com base em testes feitos durante a execução.

### if

O mais trivial dos comandos condicionais é o `if`. Ele testa a condição e executa o comando indicado se o resultado for `true` (valor diferente de zero). Ele possui duas sintaxes:

```
if (expressão)
   comando;
```

```
if (expressão):
   comando;
   . . .
```

```
   comando;
endif;
```

Para incluir mais de um comando no `if` da primeira sintaxe, é preciso utilizar um bloco, demarcado por chaves.

O `else` é um complemento opcional para o `if`. Se utilizado, o comando será executado se a expressão retornar o valor `false` (zero). Suas duas sintaxes são:

```
if (expressão)
   comando;
else
   comando;
```

```
if (expressão):
   comando;
   . . .
   comando;
else
   comando;
   . . .
   comando;
endif;
```





A seguir, temos um exemplo do comando `if` utilizado com `else`:

```
if ($a > $b)
  $maior = $a;
else
  $maior = $b;
```

O exemplo acima coloca em $maior o maior valor entre $a e $b

Em determinadas situações é necessário fazer mais de um teste, e executar condicionalmente diversos comandos ou blocos de comandos. Para facilitar o entendimento de uma estrutura do tipo:

```
if (expressao1)
   comando1;
else
  if (expressao2)
    comando2;
  else
    if (expressao3)
      comando3;
    else
      comando4;
```

foi criado o comando, também opcional `elseif`. Ele tem a mesma função de um `else` e um `if` usados sequencialmente, como no exemplo acima. Num mesmo `if` podem ser utilizados diversos `elseif`'s, ficando essa utilização a critério do programador, que deve zelar pela legibilidade de seu script.

O comando `elseif` também pode ser utilizado com dois tipos de sintaxe. Em resumo, a sintaxe geral do comando `if` fica das seguintes maneiras:

```
if (expressao1)
  comando;
[ elseif (expressao2)
   comando; ]
[ else
   comando; ]


if (expressao1) :
  comando;
   . . .
  comando;
[ elseif (expressao2)
  comando;
   . . .
  comando; ]
[ else
  comando;
   . . .
  comando; ]
endif;
```





## switch

O comando `switch` atua de maneira semelhante a uma série de comandos `if` na mesma expressão. Frequentemente o programador pode querer comparar uma variável com diversos valores, e executar um código diferente a depender de qual valor é igual ao da variável. Quando isso for necessário, deve-se usar o comando `switch`. O exemplo seguinte mostra dois trechos de código que fazem a mesma coisa, sendo que o primeiro utiliza uma série de `if`'s e o segundo utiliza `switch`:

```
if ($i == 0)
   print “i é igual a zero”;
elseif ($i == 1)
   print “i é igual a um”;
elseif ($i == 2)
   print “i é igual a dois”;

switch ($i) {
case 0:
     print “i é igual a zero”;
     break;
```

```
case 1:
     print “i é igual a um”;
     break;
case 2:
     print “i é igual a dois”;
     break;
}
```

É importante compreender o funcionamento do `switch` para não cometer enganos. O comando `switch` testa linha a linha os cases encontrados, e a partir do momento que encontra um valor igual ao da variável testada, passa a executar todos os comandos seguintes, mesmo os que fazem parte de outro teste, até o fim do bloco. por isso usa-se o comando `break`, quebrando o fluxo e fazendo com que o código seja executado da maneira desejada. Veremos mais sobre o `break` mais adiante. Veja o exemplo:



```
switch ($i) {
case 0:
     print “i é igual a zero”;
case 1:
     print “i é igual a um”;
case 2:
     print “i é igual a dois”;
    }
```



No exemplo acima, se $i for igual a zero, os três comandos “print” serão executados. Se $i for igual a 1, os dois últimos “print” serão executados. O comando só funcionará da maneira desejada se $i for igual a 2.

Em outras linguagens que implementam o comando switch, ou similar, os valores a serem testados só podem ser do tipo inteiro. Em PHP é permitido usar valores do tipo string como elementos de teste do comando switch. O exemplo abaixo funciona perfeitamente:

```
switch ($s) {
case “casa”:
     print “A casa é amarela”;
case “arvore”:
     print “a árvore é bonita”;
case “lampada”:
     print “joao apagou a lampada”;
    }
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# comandos de repetição

## while

O `while` é o comando de repetição (laço) mais simples. Ele testa uma condição e executa um comando, ou um bloco de comandos, até que a condição testada seja falsa. Assim como o `if`, o `while` também possui duas sintaxes alternativas:

```
while (<expressao>)
   <comando>;
```

```
while (<expressao>):
   <comando>;
    . . .
   <comando>;
endwhile;
```

A expressão só é testada a cada vez que o bloco de instruções termina, além do teste inicial. Se o valor da expressão passar a ser `false` no meio do bloco de instruções, a execução segue até o final do bloco. Se no teste inicial a condição for avaliada como `false`, o bloco de comandos não será executado.

O exemplo a seguir mostra o uso do `while` para imprimir os números de 1 a 10:

```
$i = 1;
while ($i <=10)
  print $i++;
```

## do... while

O laço `do..while` funciona de maneira bastante semelhante ao `while`, com a simples diferença que a expressão é testada ao final do bloco de comandos. O laço `do..while` possui apenas uma sintaxe, que é a seguinte:

```
do {
   <comando>
   . . .
   <comando>
} while (<expressao>);
```

O exemplo utilizado para ilustrar o uso do `while` pode ser feito da seguinte maneira utilizando o `do..while`:

```
$i = 0;
do {
  print ++$i;
} while ($i < 10);
```

## for

O tipo de laço mais complexo é o `for`. Para os que programam em C, C++ ou Java, a assimilação do funcionamento do for é natural. Mas para aqueles que estão acostumados a linguagens como Pascal, há uma grande mudança para o uso do for. As duas sintaxes permitidas são:

```
for (<inicializacao>;<condicao>;<incremento>)
   <comando>;
```

```
for (<inicializacao>;<condicao>;<incremento>) :
   <comando>;
   . . .
   <comando>;
endfor;
```

As três expressões que ficam entre parênteses têm as seguintes finalidades:

Inicialização: comando ou sequencia de comandos a serem realizados antes do inicio do laço. Serve para inicializar variáveis.

Condição: Expressão booleana que define se os comandos que estão dentro do laço serão executados ou não. Enquanto a expressão for verdadeira (valor diferente de zero) os comandos serão executados.

Incremento: Comando executado ao final de cada execução do laço.

Um comando `for` funciona de maneira semelhante a um `while` escrito da seguinte forma:

```
<inicializacao>
while (<condicao>) {
comandos
...
<incremento>
}
```

# Quebra de fluxo

## Break

O comando `break` pode ser utilizado em laços de `do`, `for` e `while`, além do uso já visto no comando `switch`. Ao encontrar um `break` dentro de um desses laços, o interpretador PHP para imediatamente a execução do laço, seguindo normalmente o fluxo do script.

```
while ($x > 0) {
  ...
  if ($x == 20) {
```

```
        echo “erro! x = 20”;
        break;
    ...
}
```

No trecho de código acima, o laço `while` tem uma condição para seu término normal ($x <= 0), mas foi utilizado o break para o caso de um término não previsto no início do laço. Assim o interpretador seguirá para o comando seguinte ao laço.

## Continue

O comando `continue` também deve ser utilizado no interior de laços, e funciona de maneira semelhante ao `break`, com a diferença que o fluxo ao invés de sair do laço volta para o início dele. Vejamos o exemplo:

```
for ($i = 0; $i < 100; $i++) {
  if ($i % 2) continue;
  echo “ $i “;
}
```

O exemplo acima é uma maneira ineficiente de imprimir os números pares entre 0 e 99. O que o laço faz é testar se o resto da divisão entre o número e 2 é 0. Se for diferente de zero (valor lógico `true`) o interpretador encontrará um `continue`, que faz com que os comandos seguintes do interior do laço sejam ignorados, seguindo para a próxima iteração.

# 8. Funções

## Definindo funções

A sintaxe básica para definir uma função é:

```
function nome_da_função([arg1, arg2, arg3]) {
        Comandos;
        ... ;
        [return <valor de retorno>];
}
```



Qualquer código PHP válido pode estar contido no interior de uma função. Como a checagem de tipos em PHP é dinâmica, o tipo de retorno não deve ser declarado, sendo necessário que o programador esteja atento para que a função retorne o tipo desejado. É recomendável que esteja tudo bem documentado para facilitar a leitura e compreensão do código. Para efeito de documentação, utiliza-se o seguinte formato de declaração de função:

```
tipo function nome_da_funcao(tipo arg1, tipo arg2, ...);
```



Este formato só deve ser utilizado na documentação do script, pois o PHP não aceita a declaração de tipos. Isso significa que em muitos casos o programador deve estar atento ao tipos dos valores passados como parâmetros, pois se não for passado o tipo esperado não é emitido nenhum alerta pelo interpretador PHP, já que este não testa os tipos.

## Valor de retorno

Toda função pode opcionalmente retornar um valor, ou simplesmente executar os comandos e não retornar valor algum.

Não é possível que uma função retorne mais de um valor, mas é permitido fazer com que uma função retorne um valor composto, como listas ou arrays.

## Argumentos

É possível passar argumentos para uma função. Eles devem ser declarados logo após o nome da função, entre parênteses, e tornam-se variáveis pertencentes ao escopo local da função. A declaração do tipo de cada argumento também é utilizada apenas para efeito de documentação.

Exemplo:

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
function imprime($texto){
   echo $texto;
}

imprime("teste de funções");
```

## Passagem de parâmetros por referência

Normalmente, a passagem de parâmetros em PHP é feita por valor, ou seja, se o conteúdo da variável for alterado, essa alteração não afeta a variável original.



Exemplo:

```
function mais5($numero) {
   $numero += 5;
}

$a = 3;
mais5($a); //$a continua valendo 3
```

No exemplo acima, como a passagem de parâmetros é por valor, a função mais5 é inútil, já que após a execução sair da função o valor anterior da variável é recuperado. Se a passagem de valor fosse feita por referência, a variável $a teria 8 como valor. O que ocorre normalmente é que ao ser chamada uma função, o interpretador salva todo o escopo atual, ou seja, os conteúdos das variáveis. Se uma dessas variáveis for passada como parâmetro, seu conteúdo fica preservado, pois a função irá trabalhar na verdade com uma cópia da variável. Porém, se a passagem de parâmetros for feita por referência, toda alteração que a função realizar no valor passado como parâmetro afetará a variável que o contém.



Há duas maneiras de fazer com que uma função tenha parâmetros passados por referência: indicando isso na declaração da função, o que faz com que a pasagem de parâmetros sempre seja assim; e também na própria chamada da função. Nos dois casos utiliza-se o modificador "&". Vejamos um exemplo que ilustra os dois casos:

```
function mais5(&$num1, $num2) {
   $num1 += 5;
   $num2 += 5;
}

$a = $b = 1;
mais5($a, $b); /* Neste caso, só $num1 terá seu valor alterado, pois a passagem por referência está definida na declaração da função. */


mais5($a, &$b); /* Aqui as duas variáveis terão seus valores alterados. */
```

## Argumentos com valores pré-definidos (default)

Em PHP é possível ter valores *default* para argumentos de funções, ou seja, valores que serão assumidos em caso de nada ser passado no lugar do argumento. Quando algum parâmetro é declarado desta maneira, a passagem do mesmo na chamada da função torna-se opcional.

```
function teste($vivas = “testando”) {
   echo $vivas;
}

teste(); // imprime “testando”
teste(“outro teste”); // imprime “outro teste”
```



É bom lembrar que quando a função tem mais de um parâmetro, o que tem valor *default* deve ser declarado por último:

```
function teste($figura = circulo, $cor) {
  echo “a figura é um “, $figura, “ de cor “ $cor;
}

teste(azul);
/* A função não vai funcionar da maneira esperada, ocorrendo um erro no interpretador. A declaração correta é: */

function teste2($cor, $figura = circulo) {
  echo “a figura é um “, $figura, “ de cor “ $cor;
}

teste2(azul);

/* Aqui a funcao funciona da maneira esperada, ou seja, imprime o texto: “a figura é um círculo de cor azul” */
```



# Contexto

O contexto é o conjunto de variáveis e seus respectivos valores num determinado ponto do programa. Na chamada de uma função, ao iniciar a execução do bloco que contém a implementação da mesma é criado um novo contexto, contendo as variáveis declaradas dentro do bloco, ou seja, todas as variáveis utilizadas dentro daquele bloco serão eliminadas ao término da execução da função.

# Escopo

O escopo de uma variável em PHP define a porção do programa onde ela pode ser utilizada. Na maioria dos casos todas as variáveis têm escopo global. Entretanto, em funções definidas pelo usuário um escopo local é criado. Uma variável de escopo global não pode ser utilizada no interior de uma função sem que haja uma declaração.

```
Exemplo:
$vivas = “Testando”;

function Teste() {
  echo $vivas;
}

Teste();
```

O trecho acima não produzirá saída alguma, pois a variável $vivas é de escopo global, e não pode ser referida num escopo local, mesmo que não haja outra com nome igual que cubra a sua visibilidade. Para que o script funcione da forma desejada, a variável global a ser utilizada deve ser declarada.

```
Exemplo:
$vivas = “Testando”;

function Teste() {
  global $vivas;
  echo $vivas;
}

Teste();
```



Uma declaração “global” pode conter várias variáveis, separadas por vírgulas. Uma outra maneira de acessar variáveis de escopo global dentro de uma função é utilizando um array pré-definido pelo PHP cujo nome é $GLOBALS. O índice para a variável referida é o proprio nome da variável, sem o caracter $. O exemplo acima e o abaixo produzem o mesmo resultado:



```
Exemplo:
$vivas = "Testando";

function Teste() {
  echo $GLOBALS["vivas"]; // imprime $vivas
  echo $vivas; // não imprime nada
}

Teste();
```

# 9. Variáveis

## O modificador static

Uma variável estática é visível num escopo local, mas ela é inicializada apenas uma vez e seu valor não é perdido quando a execução do script deixa esse escopo. Veja o seguinte exemplo:

```
function Teste() {
  $a = 0;
  echo $a;
  $a++;
}
```



O último comando da função é inútil, pois assim que for encerrada a execução da função a variável $a perde seu valor. Já no exemplo seguinte, a cada chamada da função a variável $a terá seu valor impresso e será incrementada:

```
function Teste() {
  static $a = 0;
  echo $a;
  $a++;
}
```



O modificador static é muito utilizado em funções recursivas, já que o valor de algumas variáveis precisa ser mantido. Ele funciona da seguinte forma: O valor das variáveis declaradas como estáticas é mantido ao terminar a execução da função. Na próxima execução da função, ao encontrar novamente a declaração com static, o valor da variável é recuperado.

Em outras palavras, uma variável declarada como `static` tem o mesmo "tempo de vida" que uma variável global, porém sua visibilidade é restrita ao escopo local em que foi declarada e só é recuperada após a declaração.

Exemplo:

```
function Teste() {
  echo "$a";
  static $a = 0;
  $a++;
}
```

O exemplo acima não produzirá saída alguma. Na primeira execução da função, a impressão ocorre antes da atribuição de um valor à função, e portanto o conteúdo de $a é nulo (string vazia). Nas execuções seguintes da função Teste() a impressão ocorre antes da recuperação do valor de `$a`, e portanto nesse momento seu valor ainda é nulo. Para que a função retorne algum valor o modificador `static` deve ser utilizado.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# Variáveis Variáveis

O PHP tem um recurso conhecido como variáveis variáveis, que consiste em variáveis cujos nomes também são variáveis. Sua utilização é feita através do duplo cifrão ($$).

```
$a = "teste";
$$a = "Mauricio Vivas";

O exemplo acima e equivalente ao seguinte:

$a = "teste";
$teste = "Mauricio Vivas";
```

# Variáveis enviadas pelo navegador

Para interagir com a navegação feita pelo usuário, é necessário que o PHP possa enviar e receber informações para o software de navegação. A maneira de enviar informações, como já foi visto anteriormente, geralmente é através de um comando de impressão, como o *echo*. Para receber informações vindas do navegador através de um *link* ou um formulário html o PHP utiliza as informações enviadas através da URL. Por exemplo: se seu script php está localizado em `"http://localhost/teste.php3"` e você o chama com a url `"http://localhost/teste.php3?vivas=teste"`, automaticamente o PHP criará uma variável com o nome $vivas contendo a string "teste". Note que o conteúdo da variável está no formato `urlencode`. Os formulários html já enviam informações automaticamente nesse formato, e o PHP decodifica sem necessitar de tratamento pelo programador.

## URLencode

O formato `urlencode` é obtido substituindo os espaços pelo caracter "+" e todos os outros caracteres não alfa-numéricos (com exceção de "_") pelo caracter "%" seguido do código ASCII em hexadecimal.

Por exemplo: o texto "Testando 1 2 3 !!" em urlencode fica "Testando+1+2+3+%21%21"

O PHP possui duas funções para tratar com texto em urlencode. Seguem suas sintaxes:

```
string urlencode(string texto);
string urldecode(string texto);
```

Essas funções servem respectivamente para codificar ou decodificar um texto passado como argumento. Para entender melhor o que é um argumento e como funciona uma função, leia o tópico “funções”.

# Variáveis de ambiente

O PHP possui diversas variáveis de ambiente, como a $PHP_SELF, por exemplo, que contém o nome e o path do próprio arquivo. Algumas outras contém informações sobre o navegador do usuário, o servidor http, a versão do PHP e diversas informações. Para ter uma listagem de todas as variáveis e constantes de ambiente e seus respectivos conteúdos, deve-se utilizar a função `phpinfo()`.



# Verificando o tipo de uma variável

Por causa da tipagem dinâmica utilizada pelo PHP, nem sempre é possível saber qual o tipo de uma variável em determinado instantese não contar com a ajuda de algumas funções que ajudam a verificar isso. A verificação pode ser feita de duas maneiras:



## Função que retorna o tipo da variável

Esta função é a `gettype`. Sua assinatura é a seguinte:

```
string gettype(mixed var);
```

A palavra “mixed” indica que a variável var pode ser de diversos tipos.

A função `gettype` pode retornar as seguintes strings: `“integer”, “double”, “string”, “array”, “object” e “unknown type”.`

## Funções que testam o tipo da variável

São as funções `is_int, is_integer, is_real, is_long, is_float, is_string, is_array e is_object`. Todas têm o mesmo formato, seguindo modelo da assinatura a seguir:

```
int is_integer(mixed var);
```

Todas essas funções retornam `true` se a variável for daquele tipo, e `false` em caso contrário.

# Destruindo uma variável

É possível desalocar uma variável se ela não for usada posteriormente através da função `unset`, que tem a seguinte assinatura:

```
int unset(mixed var);
```

A função destrói a variável, ou seja, libera a memória ocupada por ela, fazendo com que ela deixe de existir. Se mais na frente for feita uma chamada á variável, será criada uma nova variável de mesmo nome e de conteúdo vazio, a não ser que a chamada seja pela função `isset`. Se a operação for bem sucedida, retorna `true`.



# Verificando se uma variável possui um valor

Existem dois tipos de teste que podem ser feitos para verificar se uma variável está setada: com a função `isset` e com a função `empty`.

## A função isset



Possui o seguinte protótipo:

```
int isset(mixed var);
```

E retorna `true` se a variável estiver setada (ainda que com uma string vazia ou o valor zero), e `false` em caso contrário.

## A função empty

Possui a seguinte assinatura:

```
int empty(mixed var);
```

E retorna `true` se a variável não contiver um valor (não estiver setada) ou possuir valor 0 (zero) ou uma string vazia. Caso contrário, retorna `false`.

# 10. Classes e Objetos

## Classe

Uma classe é um conjunto de variáveis e funções relacionadas a essas variáveis. Uma vantagem da utilização é poder usufruir do recurso de encapsulamento de informação. Com o encapsulamento o usuário de uma classe não precisa saber como ela é implementada, bastando para a utilização conhecer a interface, ou seja, as funções disponíveis. Uma classe é um tipo, e portanto não pode ser atribuída a uma variável. Para definir uma classe, deve-se utilizar a seguinte sintaxe:



```
class Nome_da_classe {
   var $variavel1;
   var $variavel2;
   function funcao1 ($parametro) {
    /* === corpo da função === */
   }
}
```

## Objeto



Como foi dito anteriormente, classes são tipos, e não podem ser atribuídas a variáveis. Variáveis do tipo de uma classe são chamadas de objetos, e devem ser criadas utilizando o operador `new`, seguindo o exemplo abaixo:

```
$variavel = new $nome_da_classe;
```

Para utilizar as funções definidas na classe, deve ser utilizado o operador "`->`", como no exemplo:

```
$variavel->funcao1(
```

## A variável $this

Na definição de uma classe, pode-se utilizar a variável `$this`, que é o próprio objeto. Assim, quando uma classe é instanciada em um objeto, e uma função desse objeto na definição da classe utiliza a variável `$this`, essa variável significa o objeto que estamos utilizando.

Como exemplo da utilização de classes e objetos, podemos utilizar a classe conta, que define uma conta bancária bastante simples, com funções para ver saldo e fazer um crédito.

```
class conta {
   var $saldo;
   function saldo() {
     return $this->saldo;
   }
   function credito($valor) {
     $this->saldo += $valor;
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
    }
}

$minhaconta = new conta;
$minhaconta->saldo();     // a variavel interna não foi
                          // inicializada, e não contém
                          // valor algum
$minhaconta->credito(50);
$minhaconta->saldo(); // retorna 50
```

# SubClasses

Uma classe pode ser uma extensão de outra. Isso significa que ela herdará todas as variáveis e funções da outra classe, e ainda terá as que forem adicionadas pelo programador. Em PHP não é permitido utilizar herança múltipla, ou seja, uma classe pode ser extensão de apenas uma outra.Para criar uma classe extendida, ou derivada de outra, deve ser utilizada a palavra reservada `extends`, como pode ser visto no exemplo seguinte:

```
class novaconta extends conta {
  var $numero;
  function numero() {
     return $this->numero;
  }
}
```

A classe acima é derivada da classe conta, tendo as mesmas funções e variáveis, com a adição da variável `$numero` e a função `numero()`.

# Construtores

Um construtor é uma função definida na classe que é automaticamente chamada no momento em que a classe é instanciada (através do operador `new`). O construtor deve ter o mesmo nome que a classe a que pertence. Veja o exemplo:

```
class conta {
   var $saldo;

   function conta () {
     $this.saldo = 0;
   }

   function saldo() {
     return $this->saldo;
   }
   function credito($valor) {
     $this->saldo += $valor;
   }
}
```

Podemos perceber que a classe conta agora possui um construtor, que inicializa a variável `$saldo` com o valor 0.

Um construtor pode conter argumentos, que são opcionais, o que torna esta ferramenta mais poderosa. No exemplo acima, o construtor da classe conta pode receber como argumento um valor, que seria o valor inicial da conta.

Vale observar que para classes derivadas, o construtor da classe pai não é automaticamente herdado quando o construtor da classe derivada é chamado.

# 12. Conclusões

A realização deste Projeto Supervisionado possibilitou o estudo da linguagem PHP, que se mostrou uma ferramenta poderosa e simples de utilizar na construção de sites para a World Wide Web dinâmicos, possibilitando uma maior interação com o usuário e a armazenagem das informações em Bancos de Dados.

Após a conclusão da aplicação, tornou-se claro que a combinação de scripts *server-side*, como é o PHP, com scripts *client-side*, como JavaScript, por exemplo, possibilita um maior aproveitamento dos recursos disponíveis para criar páginas dinâmicas, e no processo de criação deve-se ponderar bastante para concluir qual dos dois tipos de scripts deve ser utilizado para determinado fim.

Entre as linguagens de script *server-side*, PHP surgiu como uma ótima opção, por diversos motivos: o custo de aquisição, que não existe; a portabilidade, permitindo que uma aplicação seja desenvolvida em uma plataforma para ser executada em outra; a simplicidade, já que os scripts ficam no próprio código html, e possuem uma sintaxe bastante simples; a possibilidade de trabalhar com diversos bancos de dados e servidores http, além do grande número de funções pré-definidas, entre outras coisas.

Por esses e outros motivos, é possível afirmar que o estudo sobre PHP foi bastante enriquecedor, por ter produzido uma documentação em português para a linguagem e ter motivado o aluno a continuar se dedicando ao tema.

# 13. Bibliografia e Referências

A pesquisa foi baseada no manual de PHP, disponível em ***www.php.net***, e em diversos tutoriais disponíveis no site ***www.phpbuilder.com***. Esses dois endereços contém uma vasta documentação sobre a linguagem, além de endereços para listas de discussão, onde pode-se solicitar ajuda de programadores mais experientes.

Uma boa referência em português é a lista *"PHP para quem fala Português"*, que pode ser assinada no endereço ***www.egroups.com/group/php-pt/***.

Em inglês, além dos endereços citados acima, uma boa fonte é o site *PHPWizard*, que pode ser encontrado em ***www.phpwizard.net***.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# APÊNDICE 01 - Funções para tratamento de strings

## Funções relacionadas a HTML

### htmlspecialchars

```
string htmlspecialchars(string str);
```

Retorna a string fornecida, substituindo os seguintes caracteres:

- & para '&'
- " para '"'
- < para '<'
- > para '>'

### htmlentities

```
string htmlentities(string str);
```

Funciona de maneira semelhante ao comando anterior, mas de maneira mais completa, pois converte todos os caracteres da string que possuem uma representação especial em html, como por exemplo:

- º para 'º'
- ª para 'ª'
- á para 'á'
- ç para 'ç'

### nl2br

```
string nl2br(string str);
```

Retorna a string fornecida substituindo todas as quebras de linha (“\n”) por quebras de linhas em html (“<br>”).

```
Exemplo:
echo nl2br(“Mauricio\nVivas\n”);

Imprime:
Maurício<br>Vivas<br>
```

## get_meta_tags

```
array get_meta_tags(string arquivo);
```

Abre um arquivo html e percorre o cabeçalho em busca de “meta” tags, retornando num array todos os valores encontrados.

```
Exemplo:
No arquivo teste.html temos:
...
<head>
<meta name="author" content="jose">
<meta name="tags" content="php3 documentation">
...
</head><!-- busca encerra aqui -->
...

a execução da função:

get_meta_tags(“teste.html”);

retorna o array:

array(“author”=>”jose”,”tags”=>"php3 documentation");
```



## strip_tags

```
string strip_tags(string str);
```

Retorna a string fornecida, retirando todas as tags html e/ou PHP encontradas.

```
Exemplo:
strip_tags('<a href="teste1.php3">testando</a><br>');

Retorna a string “testando”
```



## urlencode

```
string urlencode(string str);
```

Retorna a string fornecida, convertida para o formato urlencode. Esta função é útil para passar variáveis para uma próxima página.

## urldecode

```
string urldecode(string str);
```

Funciona de maneira inversa a urlencode, desta vez decodificando a string fornecida do formato urlencode para texto normal.

# Funções relacionadas a arrays

## Implode e join

```
string implode(string separador, array partes);
string join(string separador, array partes);
```

As duas funções são idênticas. Retornam uma string contendo todos os elementos do array fornecido separados pela string também fornecida.

```
Exemplo:
$partes = array("a", "casa número", 13, "é azul");
$inteiro = join(" ",$partes);

$inteiro passa a conter a string:
“a casa número 13 é azul”
```



## split

```
array split(string padrao, string str, int [limite]);
```

Retorna um array contendo partes da string fornecida separadas pelo padrão fornecido, podendo limitar o número de elementos do array.



```
Exemplo:
$data = “11/14/1975”;
$data_array = split(“/”,$data);

O código acima faz com que a variável $data_array receba o valor:
array(11,14,1975);
```

## explode

```
array explode(string padrao, string str);
```

Funciona de maneira bastante semelhante à função `split`, com a diferença que não é possível estabelecer um limite para o número de elementos do array.

# Comparações entre strings

## similar_text

```
int similar_text(string str1, string str2, double [porcentagem]);
```

Compara as duas strings fornecidas e retorna o número de caracteres coincidentes. Opcionalmente pode ser fornecida uma variável, passada por referência *(ver tópico sobre funções)*, que receberá o valor percentual de igualdade entre as strings. Esta função é *case sensitive*, ou seja, maiúsculas e minúsculas são tratadas como diferentes.

```
Exemplo:

$num = similar_text("teste", "testando",&$porc);

As variáveis passam a ter os seguintes valores:

$num == 4; $porc == 61.538461538462
```



## strcasecmp

```
int strcasecmp(string str1, string str2);
```

Compara as duas strings e retorna 0 (zero) se forem iguais, um valor maior que zero se `str1 > str2`, e um valor menor que zero se `str1 < str2`. Esta função é *case insensitive*, ou seja, maiúsculas e minúsculas são tratadas como iguais.



## strcmp

```
int strcasecmp(string str1, string str2);
```

Funciona de maneira semelhante à função `strcasecmp`, com a diferença que esta é *case sensitive*, ou seja, maiúsculas e minúsculas são tratadas como diferentes.

## strstr

```
string strstr(string str1, string str2);
string strchr(string str1, string str2);
```

As duas funções são idênticas. Procura a primeira ocorrência de `str2` em `str1`. Se não encontrar, retorna uma string vazia, e se encontrar retorna todos os caracteres de `str1` a partir desse ponto.

```
Exemplo:

strstr("Mauricio Vivas", "Viv"); // retorna “Vivas”
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

### stristr

```
string strstr(string str1, string str2);
```

Funciona de maneira semelhante à função `strstr`, com a diferença que esta é *case insensitive*, ou seja, maiúsculas e minúsculas são tratadas como iguais.

### strpos

```
int strpos(string str1, string str2, int [offset] );
```

Retorna a posição da primeira ocorrência de `str2` em `str1`, ou zero se não houver. O parâmetro opcional `offset` determina a partir de qual caracter de str1 será efetuada a busca. Mesmo utilizando o offset, o valor de retorno é referente ao início de str1.



### strrpos

```
int strrpos(string haystack, char needle);
```

Retorna a posição da última ocorrência de `str2` em `str1`, ou zero se não houver.

# Funções para edição de strings



### chop

```
string chop(string str);
```

Retira espaços e linhas em branco do final da string fornecida.

```
Exemplo:
chop(“   Teste      \n  \n  “); // retorna “   Teste”
```

### ltrim

```
string ltrim(string str);
```

Retira espaços e linhas em branco do final da string fornecida.

```
Exemplo:
ltrim(“   Teste  \n  \n “); // retorna “Teste  \n  \n”
```

## trim

```
string trim(string str);
```

Retira espaços e linhas em branco do início e do final da string fornecida.

```
Exemplo:
trim("   Teste  \n  \n "); // retorna "Teste"
```

## strrev

```
string strrev(string str);
```

Retorna a string fornecida invertida.

```
Exemplo:

strrev("Teste"); // retorna "etseT"
```

## strtolower

```
string strtolower(string str);
```

Retorna a string fornecida com todas as letras minúsculas.

```
Exemplo:

strtolower("Teste"); // retorna "teste"
```

## strtoupper

```
string strtoupper(string str);
```

Retorna a string fornecida com todas as letras maiúsculas.

```
Exemplo:

strtolower("Teste"); // retorna "TESTE"
```

### ucfirst

```
string ucfirst(string str);
```

Retorna a string fornecida com o primeiro caracter convertido para letra maiúscula.

```
Exemplo:
ucfirst(“teste de funcao”); // retorna “Teste de funcao”
```

### ucwords

```
string ucwords(string str);
```

Retorna a string fornecida com todas as palavras iniciadas por letras maiúsculas.

```
Exemplo:
ucwords(“teste de funcao”); // retorna “Teste De Funcao”
```

### str_replace

```
string str_replace(string str1, string str2, string str3);
```

Altera todas as ocorrências de `str1` em `str3` pela string `str2`.

## Funções diversas

### chr

```
string chr(int ascii);
```

Retorna o caracter correspondente ao código ASCII fornecido.

### ord

```
int ord(string string);
```

Retorna o código ASCII correspontente ao caracter fornecido.

## echo

```
echo(string arg1, string [argn]... );
```

Imprime os argumentos fornecidos.

## print

```
print(string arg);
```

Imprime o argumento fornecido.

## strlen

```
int strlen(string str);
```

Retorna o tamanho da string fornecida.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# APÊNDICE 02 - Funções para tratamento de arrays

## Funções Genéricas

### Array

```
array array(...);
```

É a função que cria um array a partir dos parâmetros forncidos. É possível fornecer o índice de cada elemento. Esse índice pode ser um valor de qualquer tipo, e não apenas de inteiro. Se o índice não for fornecido o PHP atribui um valor inteiro sequencial, a partir do 0 ou do último índice inteiro explicitado. Vejamos alguns exemplos:

```
Exemplo 1
$teste = array("um", "dois","tr"=>"tres",5=>"quatro","cinco");

Temos o seguinte mapeamento:
0 => “um” (0 é o primeiro índice, se não houver um explicito)
1 => “dois” (o inteiro seguinte)
“tr” => “tres”
5 => “quatro” (valor explicitado)
6 => “cinco” (o inteiro seguinte ao último atribuído, e não o próximo valor, que seria 2)

Exemplo 2
$teste = array("um", 6=>"dois","tr"=>"tres",5=>"quatro","cinco");

Temos o seguinte mapeamento:
0 => “um”
6 => “dois”
“tr” => tres
5 => “quatro” (seria 7, se não fosse explicitado)
7 => “cinco” (seria 6, se não estivesse ocupado)
```

Em geral, não é recomendável utilizar arrays com vários tipos de índices, já que isso pode confundir o programador. No caso de realmente haver a necessidade de utilizar esse recurso, deve-se ter bastante atenção ao manipular os índices do array.

### range

```
array range(int minimo, int maximo);
```

A função range cria um array cujos elementos são os inteiros pertencentes ao intervalo fornecido, inclusive. Se o valor do primeiro parâmetro for maior do que o do segundo, a função retorna `false` (valor vazio).

### shuffle

```
void shuffle(array &arr);
```

Esta função “embaralha” o array, ou seja, troca as posições dos elementos aleatoriamente e não retorna valor algum.

### sizeof

```
int sizeof(array arr);
```

Retorna um valor inteiro contendo o número de elementos de um array. Se for utilizada com uma variável cujo valor não é do tipo array, retorna 1. Se a variável não estiver setada ou for um array vazio, retorna 0.



# Funções de “navegação”

Toda variável do tipo array possui um ponteiro interno indicando o próximo elemento a ser acessado no caso de não ser especificado um índice. As funções seguintes servem para modificar esse ponteiro, permitindo assim percorrer um array para verificar seu conteúdo (chaves e elementos).



### reset

```
mixed reset(array arr);
```

Seta o ponteiro interno para o primeiro elemento do array, e retorna o conteúdo desse elemento.

### end

```
mixed end(array arr);
```

Seta o ponteiro interno para o último elemento do array, e retorna o conteúdo desse elemento.

### next

```
mixed next(array arr);
```

Seta o ponteiro interno para o próximo elemento do array, e retorna o conteúdo desse elemento.

Obs.: esta não é uma boa função para determinar se um elemento é o último do array, pois pode retornar `false` tanto no final do array como no caso de haver um elemento vazio.

### prev

```
mixed prev(array arr);
```

Seta o ponteiro interno para o elemento anterior do array, e retorna o conteúdo desse elemento. Funciona de maneira inversa a `next`.

### pos

```
mixed pos(array arr);
```

Retorna o conteúdo do elemento atual do array, indicado pelo ponteiro interno.

### key

```
mixed key(array arr);
```

Funciona de maneira bastante semelhante a `pos`, mas ao invés de retornar o elemento atual indicado pelo ponteiro interno do array, retorna seu índice.

### each

```
array each(array arr);
```

Retorna um array contendo o índice e o elemento atual indicao pelo ponteiro interno do array. o valor de retorno é um array de quatro elementos, cujos índices são 0, 1, “key” e “value”. Os elementos de índices 0 e “key” armazenam o índice do valor atual, e os elementos de índices 1 e “value” contém o valor do elemento atual indicado pelo ponteiro.

Esta função pode ser utilizada para percorrer todos os elementos de um array e determinar se já foi encontrado o último elemento, pois no caso de haver um elemento vazio, a função não retornará o valor `false`. A função `each` só retorna `false` depois q o último elemento do array foi encontrado.

Exemplo:

```
/*função que percorre todos os elementos de um array e imprime seus índices e valores */
function imprime_array($arr) {
  reset($arr);
  while (list($chave,$valor) = each($arr))
    echo “Chave: $chave. Valor: $valor”;
}
```

# Funções de ordenação

São funções que servem para arrumar os elementos de um array de acordo com determinados critérios. Estes critérios são: manutenção ou não da associação entre índices e elementos; ordenação por elementos ou por índices; função de comparação entre dois elementos.

## sort

```
void sort(array &arr);
```

A função mais simples de ordenação de arrays. Ordena os elementos de um array em ordem crescente, sem manter os relacionamentos com os índices.

## rsort

```
void rsort(array &arr);
```

Funciona de maneir ainversa à função `sort`. Ordena os elementos de um array em ordem decrescente, sem manter os relacionamentos com os índices.



## asort

```
void asort(array &arr);
```

Tem o funcionamento bastante semelhante à função `sort`. Ordena os elementos de um array em ordem crescente, porém mantém os relacionamentos com os índices.



## arsort

```
void arsort(array &arr);
```

Funciona de maneira inversa à função `asort`. Ordena os elementos de um array em ordem decrescente e mantém os relacionamentos dos elementos com os índices.

## ksort

```
void ksort(array &arr);
```

Função de ordenação baseada nos índices. Ordena os elementos de um array de acordo com seus índices, em ordem crescente, mantendo os relacionamentos.

## usort

```
void usort(array &arr, function compara);
```

Esta é uma função que utiliza outra função como parâmetro. Ordena os elementos de um array sem manter os relacionamentos com os índices, e utiliza para efeito de comparação uma função definida pelo usuário, que deve comparar dois elementos do array e retornar 0, 1 ou –1, de acordo com qualquer critério estabelecido pelo usuário.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

## uasort

```
void uasort(array &arr, function compara);
```

Esta função também utiliza outra função como parâmetro. Ordena os elementos de um array e mantém os relacionamentos com os índices, utilizando para efeito de comparação uma função definida pelo usuário, que deve comparar dois elementos do array e retornar 0, 1 ou –1, de acordo com qualquer critério estabelecido pelo usuário.

## uksort

```
void uksort(array &arr, function compara);
```

Esta função ordena o array através dos índices, mantendo os relacionamentos com os elementos., e utiliza para efeito de comparação uma função definida pelo usuário, que deve comparar dois índices do array e retornar 0, 1 ou –1, de acordo com qualquer critério estabelecido pelo usuário.

# Sobre o autor da Apostila

*Esta apostila é de autoria* ***de*** **Maurício Vivas de Souza Barreto**
mauricio@cipsga.org.br **.....**

# GNU Free Documentation License

Version 1.1, March 2000



**0. PREAMBLE**

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

**1. APPLICABILITY AND DEFINITIONS**

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and

standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page.
For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

**2. VERBATIM COPYING**

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.



You may also lend copies, under the same conditions stated above, and you may publicly display copies.

**3. COPYING IN QUANTITY**

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.



If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto
adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

**4. MODIFICATIONS**

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

N. N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

O. If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made

by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

**5. COMBINING DOCUMENTS**

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.



In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

**6. COLLECTIONS OF DOCUMENTS**

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.



You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

**7. AGGREGATION WITH INDEPENDENT WORKS**

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document. If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

**8. TRANSLATION**

Translation is considered a kind of modification, so you may distribute translations of the
Document under the terms of section 4. Replacing Invariant Sections with translations requires especial permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

**9. TERMINATION**

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

**10. FUTURE REVISIONS OF THIS LICENSE**

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.



How to use this License for your documents.
To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document
> under the terms of the GNU Free Documentation License, Version 1.1
> or any later version published by the Free Software Foundation;
> with the Invariant Sections being LIST THEIR TITLES, with the
> Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
> A copy of the license is included in the section entitled "GNU
> Free Documentation License".



If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

# Comitê de Incentivo a Produção do Software Gratuito e Alternativo






Fundado em 29 de janeiro de 1999.

1ª Diretoria

Djalma Valois Filho
**Diretor Executivo**
dvalois@cxpostal.com

José Luiz Nunes Poyares
Diretor Administrativo

Paulo Roberto Ribeiro Guimarães
Diretor Institucional

CIPSGA
Rua Professora Ester de Melo, numero 202,
Parte, Benfica, Rio de Janeiro, RJ, CEP. 20930-010;
Telefone (Fax/Dados): 021-5564201;
e-mail: administracao@cipsga.org.br
CNPJ: 03179614-0001/70

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

Universidade da Região da Campanha
Centro de Ciências da Economia e Informática
Curso de Informática
Disciplina: Tópicos Especiais em Sistemas de Informação

# Apostila de PHP

Prof. Cristiano Cachapuz e Lima

Sumário

## 1 Introdução

PHP é uma sigla recursiva que significa *PHP HyperText Preprocessor*. O PHP é uma linguagem de código-fonte aberto, muito utilizada na Internet e especialmente criada para o desenvolvimento de aplicativos *Web*.

Note como isso é diferente de scripts CGI escritos em outras linguagens como Perl ou C --- ao invés de escrever um programa com um monte de comandos para imprimir HTML, você escreve um arquivo HTML com algum código inserido para fazer alguma coisa (nesse caso, imprimir um pouco de texto). O código PHP é delimitado por tags iniciais e finais que lhe permitem pular pra dentro e pra fora do "modo PHP".

A melhor coisa em usar PHP está no fato de ele ser extremamente simples para um iniciante, mas oferece muitos recursos para o programador profissional.

Para testar scripts PHP é necessário um servidor com suporte a esta tecnologia. Normalmente, o mais utilizado é o Apache. O banco de dados mais utilizado com os scripts PHP é o MySQL. Um exemplo de pacote pronto para execução de um ambiente Apache + PHP + MySQL é o EasyPHP (http://www.easyphp.org). Qualquer editor de textos pode ser usado para escrever os scripts PHP (ex. bloco de notas ou a ferramenta *free* MPS PHP Designer em http://www.mpsoftware.dk).

As páginas PHP devem ser salvas no diretório raiz do servidor. Para testes locais com o EasyPHP, essa pasta é c:\Arquivos de programas\EasyPHP\www. Para acessar a página, deve-se abrir o *browser* Internet Explorer e digitar-se o nome do domínio (http://127.0.0.1) e o nome da página com extensão .php. Quando o EasyPHP está sendo executado, aparece um ícone com uma letra "e" ao lado do relógio do Windows.

## 2 Exemplo de Script

Para criar o primeiro exemplo, digite o seguinte código-fonte no seu editor e salve com o nome de `teste.php` dentro do diretório raiz do servidor.

```
<html>
 <head>
  <title>Teste PHP</title>
 </head>
 <body>
 <?php echo "<p>Alô Mundo</p>"; ?>
 </body>
</html>
```

**Figura 1 – Primeiro script**

No *browser*, digite o endereço `http://127.0.0.1/teste.php` e veja o resultado. Veja também o código fonte da página (Exibir → Código fonte). É interessante notar que os comandos PHP não aparecem porque o servidor interpreta todos os scripts antes de enviar a página para o *browser*.

**O que PHP pode fazer ?**

Qualquer coisa. O PHP é focado para ser uma linguagem de script do lado do servidor, portanto, você pode fazer qualquer coisa que outro programa CGI pode fazer, como: coletar

dados de formulários, gerar páginas com conteúdo dinâmico ou enviar e receber cookies. Mas o PHP pode fazer muito mais.

Esses são os maiores campos onde os scripts PHP podem se utilizados:

- Script no lado do servidor (server-side). Este é o mais tradicional e principal campo de atuação do PHP. Você precisa de três coisas para seu trabalho. O interpretador do PHP (como CGI ou módulo), um servidor *web* e um browser. Basta rodar o servidor web conectado a um PHP instalado. Você pode acessar os resultados de seu programa PHP com um browser, visualizando a página PHP através do servidor web.
- Script de linha de comando. Você pode fazer um script PHP funcionar sem um servidor *web* ou *browser*. A única coisa necessária é o interpretador. Esse tipo de uso é ideal para script executados usando o `cron` ou o Agendador de Tarefas (no Windows). Esses scripts podem ser usados também para rotinas de processamento de texto.
- Escrevendo aplicações GUI no lado do cliente (client-side). O PHP não é (provavelmente) a melhor linguagem para produção de aplicações com interfaces em janelas, mas o PHP faz isso muito bem, e se você deseja usar alguns recursos avançados do PHP em aplicações no lado do cliente poderá utilizar o PHP-GTK para escrever esses programas. E programas escritos desta forma ainda serão independentes de plataforma. O PHP-GTK é uma extensão do PHP, não disponível na distribuição oficial. Se você está interessado no PHP-GTK, visite o site http://gtk.php.net.

O PHP pode ser utilizado na maioria dos sistemas operacionais, incluindo Linux, várias variantes Unix (incluindo HP-UX, Solaris e OpenBSD), Microsoft Windows, Mac OS X, RISC OS, e provavelmente outros. O PHP também é suportado pela maioria dos servidores web atuais, incluindo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape and iPlanet Servers, Oreilly Website Pro Server, Caudium, Xitami, OmniHTTPd, e muitos outros. O PHP pode ser configurado como módulo para a maioria dos servidores, e para os outros como um CGI comum.

Com o PHP, portanto, você tem a liberdade para escolher o sistema operacional e o servidor web. Do mesmo modo, você pode escolher entre utilizar programação estrutural ou programação orientada a objeto, ou ainda uma mistura deles. Mesmo não desenvolvendo nenhum recurso padrão de OOP (Object Oriented Programming, Programação Orientada a Objetos) na versão atual do PHP, muitas bibliotecas de código e grandes aplicações (incluindo a biblioteca PEAR) foram escritos somente utilizando OOP.

Com PHP você não está limitado a gerar somente HTML. As habilidades do PHP incluem geração de imagens, arquivos PDF e animações Flash (utilizando libswf ou Ming) criados dinamicamente. Você pode facilmente criar qualquer padrão texto, como XHTML e outros arquivos XML. O PHP pode gerar esses padrões e os salvar no sistema de arquivos, em vez de imprimi-los, formando um cache dinâmico de suas informações no lado do servidor.

Talvez a mais forte e mais significativa característica do PHP é seu suporte a uma ampla variedade de banco de dados. Escrever uma página que consulte um banco de dados é incrivelmente simples. Os seguintes bancos de dados são atualmente suportados:

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

| | | |
|---|---|---|
| Adabas D | Ingres | Oracle (OCI7 and OCI8) |
| dBase | InterBase | Ovrimos |
| Empress | FrontBase | PostgreSQL |
| FilePro (read-only) | mSQL | Solid |
| Hyperwave | Direct MS-SQL | Sybase |
| IBM DB2 | MySQL | Velocis |
| Informix | ODBC | Unix dbm |

Adicionalmente, o PHP suporta ODBC (Open Database Connection, ou Padrão Aberto de Conexão com Bancos de Dados), permitindo que você utilize qualquer outro banco de dados que suporte esse padrão mundial.

O PHP também tem suporte para comunicação com outros serviços utilizando protocolos como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (em Windows) e incontáveis outros. Você pode abrir sockets de rede e interagir diretamente com qualquer protocolo. O PHP também suporta o intercâmbio de dados complexos WDDX, utilizado em virtualmente todas as linguagens de programação para *web*. Falando de comunicação, o PHP implementa a instanciação de objetos Java e os utiliza transparentemente como objetos PHP. Você ainda pode usar sua extensão CORBA para acessar objetos remotos.

O PHP é extremamente útil em recursos de processamento de texto, do POSIX Estendido ou expressões regulares Perl até como interpretador para documentos XML. Para acessar e processar documentos XML, são suportados os padrões SAX e DOM. Você ainda pode usar nossa extensão XSLT para transformar documentos XML.

Utilizando o PHP no campo do e-commerce, você poderá usar as funções específicas para Cybescash, CyberMUT, Verysign Payflow Pro e CCVS, práticos sistemas de pagamento online.

Por último mas longe de terminar, temos também outras extensões interessantes: funções para o search engine mnoGoSearch, funções para Gateway IRC, vários utilitários de compressão (gzip, bz2), calendário e conversões de datas, tradução, etc.

## 3 Configuração (php.ini)

As configurações do PHP ficam armazenadas em um arquivo denominado php.ini e que é carregado cada vez que o servidor é iniciado. No Windows, ele fica na pasta c:\Windows.

Exemplo:

```
[PHP]

;;;;;;;;;;;
; WARNING ;
;;;;;;;;;;;
; This is the default settings file for new PHP installations.
; By default, PHP installs itself with a configuration suitable for
; development purposes, and *NOT* for production purposes.
; For several security-oriented considerations that should be taken
; before going online with your site, please consult php.ini-
recommended
```

```
   ; and http://php.net/manual/en/security.php.


   ;;;;;;;;;;;;;;;;;;;
   ; About this file ;
   ;;;;;;;;;;;;;;;;;;;
   ; This file controls many aspects of PHP's behavior.  In order for PHP
to
   ; read it, it must be named 'php.ini'.  PHP looks for it in the
current
   ; working directory, in the path designated by the environment
variable
   ; PHPRC, and in the path that was defined in compile time (in that
order).
   ; Under Windows, the compile-time path is the Windows directory.  The
   ; path in which the php.ini file is looked for can be overridden using
   ; the -c argument in command line mode.
   ;
   ; The syntax of the file is extremely simple.  Whitespace and Lines
   ; beginning with a semicolon are silently ignored (as you probably
guessed).
   ; Section headers (e.g. [Foo]) are also silently ignored, even though
   ; they might mean something in the future.
```

**Figura 2 – Trecho de exemplo do php.ini**

Através de modificações neste arquivo é possível alterar várias opções no comportamento do PHP. Todas as linhas iniciadas por ponto-e-vírgula são comentários.

## 4 Sintaxe Básica

Tags especiais indicam ao PHP onde estão os blocos de código. A tag de abertura é formada por um sinal de "menor que" (<), um sinal de interrogação (?) e a sigla php. A tag de fechamento é formada por um ponto interrogação (?) e sinal de "maior que" (>).

Ex:

```
<?php
...
?>
```

Exercício: digite o código da figura 1 e salve no diretório raiz do servidor Apache. Veja o resultado da página através de seu carregamento no *browser*.

O sinal de ponto-e-vírgula (;) indica o final de um comando (ver figura 1). A próxima figura mostra outro exemplo.

```
   <html>
   <head>
   <title>Teste PHP</title>
   </head>
   <body>

   <?php
   $a = 10;
   $b = 15;
```

```
    $c = $a + $b;
    echo "$a mais $b é igual a $c";
    ?>

    </body>
    </html>
```

**Figura 3 – Exemplo de código**

Os comentários de mais de uma linha no PHP são obtidos através de /* e */. Os comentários de apenas uma linha são obtidos através de //.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
/*
O código abaixo soma duas variáveis
e exibe o valor encontrado
*/
$a = 10;
$b = 15;
$c = $a + $b;
echo "$a mais $b é igual a $c";
?>

</body>
</html>
```

**Figura 4 – Exemplo de código com comentários de mais de uma linha**

Os comentários não aparecem no *browser*.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$a = 10; //A variável $a recebe o valor 10
$b = 15; //A variável $b recebe o valor 15
//A variável $c recebe o valor da soma
$c = $a + $b;
//O resultado obtido é exibido
echo "$a mais $b é igual a $c";
?>

</body>
</html>
```

**Figura 5 - Exemplo de código com comentários de uma linha**

Palavras-chave do PHP

| | | | | | |
|---|---|---|---|---|---|
| and | do | for | include | require | true |
| break | else | foreach | list | return | var |
| case | elseif | function | new | static | virtual |
| class | extends | global | not | switch | xor |
| continue | false | if | or | this | while |
| default | | | | | |

## 5 Variáveis

Variáveis armazenam valores. Pode-se referir a variáveis para obter seu valor ou para alterar seu conteúdo.

No PHP elas são representadas por um cifrão ($) mais o nome da variável. Os nomes de variáveis válidos são iniciados por letras ou por um subscrito ( _ ). Existe diferenciação entre nomes de variáveis maiúsculas e minúsculas.

Ex: $a, $_A, $_a

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$a = 10;
$A = 20;
echo "O valor de 'a' é $a e o de 'A' é $A";
?>

</body>
</html>
```

**Figura 6 – Exemplo de código com variáveis minúscula e maiúscula**

Quando a variável é declarada dentro de uma função, ela só estará disponível para o código desta função. O código a seguir gera um erro devido a essa característica.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

function soma($a)
{
  $b = $a + 5;
}

soma(10);

echo "o valor de 'b' é $b";
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
?>

</body>
</html>
```

**Figura 7 – Declaração de variável dentro de função**

Para evitar este tipo de problema, deve-se definir a variável como global. O código a seguir resolve o problema do código anterior. Compare os resultados dos dois scripts.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

function soma($a)
{
  global $b;
  $b = $a + 5;
}

soma(10);

echo "o valor de 'b' é $b";

?>

</body>
</html>
```

**Figura 8 – Declaração de variável global**

## 6 Tipos de dados

O PHP suporta vários tipos de dados:

Inteiro – Números inteiros (isto é, números sem ponto decimal)

Números de dupla precisão – Números reais (isto é, números que contêm um ponto decimal)

*String* – Texto entre aspas simples (´ ´) ou duplas (“ “)

Booleanos – armazenam valores verdadeiros ou falsos, usados em testes de condições

*Array* – Grupo de elementos do mesmo tipo

Objeto – Grupo de atributos e métodos

Recurso – Uma origem de dados externa

Nulo – Nenhum valor

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>
```

```
<?php

$a = True;

if ($a)
{
  echo "Verdadeiro";
}
else
{
  echo "Falso";
}

?>

</body>
</html>
```

**Figura 9 – Código com dados booleanos**

Teste o código anterior alterando o valor da variável para False.

Pode-se armazenar valores inteiros, positivos ou negativos. Pode-se usar também valores hexadecimais.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$a = 0x1A; //Coresponde ao decimal 26
$b = -16;
$c = $a + $b;

echo "a + b = $c";

?>

</body>
</html>
```

**Figura 10 – Código com variáveis hexadecimal e valor negativo**

Os valores de ponto flutuante são representados através de ponto ( . ).

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$preco = 11.90;
```

```
$soma = $preco * 4;

echo "Quatro revistas W custam R$ $soma<br>";

?>

</body>
</html>
```

**Figura 11 – Código com variável de ponto flutuante**

As strings são armazenadas dentro de aspas duplas ( “ ) ou simples ( ‘ ).

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$texto1 = 'Esse é o primeiro texto.<br>';
$texto2 = "Esse é o segundo texto.";

echo $texto1;
echo $texto2;

?>

</body>
</html>
```

**Figura 12 – Código com strings entre aspas simples e duplas**

As variáveis do tipo matriz ou array permitem o armazenamento de diversos elementos referenciados por uma mesma referência. Maiores detalhes serão vistos na seção 14.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$frutas = array(
  1 => "Laranja",
  2 => "Maçã",
  3 => "Uva");

echo "<li> $frutas[1]<br>";
echo "<li> $frutas[2]<br>";
echo "<li> $frutas[3]<br>";

?>
```

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

**Figura 13 – Código com matriz**

## 7 Constantes

Constantes são identificadores para valores simples. O seu conteúdo não muda durante a execução do código. Elas são criadas com a função `define` e, por convenção, são escritas com letras maiúsculas e não usam o cifrão no início.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
define("CONSTANTE", "Alô mundo.");
echo CONSTANTE;
?>

</body>
</html>
```

**Figura 14 – Código com constante**

O PHP implementa algumas constantes, a maioria são matemáticas. O código seguinte demonstra o uso da constante `M_PI`.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

function calculaAreaCirculo($raio)
{
  return M_PI * pow($raio, 2);
}

$meuRaio = 5;
$area = calculaAreaCirculo($meuRaio);

echo "<b>Raio</b> = $meuRaio<br>";
echo "<b>Área</b> = $area";

?>

</body>
</html>
```

**Figura 15 – Código com a constante M_PI**

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

## 8 Expressões

Tudo que tem um valor pode ser considerado uma expressão. O código a seguir demonstra na prática.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$b = ($a = 5);
echo "O valor de 'b' é $b";

?>

</body>
</html>
```

**Figura 16 – Código com uso de expressão**

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$b = $a = 5;
echo "O valor de 'b' é $b";

?>

</body>
</html>
```

**Figura 17 – Variação do código anterior**

Expressões de comparação retornam valores booleanos, sendo vazio representando verdadeiro e um representando falso. As expressões de comparação são usadas em declarações condicionais para determinar se um bloco de código será executado ou não.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$valor = (5 < 10);
echo "O valor da expressão '5 > 10' é $valor";
```

```
?>

</body>
</html>
```

**Figura 18 – Código com expressão de comparação**

## 9 Operadores

São usados para efetuarem operações sobre as variáveis e constantes. Os operadores do PHP são:

+ soma
- subtração
* multiplicação
/ divisão
^ exponenciação
% módulo, resto da divisão
++ acrescenta um a uma variável
-- subtrai um de uma variável
+= soma um valor a uma variável e lhe atribui o resultado

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$x = 2;
echo($x + 2);
echo "<br>";
$x = 2;
echo(5 - $x);
echo "<br>";
$x = 4;
echo($x * 5);
echo "<br>";
$x = 15;
echo($x / 5);
echo "<br>";
$x = 10;
echo($x % 8);
echo "<br>";
$x = 5;
$x++;
echo($x);
echo "<br>";
$x = 5;
$x--;
echo($x);
echo "<br>";
$x = 8;
echo($x);
```

```
echo "<br>";
$x = 8;
$x = $x + 10;
echo($x);
echo "<br>";
$x = 8;
$x += 10;
echo($x);
?>

</body>
</html>
```

**Figura 19 – Código com diversas operações matemáticas**

Há também os operadores de comparação. Uma comparação sempre gera um dos dois valores possíveis: vazio, que corresponde a falso, e 1, que corresponde a verdadeiro.

= = é igual a
! = não é igual a
> é maior que
< é menor que
>= é maior ou igual a
<= é menor ou igual a

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$x = 5;

$resultado = ($x == 8);

if($resultado == 1)
{
  echo "verdadeiro";
}
else
{
  echo "falso";
}

echo "<br>";

$x = 5;

$resultado = ($x != 8);

if($resultado == 1)
{
  echo "verdadeiro";
}
else
```

```
{
  echo "falso";
}

echo "<br>";

$x = 5;

$resultado = ($x > 8);

if($resultado == 1)
{
  echo "verdadeiro";
}
else
{
  echo "falso";
}

echo "<br>";

$x = 5;

$resultado = ($x > 8);

if($resultado == 1)
{
  echo "verdadeiro";
}
else
{
  echo "falso";
}

echo "<br>";

$x = 5;

$resultado = ($x >= 8);

if($resultado == 1)
{
  echo "verdadeiro";
}
else
{
  echo "falso";
}

echo "<br>";

$x = 5;

$resultado = ($x <= 8);

if($resultado == 1)
{
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
  echo "verdadeiro";
}
else
{
  echo "falso";
}
?>

</body>
</html>
```

**Figura 20 – Código com operadores de comparação**

### Operadores lógicos

and ou && - operador lógico "e", apenas retornando verdadeiro quando as duas condições envolvidas no teste forem verdadeiras

or ou || operador lógico "ou", retornando verdadeiro quando uma ou as duas condições envolvidas no teste forem verdadeiras

! operador lógico "não", invertendo o resultado de um teste

xor – operador lógico "ou exclusivo" que determina se uma de duas condições é verdadeira mas não ambas. Se ambas forem verdadeiras, o teste final será falso

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$x = 6;
$y = 3;

$resultado = ($x < 10 && $y > 1);

if($resultado == 1)
{
  echo "verdadeiro";
}
else
{
  echo "falso";
}

echo "<br>";

$x = 6;
$y = 3;

$resultado = ($x == 5 || $y == 5);

if($resultado == 1)
{
  echo "verdadeiro";
}
```

```
else
{
  echo "falso";
}

echo "<br>";

$x = 6;
$y = 3;

$resultado = (!($x == $y));

if($resultado == 1)
{
  echo "verdadeiro";
}
else
{
  echo "falso";
}
?>

</body>
</html>
```

**Figura 21 – Código com operadores lógicos**





## 10 Estruturas de Controle

No PHP, as estruturas de controle são formadas por declarações condicionais e de *looping*:

if – executa uma ação se uma condição for atendida. O bloco de comandos a ser executado deve ser escrito entre chaves;

else – pode-se colocar um conjunto de comandos alternativos caso o teste do if seja negativo. A declaração else deve vir logo após o bloco de código relacionado ao if (ver figura 24). O comando if também pode ser usado após a declaração else (figura 25).

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$x = 20;

if ($x > 10)
{
  echo("O valor da variável é maior que 10.");
}
```

```
?>

</body>
</html>
```

**Figura 22 – Código com declaração condicional if verdadeiro**

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$x = 5;

if ($x > 10)
{
  echo("O valor da variável é maior que 10.");
}
?>


</body>
</html>
```

**Figura 23 – Código com declaração condicional if falso**

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$x = 5;

if ($x > 10)
{
  echo("O valor da variável é maior que 10.");
}
else
{
  echo("O valor da variável é menor que 10.");
}
?>

</body>
</html>
```

**Figura 24 – Código com declaração condicional if e else**

```
<html>
<head>
<title>Teste PHP</title>
</head>
```

```
<body>

<?php
$cor = "branco";

if ($cor == "vermelho")
{
  echo("A variável contém o valor 'vermelho'.");
}
else if ($cor == "azul")
{
  echo("A variável contém o valor 'azul'.");
}
else if ($cor == "amarelo")
{
  echo("A variável contém o valor 'amarelo'.");
}
else
{
  echo("O valor da variável não foi identificado.");
}
?>

</body>
</html>
```

**Figura 25 – Código com declaração condicional if e else if**

switch / case – forma de testar uma dentre várias possibilidades. A declaração default executa caso nenhuma das opções for verdadeira (figura 26). A declaração break faz com que o restante do código não seja executado caso o teste seja verdadeiro.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$d = getdate();

switch ($d['wday'])
{
case 5:
  echo("Finalmente Sexta");
  break;
case 6:
  echo("Super Sábado");
  break;
case 0:
  echo("Domingo Sonolento");
  break;
default:
  echo("Estou esperando pelo fim da semana");
}
?>
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

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

**Figura 26 – Código com declaração condicional switch e case**

for – estrutura de looping que executa um bloco de código quantas vezes for indicado em uma variável. Deve-se definir a variável que será testada no looping, uma condição de teste e o incremento (ou decremento) da variável de controle.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

for ($i = 1; $i < 10; $i++)
{
  echo("Linha $i <br>");
}

?>

</body>
</html>
```

**Figura 27 – Código com looping definido pelo comando if**

while – estrutura de looping que não necessita de um número determinado de iterações. Ele é executado enquanto uma condição for verdadeira.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$i = 1;

while ($i < 10000)
{
  echo($i);
  $i *= 2;
  echo(" vezes 2 é igual a $i <br>");
}

?>

</body>
</html>
```

**Figura 28 – Código com declaração condicional while**

do-while– outra forma de looping que executa um bloco de código, testa uma condição e repete novamente o bloco de código (ou não).

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
$i = 1;

do
{
  echo ("Linha $i <br>");
  $i++;
}
while ($i < 10)

?>

</body>
</html>
```

**Figura 29 - Código com declaração condicional do-while**





## 11 Funções

Uma função é um bloco de código reutilizável que é executado devido a um evento ou pela chamada de outra função. Deve-se usar a declaração function para criar uma função. Os parâmetros usados pela função são declarados entre parênteses. Os comandos a serem executados pela função devem estar entre chaves (figura 30). A declaração return retorna um valor quando a função é chamada. Esta declaração não é necessária se a função não retorna nenhum valor.

Para se chamar uma função, deve-se escrever seu nome e indicar os parâmetros entre parênteses.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
function soma($valor1, $valor2)
{
  $resultado = $valor1 + $valor2;
  return ($resultado);
}


$x = soma(7, 8);
```

```
echo($x);
?>

</body>
</html>
```

**Figura 30 – Exemplo de uma função**

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
function escreveTexto()
{
  echo("Já sei criar funções!");
}

escreveTexto();
?>

</body>
</html>
```

**Figura 31 – Segundo exemplo de declaração de função**





## 12 Classes e Objetos

Uma classe é uma coleção de atributos e métodos. O código a seguir define uma classe chamada CarrinhoDeCompras, que é uma matriz associativa com os artigos do carrinho e duas funções: uma para adicionar e outra para remover os itens.

Classes são tipos, isto é, "rascunhos" para a criação de objetos. Deve-se utilizar o operador new para criar uma variável do tipo CarrinhoDeCompras.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
class CarrinhoDeCompras
{
    var $items;  // Items no carrinho de compras

    // Adiciona $num artigos do $artnr ao carrinho

    function adiciona_item ($artnr, $num)
    {
        $this->items[$artnr] = $num;
    }
```

```
    // Retira $num artigos do $artnr do carrinho

    function remove_item ($artnr, $num)
    {
        if ($this->items[$artnr] > $num) {
            $this->items[$artnr] -= $num;
            return true;
        } else {
            return false;
        }
    }
}
?>

</body>
</html>
```

**Figura 32 – Código de definição de uma classe**

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php
class CarrinhoDeCompras
{
    var $itens;  // Itens no carrinho de compras

    // Adiciona $num artigos do $artnr ao carrinho

    function adiciona_item ($artnr, $num)
    {
        $this->itens[$artnr] = $num;
    }

    // Retira $num artigos do $artnr do carrinho

    function remove_item ($artnr, $num)
    {
        if ($this->itens[$artnr] > $num) {
            $this->itens[$artnr] -= $num;
            return true;
        } else {
            return false;
        }
    }
}

$carrinho = new CarrinhoDeCompras;
$carrinho->adiciona_item("Banana", 12);
?>

</body>
</html>
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

**Figura 33 – Definição de classe e instanciação de um objeto do tipo CarrinhoDeCompras**

## 13 Referências

Referências, em PHP, significa acessar o mesmo conteúdo de variável através de vários nomes. Isto não é parecido como os ponteiros em C: aqui temos apelidos numa tabela simbólica. Note que no PHP nomes de variáveis e conteúdo de variável são tratados diferentemente, então um mesmo conteúdo pode ter nomes diferentes. A analogia mais próxima é a dos arquivos e seus nomes em sistemas UNIX: nomes de variáveis são o caminho completo dos arquivos, enquanto o conteúdo da variável são os dados desse arquivo. Assim, referências pode ser tomadas como os links físicos dentro do sistema de arquivos UNIX.

No PHP, pode-se acessar valores de variáveis através de diferentes nomes. Pode-se usar o sinal de igual seguido do sinal de "e comercial" (&).

Referências PHP permitem fazer duas variáveis se referirem ao mesmo conteúdo. Ou seja:

```
<?php
$a =& $b
?>
```

aqui `$a` e `$b` apontam para a mesma variável.

> **Nota:** `$a` e `$b` são completamente iguais aqui, mas não porque `$a` está apontando para `$b` ou vice versa, mas sim que `$a` e `$b` apontam para o mesmo lugar.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$a =& $b;

$b = 100;

echo $a;

?>

</body>
</html>
```

**Figura 34 – Código de exemplo de atribuição de valor a duas variáveis (a e b)**

O comando unset remove uma referência previamente declarada, mas ela mantém o último valor recebido.

```
<html>
<head>
```

```
<title>Teste PHP</title>
</head>
<body>

<?php

$a =& $b;

$b = 100;

unset($b);

$b = 200;

echo $a;

?>

</body>
</html>
```

**Figura 35 – Código de remoção de uma referência**

## 14 Matrizes

Matrizes são variáveis que armazenam mais de um valor simultaneamente. Uma matriz no PHP é atualmente um mapa ordenado. Um mapa é um tipo que relaciona *valores* para *chaves*. Este tipo é otimizado de várias maneiras, então você pode usá-lo como um array real, ou uma lista (vetor), *hashtable* (que é uma implementação de mapa), dicionário, coleção, pilha, fila, etc.

As referências aos elementos da matriz podem ser declaradas como valores numéricos ou strings (figura 37).

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$funcionarios = array(0 => "José",
                      1 => "João",
                      2 => "Maria",
                      3 => "Pedro",
                      4 => "Carla");

echo "<b>Funcionários</b>";
echo "<ul>";
echo "<li>" . $funcionarios[0];
echo "<li>" . $funcionarios[1];
echo "<li>" . $funcionarios[3];
echo "</ul><p>";
echo "<b>Funcionárias</b>";
```

```
echo "<ul>";
echo "<li>" . $funcionarios[2];
echo "<li>" . $funcionarios[4];
echo "</ul>";

?>

</body>
</html>
```

**Figura 36 – Código com declaração de matriz**

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$siglas = array("SP" => "São Paulo",
                "RJ" => "Rio de Janeiro",
                "MG" => "Minas Gerais");

echo $siglas["SP"];

?>

</body>
</html>
```

**Figura 37 – Código com declaração de matriz e referência através de string**

## 15 Inclusão de Arquivos

O comando `include` permite a inclusão de outros arquivos php dentro do script que está sendo executado. Pode-se criar uma função que imprime a data atual e pode-se reusá-lo sem precisar reescrever o código cada vez que for necessário. No exemplo a seguir, pode-se chamar o primeiro script de cabecalho.php e o próximo script o inclui através do comando include.

```
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?php

$meses = array(1  => "Janeiro",
               2  => "Fevereiro",
               3  => "Março",
```

```
                4  => "Abril",
                5  => "Maio",
                6  => "Junho",
                7  => "Julho",
                8  => "Agosto",
                9  => "Setembro",
                10 => "Outubro",
                11 => "Novembro",
                12 => "Dezembro");

$hoje = getdate();
$dia = $hoje["mday"];
$mes = $hoje["mon"];
$nomeMes = $meses[$mes];
$ano = $hoje["year"];

echo "Olá. Hoje é dia $dia de $nomeMes de $ano."

?>

</body>
</html>
```

**Figura 38 – Script que é salvo com o nome de cabecalho.php**

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
include("cabecalho.php");
?>

</body>
</html>
```

**Figura 39 – Código com inclusão de arquivo externo chamado cabecalho.php**

## 16 Cookies

Cookies são formas de armazenar informações a respeito de uma sessão dentro do disco rígido do usuário cliente. O comando setcookie armazena um cookie com as informações que se desejam recuperar em seguida. Quando não for declarado um tempo de vida, o cookie se auto-destrói quando a sessão é encerrada (quando o browser for fechado).

```
<?php

if (isset($HTTP_POST_VARS['usuario'])) {

  $user = $HTTP_POST_VARS['usuario'];

  setcookie("usuario", $user);
  $mensagem = "Usuário $user conectado.<p>";
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
}
else
{

  $mensagem = "Digite o seu nome de usuário<p>";

}

?>
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?
echo $mensagem;
?>

<form method="post" action="teste.php">
Nome de Usuário: <input type="text" name="usuario">
<br>
<input type="submit" value="Enviar">
</form>

</body>
</html>
```

**Figura 40 – Código que cria um cookie com o nome do usuário**

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php

$user = $_COOKIE["usuario"];

echo "O usuário $user está conectado.";

?>

</body>
</html>
```

**Figura 41 – Código que recupera os dados do cookie criado anteriormente**

O código a seguir demonstra o uso de um cookie com "tempo de vida" definido em 3600 segundos, isto é, uma hora. Após uma hora decorrida de sua criação, ele é removido.

```
<?php

if (isset($_POST['usuario'])) {
```

```
  $user = $_POST['usuario'];

  setcookie("usuario", $user, time() + 3600); // Expira em uma hora

  $mensagem = "Usuário $user conectado.<p>";

}
else
{

  $mensagem = "Digite o seu nome de usuário<p>";

}

?>
<html>
<head>
<title>Teste PHP</title>
</head>
<body>

<?
echo $mensagem;
?>

<form method="post" action="teste.php">
Nome de Usuário: <input type="text" name="usuario">
<br>
<input type="submit" value="Enviar">
</form>

</body>
</html>
```

**Figura 42 – Código que cria um cookie com o nome do usuário que dura uma hora**

## 17 Parâmetros

O uso de parâmetros facilita a programação porque permite a passagem de dados entre o browser e o script ou entre scripts. A passagem de parâmetros entre o browser e o script é feita dentro da URL, por exemplo e é manipulada pela função $_GET.

Nesse exemplo a seguir, cada um dos links envia um valor diferente para a página que é aberta (teste.php). Para enviar um parâmetro, a sintaxe inclui um sinal de interrogação, o nome da variável, um sinal de igual e o valor da variável.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php

if (isset($_GET["valor"]))
```

```
{
  $valor = $_GET["valor"];
  echo "Você clicou no link $valor <p>";
}
else
{
  echo "Clique em um dos links abaixo:<p>";
}

?>

<a href="teste.php?valor=1">link 1</a><br>
<a href="teste.php?valor=2">link 2</a><br>
<a href="teste.php?valor=3">link 3</a><br>
<a href="teste.php?valor=4">link 4</a><br>
<a href="teste.php?valor=5">link 5</a><br>

</body>
</html>
```

**Figura 43 – Código com passagem de parâmetro**

Caso exista necessidade de se passar mais de um parâmetro, deve-se separá-los através de “e comercial” (&), conforme figura 44.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
if (isset($_GET["nome"]) && isset($_GET["sobrenome"])) {
  $nome = $_GET["nome"];
  $sobrenome = $_GET["sobrenome"];
  echo "O nome selecionado foi $nome $sobrenome<p>";
}
else
{
  echo "Selecione um nome<p>";
}

?>

<a href="teste.php?nome=Pedro&sobrenome=Silva">Pedro Silva</a><br>
<a href="teste.php?nome=Maria&sobrenome=Santos">Maria Santos</a><br>

</body>
</html>
```

**Figura 44 – Código com passagem de mais de um parâmetro**

## 18 Formulários

Os valores enviados através de um formulário podem ser recuperados pela variável pré-definida $_POST. Através dela é possível obter os dados que foram enviados através do

método POST do HTML, bastando indicar o nome do campo do formulário. No comando action do formulário, deve-se indicar a página PHP que irá receber os valores. O mesmo documento pode conter o código e o formulário (figura 45).

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php

if (isset($_POST["pnome"]) && isset($_POST["snome"]))
{
  $pnome = $_POST["pnome"];
  $snome = $_POST["snome"];

  echo "Olá $pnome $snome.<p>";
}
else
{
  echo "Digite o seu nome.<p>";
}

?>

<form method="post" action="teste.php">
Primeiro Nome: <input type="text" name="pnome">
<br>
Sobrenome: <input type="text" name="snome">
<br><br>
<input type="submit" value="Enviar">
</form>

</body>
</html>
```

**Figura 45 – Código com formulário enviando dados através do método POST**

Se for usado o método GET, os dados podem ser visualizados na URL do browser. Para recuperar estes dados, deve-se usar a variável pré-definida $_GET. Executar os códigos das figuras 45 e 46 e analisar o comportamento do browser.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php

if (isset($_GET["pnome"]) && isset($_GET["snome"]))
{
  $pnome = $_GET["pnome"];
  $snome = $_GET["snome"];
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
  echo "Olá $pnome $snome.<p>";
}
else
{
  echo "Digite o seu nome.<p>";
}

?>

<form method="get" action="teste.php">
Primeiro Nome: <input type="text" name="pnome">
<br>
Sobrenome: <input type="text" name="snome">
<br><br>
<input type="submit" value="Enviar">
</form>

</body>
</html>
```

**Figura 46 – Código com formulário enviando dados através do método GET**

## 19 Uploads

O PHP permite que sejam enviados arquivos para o servidor. Deve-se modificar o arquivo de configuração (php.ini):

```
file_uploads   = On
upload_tmp_dir ="C:\Arquivos de programas\EasyPHP\tmp\"
upload_max_filesize = 2M
```

Para o próximo exemplo, o diretório de upload deve ser trocado para C:\temp. É necessário reiniciar o servidor a cada modificação em algum arquivo de configuração.

Todas as informações sobre o arquivo enviado ficam armazenadas na variável $_FILES. O comando que trata o envio do arquivo é `move_uploaded_file`. No exemplo a seguir, o usuário envia um arquivo de no máximo 30 Kb.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php

if (isset($_FILES['arquivo']['name'])){

  $uploaddir = 'c:\\temp\\';
  $arquivo = $uploaddir. $_FILES['arquivo']['name'];

  if (move_uploaded_file($_FILES['arquivo']['tmp_name'], $arquivo)) {
    print "O arquivo foi gravado com sucesso.";
```

```
  }
  else
  {
    print "Erro. O arquivo não foi enviado.";
  }
}
?>

<form enctype="multipart/form-data" action="teste.php" method="POST">
<input type="hidden" name="MAX_FILE_SIZE" value="30000">
Enviar este arquivo: <input name="arquivo" type="file">
<input type="submit" value="Envia Arquivo">
</form>

</body>
</html>
```

**Figura 47 – Código com upload de arquivo e armazenamento na pasta C:\temp**

## 20 Envio de e-mails

O PHP permite que se enviem e-mails de forma automatizada. Para isso, deve-se ajustar o arquivo de configuração (php.ini) para se indicar o servidor SMTP:

```
[mail function]
SMTP           =     localhost   ;for win32 only
sendmail_from  =     me@localhost.com  ;for win32 only
;sendmail_path =                       ;for unix only, may supply
arguments as well (default is 'sendmail -t -i')
```

A opção SMTP indica o endereço ou número IP do servidor SMTP. A opção sendmail_from indica o endereço do remetente da mensagem. É necessário reiniciar o servidor a cada modificação em algum arquivo de configuração. O próximo exemplo envia uma mensagem para o destinatário.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
$destinatario = "cristiano@urcamp.tche.br";
$assunto      = "Como enviar e-mais via PHP";
$mensagem    = "
<h2>Envio de e-mails via PHP</h2>

<p>Depois que o servidor está configurado, é muito simples enviar e-mails
com o PHP, usando apenas a função mail(). Você deve indicar como
parâmetros o destinatário, o assunto, e a mensagem. Para enviar
cabeçalhos adicionais, como informações sobre o formato da mensagem, há
um quarto parâmetro.</p>
";
$cabecalho  = "
MIME-Version: 1.0\r\n
Content-type: text/html; charset=iso-8859-1\r\n";
```

```
mail($destinatario, $assunto, $mensagem, $cabecalho);

echo "e-mail enviado com sucesso";
?>

</body>
</html>
```

**Figura 48 – Código com envio de e-mail**



## 21 Introdução ao MySQL



O MySQL é o gerenciador de banco de dados mais usado com o PHP. Existem muitas funções pré-definidas para manipulação de conexões com bancos de dados.

A função mysql_connect tenta uma conexão com um servidor MySQL. Deve-se passar como parâmetros: o nome do servidor (ou número IP) onde o MySQL está sendo executado, o nome de usuário e a senha deste usuário. O comando alternativo die trata um possível fracasso na conexão.

A função mysql_selectdb seleciona qual base será selecionada dentro do banco de dados que foi conectado. O comando alternativo die trata um possível fracasso na seleção da base, podendo ser incluída uma mensagem customizada.



A função mysql_query faz consultas à base previamente selecionada. Deve-se passar, como parâmetros, os comandos SQL apropriados. Novamente, o comando alternativo die pode tratar um não sucesso na consulta.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
$link = mysql_connect("127.0.0.1", "root", "")
  or die("Não foi possível conectar");

mysql_select_db("teste")
  or die("Não foi possível selecionar o banco de dados");

$consulta = "SELECT * FROM Clientes";
$resultado = mysql_query($consulta)
  or die("Falha na execução da consulta");

echo "Consulta executada com sucesso";
?>

</body>
</html>
```

**Figura 49 – Código com conexão a um banco de dados MySQL, seleção de uma base teste e consulta todos os registros da tabela Clientes**

## 22 Exibição

Para que os registros da consulta sejam exibidos, deve-se usar a função mysql_fetch_assoc, que retorna uma matriz com a linha atual e move para a próxima. Para se imprimir todos os resultados de uma query, é necessária a construção de uma estrutura de repetição (while) até que a função mysql_fetch_assoc não retorne nenhum valor (vazio). Para melhorar a apresentação dos resultados, é possível usar tags HTML que incluam os dados dentro de tabelas, por exemplo.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
$link = mysql_connect("127.0.0.1", "root", "")
  or die("Não foi possível conectar");

mysql_select_db("teste")
  or die("Não foi possível selecionar o banco de dados");

$consulta = "SELECT * FROM Clientes";
$resultado = mysql_query($consulta)
  or die("Falha na execução da consulta");

$linha = mysql_fetch_assoc($resultado);

$NomeDaEmpresa = $linha["NomeDaEmpresa"];
$NomeDoContato = $linha["NomeDoContato"];

echo "<b>Nome da empresa:</b> $NomeDaEmpresa<br>";
echo "<b>Nome do contato:</b> $NomeDoContato";

?>

</body>
</html>
```

**Figura 50 – Código com impressão de um registro a partir de uma consulta**

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
$link = mysql_connect("127.0.0.1", "root", "")
  or die("Não foi possível conectar");

mysql_select_db("teste")
  or die("Não foi possível selecionar o banco de dados");
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
$consulta = "SELECT * FROM Clientes";
$resultado = mysql_query($consulta)
  or die("Falha na execução da consulta");

while ($linha = mysql_fetch_assoc($resultado))
{

  $NomeDaEmpresa = $linha["NomeDaEmpresa"];
  $NomeDoContato = $linha["NomeDoContato"];

  echo "<b>Nome da empresa:</b> $NomeDaEmpresa<br>";
  echo "<b>Nome do contato:</b> $NomeDoContato<p>";
}

?>

</body>
</html>
```

**Figura 51 – Código com impressão de todos registros a partir de uma consulta**

## 23 Consulta e Ordenação

Pode-se fazer consultas atendendo a certos critérios, que vão fazer com que a consulta seja refinada. No próximo exemplo, desejamos selecionar apenas os clientes de São Paulo. Nesse caso, a cláusula que deve ser alterada é a que faz a consulta SQL.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php

$link = mysql_connect("127.0.0.1", "root", "")
  or die("Não foi possível conectar");

mysql_select_db("teste")
  or die("Não foi possível selecionar o banco de dados");

$consulta = "SELECT NomeDaEmpresa, NomeDoContato
               FROM Clientes
               WHERE Cidade = 'São Paulo'";

$resultado = mysql_query($consulta)
  or die("Falha na execução da consulta");

while ($linha = mysql_fetch_assoc($resultado))
{

  $NomeDaEmpresa = $linha["NomeDaEmpresa"];
  $NomeDoContato = $linha["NomeDoContato"];

  echo "<b>Nome da empresa:</b> $NomeDaEmpresa<br>";
```

```
  echo "<b>Nome do contato:</b> $NomeDoContato<p>";
}

?>

</body>
</html>
```

**Figura 52 – Código com impressão de determinados registros que satisfazem uma condição (select ... from ... where ...)**

No caso de ordenação, a cláusula order by deve ser anexada à query SQL.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
$link = mysql_connect("127.0.0.1", "root", "")
  or die("Não foi possível conectar");

mysql_select_db("teste")
  or die("Não foi possível selecionar o banco de dados");

$consulta = "SELECT NomeDaEmpresa, NomeDoContato
               FROM Clientes
               WHERE Cidade = 'São Paulo'
            ORDER BY NomeDoContato";

$resultado = mysql_query($consulta)
  or die("Falha na execução da consulta");

while ($linha = mysql_fetch_assoc($resultado))
{

  $NomeDaEmpresa = $linha["NomeDaEmpresa"];
  $NomeDoContato = $linha["NomeDoContato"];

  echo "<b>Nome da empresa:</b> $NomeDaEmpresa<br>";
  echo "<b>Nome do contato:</b> $NomeDoContato<p>";
}

?>

</body>
</html>
```

**Figura 53 – Código com impressão de determinados registros que satisfazem uma condição, ordenados por um dos atributos (select ... from ... where ... order by ...)**

## 24 Inclusão e Atualização

Para se incluir dados em uma tabela MySQL, deve-se usar o comando INSERT. No exemplo a seguir, a inclusão de dados é estática. Para se criar um aplicativo que permita

inclusão, seria necessário adaptar o script para receber dados via formulário e incluí-los no banco de dados.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
$link = mysql_connect("127.0.0.1", "root", "")
  or die("Não foi possível conectar");

mysql_select_db("teste")
  or die("Não foi possível selecionar o banco de dados");

$CodigoDoCliente = "EELTD";
$NomeDaEmpresa = "Editora Europa";
$NomeDoContato = "Rodolfo Melo";
$Cidade = "São Paulo";

$consulta = "INSERT INTO Clientes
               (CódigoDoCliente, NomeDaEmpresa, NomeDoContato, Cidade)
             VALUES
               ('$CodigoDoCliente', '$NomeDaEmpresa', '$NomeDoContato',
'$Cidade')";
$resultado = mysql_query($consulta)
  or die("Falha na execução da consulta");

echo "Dados adicionados com sucesso";

?>

</body>
</html>
```

**Figura 54 – Código com inclusão de um registro em uma tabela do banco de dados**

O comando UPDATE altera um registro de uma tabela. No exemplo a seguir, o registro cujo código é “EELTD” passa a ter o nome “Robinson Melgar”.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
$link = mysql_connect("127.0.0.1", "root", "")
  or die("Não foi possível conectar");

mysql_select_db("teste")
  or die("Não foi possível selecionar o banco de dados");

$CodigoDoCliente = "EELTD";
$NomeDoContato = "Robinson Melgar";
```

```
$consulta = "UPDATE Clientes
                SET NomeDoContato = '$NomeDoContato'
              WHERE CódigoDoCliente = '$CodigoDoCliente'";

$resultado = mysql_query($consulta)
  or die("Falha na execução da consulta");

echo "Dados alterados com sucesso";

?>

</body>
</html>
```

**Figura 55 – Código com alteração de dados via comando update**

## 25 Exclusão

O comando SQL DELETE remove um registro de uma tabela. A cláusula WHERE delimita a condição para que a remoção seja executada.

```
<html>
<head>
<title>Página PHP</title>
</head>
<body>

<?php
$link = mysql_connect("127.0.0.1", "root", "")
  or die("Não foi possível conectar");

mysql_select_db("teste")
  or die("Não foi possível selecionar o banco de dados");

$CodigoDoCliente = "EELTD";

$consulta = "DELETE FROM Clientes
                   WHERE CódigoDoCliente = '$CodigoDoCliente'";

$resultado = mysql_query($consulta)
  or die("Falha na execução da consulta");

echo "Registro excluído com sucesso";

?>

</body>
</html>
```

**Figura 56 – Código com remoção de registros**

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

## Referências

BAKKEN, S. S. et al. **PHP Manual**. Disponível em: <http://br.php.net/get/php_manual_pt_BR.chm/from/this/mirror>. Acesso em: 21 out. 2003.

EDITORA Europa. Curso de PHP. **www.com.br**, São Paulo, n. 40, set. 2003. CD-ROM.

# COMO INICIAR UM LUCRATIVO NEGÓCIO ONLINE!





***Por Plauto L. Vesaro***



## *DIREITOS DE DISTRIBUIÇÃO E REVENDA DESTE EBOOK*

A PARTIR DE AGORA VOCÊ TEM PERMISSÃO PARA DISTRIBUIR LIVREMENTE E REVENDER ESTE EBOOK PELO PREÇO QUE DESEJAR E FICAR COM 100% DOS LUCROS SOBRE CADA VENDA! VOCÊ TAMBÉM PODE UTILIZÁ-LO COMO BÔNUS PARA VENDER UM OUTRO PRODUTO OU TAMBÉM PODE OFERECÊ-LO COMO BRINDE PARA QUEM ASSINAR SUA NEWSLETTER EXATAMENTE COMO EU FAÇO. APENAS LEMBRE-SE QUE VOCÊ NÃO TEM PERMISSÃO PARA ALTERAR O CONTEÚDO DO MESMO, QUER PARCIALMENTE, QUER INTEGRALMENTE.

# 1) A INTERNET: UM GRANDE NEGÓCIO!!

Você já parou para pensar no poder de comunicação que a internet tem? Com apenas um website uma pessoa pode atingir não somente **TODAS AS PESSOAS DO BRASIL COMO TAMBÉM AS DE TODO O MUNDO!** Isso é o que faz da internet uma grande oportunidade de negócio, uma mina de ouro para aqueles que sabem manuseá-la a seu favor. E é exatamente por isso que muitos internautas atualmente faturam milhares de reais por mês apenas negociando via online. Mas o que eles fazem? Como fazem? São estas perguntas que serão respondidas neste ebook.

A maioria das pessoas quando pensam em internet, pensam em redes sociais, bate papo, lazer, mas não pensam nela como uma fonte ou oportunidade para se ganhar dinheiro. **Quero abrir seus olhos nesse tocante e lhe mostrar um ramo de negócio online surpreendentemente lucrativo!** Então o primeiro passo que você deve entender é que a internet pode ser para você uma excelente fonte de renda extra ou até mesmo sua principal fonte de renda, desde que você siga as estratégias corretas de marketing, isto é, as que já foram testadas, experimentadas e aprovadas.

# 2) O QUE VENDER NA INTERNET?

Há pessoas que tremem de medo só de pensar em vender; porém na internet qualquer um pode vender sem ter que ser um "vendedor de verdade". Isto porque as habilidades que um vendedor tradicional precisa, como, conquistar, influenciar, persuadir, buscar contatos, etc, na internet são feitas automaticamente pelo website sem que haja nenhuma intervenção humana direta. Isto é realmente **FANTÁSTICO!** Uma máquina que fará todo o serviço na qual você terá apenas

que prepará-la, publicá-la e adptá-la para que ela Conquiste, Influencie e Venda para você!! Ora, quem não quer um negócio assim?

Mas nesse momento você pode estar pensando: Mas para vender na internet, eu vou precisar de um produto, vou precisar embalar, enviar via correio e controlar fretes. E eu lhe digo: **NÃO!!** No modelo de negócio que eu lhe sugirei você **NÃO PRECISARÁ** estocar nada, nem enviar nada pelos correios. Que tipo de negócio é esse?



## VOCÊ PRECISA TER UM INFOPRODUTO! O QUE É UM INFOPRODUTO?

Infoproduto é todo produto criado que pode ser usufruido e vendido (distribuído) via internet. Exemplos: um vídeo-curso ou uma vídeo-aula; um livro eletrônico ou um eBook; um programa de computador ou um software. Qualquer produto desenvolvido dentro desses ramos pode ser chamado de infoproduto e, por conseguinte, utilizado e vendido (distribuído) via internet.



Antes de qualquer coisa, então, você precisa ter um infoproduto para iniciar o seu negócio online. Veja bem: você está livre para comercializar qualquer coisa na internet, mas leve em conta que um infoproduto não precisa de estoques, não precisa de pagamentos de fretes, não precisa de embalagens, não precisa de despesas de envio. Vender infoproduto na internet é a forma mais sábia de iniciar e ganhar muito dinheiro. Você o reproduz facilmente e ilimitadas vezes e entrega ao seu cliente via e-mail! Neste e-mail estará contido um link para o download do produto. O cliente acessa o e-mail dele, abre sua mensagem e clica no link especificado para o download. **Pronto! A entrega do produto já foi realizada!**

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

Percebeu a diferença de um infoproduto para um produto físico? Este é um segredo que está deixando muitos internautas **RICOS DE VERDADE!** Eu conheço internautas que ganham à beira de quinhentos reais QUASE QUE TODO DIA! É claro que isso não é da noite pro dia. Por trás existe muito trabalho, muita tática e muita estratégia! É preciso saber a forma correta de se trabalhar!

Então se ter um infoproduto é um elemento essencial para você começar seu negócio online três opções estão disponíveis:



***Opção 1**: Você pode adquirir produtos com direitos de revenda!*

O que são produtos com direito de revenda? Basicamente são produtos que, quando vendidos, revertem ao comprador o direito de os revender e os distribuir ilimitadamente. Além disso, o comprador fica com 100% de lucro sobre cada venda concluída.



Esta é uma ótima maneira para começar seu negócio online, pois existem na internet diversos materiais e produtos de excelente qualidade que podem ser adquiridos por um bom preço e revendidos livremente.

É importante notar, porém, o seguinte: embora o comprador adquira este direito no ato da compra, ele não pode de maneira nenhuma alterar o conteúdo do produto, quer parcialmente, quer integralmente. Assim como qualquer outro produto, os infoprodutos também possuem direitos autorais e o descumprimento dessa norma acarreta sérias ações judiciais.

Como saber que uma obra possui o direito de revenda? Esta especificação é feita pelo próprio autor(a) da obra. Em eBooks essa declaração geralmente é feita no cabeçalho; outras vezes no encerramento.

Este relatório que você está lendo agora, possui direitos ilimitados de Revenda e também pode ser distribuído livremente, **desde que mantidas as fontes e o conteúdo não seja violado ou alterado nem parcialmente nem integralmente.**

Veja alguns produtos de qualidade com **DIREITOS DE REVENDA** que você pode adquirir e revender na internet (inclusive eu os vendo e ganho dinheiro com eles):

1) **TV Digital Na Internet:** Um software disponibiliza mais de 3000 canais na internet, além de milhares de downloads de músicas, vídeos, filmes e documentários. Este software é vendido por apenas R$ 9,90 e não possui mensalidades. Veja o link desta página: http://www.tvdigitalnainternet.net
Para adquirir o direito de revenda deste produto acessar: http://www.tvdigitalnainternet.net/revenda

O que você acha, não é um produto quente? As pessoas não procuram na internet exatamente filmes, videos, músicas e canais de TV?

2) **The Ultimate BackLink Builder**: Um programa capaz de criar centenas de backlinks para seu website em menos de 20 minutos! Todos os detalhes em http://www.rendacursos.com/backlinkbuilder

***Opção 2:*** *Vender infoprodutos de terceiros como afiliado!*

O que é um afiliado? Afiliado é quando você vende um produto terceirizado ganhando uma comissão sobre cada venda. Há muitos sites por aí que oferecem o sistema de afiliado. E há alguns que realmente valem a pena você se tornar afiliado. Tanto na primeira opção quanto na segunda você pode iniciar o seu negócio sem ter o trabalho de desenvolver seu próprio produto e mesmo assim ganhar um bom dinheiro.

No entanto, é importante tomar algumas precauções antes de iniciar um negócio no sistema de afiliados. É importante:

- avaliar a confiabilidade da companhia e verificar a honestidade e compromisso com a entrega dos produtos.

- avaliar se a companhia realmente tem sucesso na internet e

- verificar quanto de comissão você ganhará por venda.



Confiabilidade e honestidade atualmente são imprescindíveis, principalmente porque há muita enganação, roubo, nesse mundo dos negócios. Se você se tornar afiliado de uma empresa ou de um sistema de trabalho, precisa **TER CERTEZA** da presença desses dois predicados: confiabilidade e honestidade. Caso contrário, ao invés de ganhar dinheiro você terá má reputação e muita dor de cabeça.



Avaliar se a companhia tem realmente sucesso na internet também é de suma importância. De nada adiante você se tornar afiliado de uma empresa que não tem sucesso e não sabe vender na internet. Se você vender os mesmos produtos que uma companhia dessas também terá o mesmo destino: o fracasso!

Verifique, também, quanto de comissão você ganhará por cada venda. Os bons sistemas de negócios são aqueles que oferecem comissões acima de 50%. Portanto não trabalhe com companhias que oferecem porcentagem abaixo disso, a menos, é claro, que o produto a ser vendido seja **MUITO BOM COM ALTO VALOR PERCEBIDO**, e que sua comissão, embora abaixo de 50%, lhe forneça também um bom valor a receber. Exemplo: Um ótimo produto que é vendido por R$ 300,00. Sua comissão é de 40% sobre cada venda. Seu lucro então é de: 40% de 300 = R$

120,00! Nada mal! Então nesse caso é claro que vale a pena trabalhar com esta companhia. Avalie sempre sua comissão!

Além de avaliar essas três características mais importantes, você poderá avaliar também: se a companhia lhe dará suporte facilmente; se lhe dará um site pronto para divulgar; se lhe dará sua comissão na data determinada; se lhe dará apoio a problemas com clientes.

Quero recomendar um sistema de afiliados que eu mesmo participo e que:



- É plenamente confiável.

- É plenamente honesto.

- Tem um nível de ganhos absolutamente **INACREDITÁVEL,** conforme você mesmo pode verificar no site que indicarei abaixo.



- Disponibiliza a todos os afiliados um site pronto, isto é, um subdominio do site principal que a própria companhia utiliza. Você, portanto, terá a mesma ferramenta que a companhia utiliza!

- Disponibiliza a todos os afiliados um Escritório Virtual, onde estão contidas inúmeras informações a respeito do negócio, inclusive, um extenso manual com diversas modalidades de divulgação do seu site.

- Comissão de aproximadamente 53% sobre cada venda!

- Negócio com alto pontencial de crescimento!

- E o mais **INCRÍVEL**: dá a você a oportunidade de construir seu próprio sistema de afiliados, alavancando ainda mais seus

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

ganhos!

Acesse: www.milreais.ganhardinheiroagora.com e leia as informações completas!

Bom, embora você possa iniciar seu negócio online utilizando as opções 1 e 2 e ganhar mesmo muito dinheiro com elas, existem duas desvantagens nessas opções que são:

- Muita concorrência: imagine que você comprou um produto com direito de revenda. Conseguiu vendê-lo para 5 pessoas ficando com 100% dos lucros. Acontece que essas 5 pessoas que compraram o produto de você também tomaram posse do direito de revendê-lo. Vamos supor que cada uma delas também tenha vendido para 5 pessoas; 5 x 5 = 25 pessoas. Essas 25 pessoas também revenderam para mais 5 pessoas cada uma. Então: 25 x 5 = 125 pessoas. Idem para estas 125 pessoas; 125 x 5 = 625 novas pessoas que também podem revender o mesmo produto. Vamos parar por aqui. Só aqui nesse momento existem você + 5 + 25 +125 + 625 = 781 pessoas revendendo o mesmo produto!





Não demorará muito e este produto estará "saturado"(isto é polêmico pois a internet também cresce muito a cada dia) no mercado acarretando a segunda desvantagem:

- desvalorização rápida: como existem muitas concorrências os vendedores começarão a vender o produto por um preço cada vez menor, no intuito de vendê-lo mais e assim, o lucro com aquele produto também será enfraquecido. Portanto, a melhor maneira de ganhar dinheiro com as opções 1 e 2 é ser pioneiro do produto ou negócio ou fazer uso de um sistema que possa te fazer construir seu próprio sistema de afiliados, como referi acima no sistema de afiliados que participo.

***Opção 3:*** *Produzindo seus próprios produtos!*

Esta é, com certeza, a melhor forma de se ganhar dinheiro na internet! Quando você cria seu próprio produto você adquire exclusividade, e se o seu produto for bom, quente, acredite, você pode ganhar milhares de reais por mês com ele! Pergunte-se a si mesmo: O que você mais gosta de fazer? Você possui algum conhecimento técnico que poderia ser transformado num curso? Qual informação você vive correndo atrás? Existe algum modo de aproveitar estas informações para escrever um eBook? Qual o ramo do seu trabalho? Seria possível extrair daí informações para desenvolvê-las num vídeo curso ou num áudio curso?



Posteriormente a essas perguntas você precisará levar em conta outras: Existe mercado para o que estou produzindo? Meu produto tem apelação para o que as pessoas em geral "precisam" ou "necessitam"? Estas perguntas são muito importantes pois não adianta nada a pessoa produzir um excelente video curso sobre algum tema, se, na verdade, quase ninguém se interessar por ele. Seu produto precisa ter apelação, isto é, precisa ter conteúdo que interesse a maioria das pessoas.



Um milionário dizia que, para ganhar dinheiro ou mesmo ficar rico **"basta oferecer para as pessoas o que elas precisam ou necessitam, com um lucro em cima disso".**

Como escrever um eBook ou livro digital?

Antes de escrever é aconselhável fazer uma lista de todos os pontos que quer considerar em seu livro. Posteriormente, descreva numa sequência lógica cada um deles em seu relatório.

Escreva de uma forma clara, de modo que todos possam entender. Sua escrita deve ter começo, meio e fim. Sempre conclua seu raciocínio. Procure escrever frases e parágrafos curtos para não cansar a leitura. Utilize títulos e subtítulos para

facilitar a compreensão do leitor.

Releia tudo que escreveu corrigindo o texto e inutilizando palavras desnecessárias simplificando ao máximo sua leitura. Após havê-lo concluído, escreva uma introdução e, se necessário, o índice, tratando-se de um eBook dividido em títulos e subtítulos.

O título do livro digital deve ser o mais sensacional possível, pois ele é um dos arranques de venda. A propósito, talvez você não saiba disto, muitas Editoras utilizam esse truque; pegam livros antigos que não venderam quase nada, fazem um nova edição mudando somente o título, e o vendem novamente.

Muitas vezes o livro torna-se um "best seller" somente por causa da mudança do título. Compreende agora porque o título é de importância capital? O título não deve ser muito longo. Escreva num papel vários títulos e depois escolha o que achar melhor, mais chamativo e simplificado para o seu eBook.

O seu livro digital também deverá ter uma capa virtual bonita. Lembre-se o que diz a regra: "as pessoas compram o livro pela capa". Portanto este elemento tem importância de mesma proporção que o título, pois, na verdade, um complementa o outro.

Para você ter uma capa 3D vai precisar de duas coisas:

Primeira: um programa para elaborar imagens que vão estampar a parte frontal da capa e outra imagem para estampar a parte lateral. Existem na internet, programas gratuitos que são utilizados para a criação de imagens de capas virtuais. Um deles é o Kronen Design. Você pode fazer o download gratuito desse programa diretamente no link do fabricante: www.kronen-media.de/download/kronendesign. No meu blog

http://www.rendacursos.com pretendo fazer um post sobre como utilizar este software.

Segunda: Você vai precisar de um programa que passe essas imagens para o formato de um livro 3D (três dimensões). Para isto recomendo um bom programa: o Quick Cover 3D. É barato e muito bom!

Como proteger sua obra de cópias não autorizadas?

Para proteger seu eBook basta apresentar na página inicial, onde está escrito o ano da publicação, o título e o seu nome como autor do livro a seguinte palavra: copyright (vide como exemplo a página inicial deste mesmo eBook). É bom também acrescentar este copyright em todas as outras páginas do seu ebook.

Pode-se também registrar seu eBook na Biblioteca Nacional para adquirir o ISBN. O site é: www.bn.br

Em qual formato seu eBook deve ser publicado?

Os dois formatos mais comuns utilizados para se publicar um eBook são: PDF e Compilador HTML.

Ebooks no formato HTML são arquivos criados da mesma extensão das páginas de internet. Para visualizá-los é necessário ter instalado o Internet Explorer 7 ou versão acima. Para produzir um ebook neste formato pode-se utilizar o Easy Ebook Creator.

Ebooks no formato PDF são arquivos produzidos em programas de textos normais como o word por exemplo e, posteriormente, transformados para o formato PDF. Existem na internet diversas ferramentas gratuitas utilizadas para transformações de arquivos em PDF. Eu particularmente utilizo um bom programa editor em PDF chamado eWRITER pro. Esse programa tem diversos

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

recursos, além do opcional de fornecer uma senha de acesso para o seu ebook, a fim de protegê-lo. Mais detalhes sobre esse produto em http://www.rendacursos.com/ebookprofissional

## 3) COMO VENDER NA INTERNET?

Antes de qualquer coisa é necessário adquirir um dominio, que é o endereço do site que será publicado na internet. Logo em seguida será necessário adquirir uma hospedagem para o seu dominio, isto é, um provedor certificado que publicará o seu site na internet.

Hoje em dia o próprio provedor de hospedagem já adquire o domínio e o disponibiliza na internet para o seu cliente. Portanto é um trabalho muito fácil. Um dominio profissional custa uns R$ 30,00 por ano e uma hospedagem profissional por volta de R$ 20,00 mensais.

Se você realmente pensa sério em montar seu negócio online, deve fazer esse investimento. Qual outro negócio próprio lhe disponibiliza iniciar com tão pouco investimento? Agora se você não tem nenhuma possibilidade de investir esses valores, procure por sites gratuitos (o que eu recomendo é: **www.freewebhostingarea.com**) ou procure montar blogs gratuitos promovendo seu produto.

Com relação a obtenção de dominio e hospedagem profissionais só posso aconselhar o serviço que eu mesmo utilizo e que é muito bom: www.hostgator.com.br

Há porém outros de grande qualidade como o indicado acima. Você pode fazer uma pesquisa na internet e verificar outros provedores.

**Uma Boa Dica:** Construa um **Mini Site** para o seu produto! Mini Sites são reconhecidamente sites que contém, de um modo geral, uma página só. Nesta página será descrita todos os benefícios do seu produto e no final o botão comprar para o cliente. Mas porque os mini sites são febre na internet e porque vendem como loucos? O que há por trás dos mini sites? As respostas para essas perguntas podem ser encontradas no ebook "Como Construir Mini Sites Arrasadores Que Vendem Como Loucos". Este ebook é distribuido gratuitamente para os assinantes da minha Newsletter (rendacursos.com).



Para editar o seu Mini Site será necessário possuir um software leitor e editor de arquivo html. Existem softwares que podem ser baixados gratuitamente da internet, como o Kompozer, NVU, etc. Basta acessar o Google e digitar "download Kompozer" ou "download NVU" que o link já aparecerá.



Para editar o seu Mini Site, além do software indicado acima será necessário possuir um que já venha pré configurado com cabeçalho, corpo e rodapé personalizáveis por meio do próprio editor de arquivo html. Para isto cai bem o produto **Mini Site Starter Kit** que eu comercializo na página: http://ww.rendacursos.com/kitminisite

Outra opção seria adquirir um template gratuito porém profissional no site: **http://www.minisitegallery.com**. Entretanto neste caso será necessário possuir também o programa Photoshop para fazer a edição das imagens, do cabeçalho e do rodapé.

Como acrescentar ao seu Mini Site diversas formas de pagamento?

Seu site poderá aceitar pagamentos por transferência bancária online, boleto bancário, cartão de crédito, dividindo a compra em

várias vezes, etc. Tudo isso feito num ambiente seguro, chamado Pagseguro - uma ferramente da empresa UOL. Para poder utilizar estes serviços do Pagseguro, acesse: http://www.pagseguro.uol.com.br e crie uma conta vendedor. Após ter criado esta conta você estará apto a inserir em seu Mini Site um botão do Pagseguro que permitirá ao seu comprador escolher dentre várias formas de pagamento. O cadastro no Pagseguro é gratuito e você pagará apenas uma pequena taxa de cada venda que fizer. Essa taxa é descontada automaticamente, assim que o cliente faz o pagamento.

Para aprender a inserir um botão de pagamento no seu Mini Site acesse o meu blog (rendacursos.com) e assita ao vídeo "Como Personalizar o Botão de Pagamento de um Mini Site Pronto".

Como enviar o seu Produto Digital por email ao seu cliente?

Antes de qualquer coisa é necessário zipar o seu produto, seja ele um ebook, seja ele um vídeo, seja ele um software. Se o seu computador já extrai arquivos zip normalmente, então basta clicar com o botão direito do mouse em cima do produto a ser zipado e clicar em comprimir.

Feito isto, transfira o seu produto para uma pasta especifica com um nome relacionado ao seu produto. Por exemplo: você criou o produto "Como Ganhar Dinheiro Com Seu Computador", um ebook em PDF. Depois de o ter zipado, o renomeou para "DinheiroComComputador" e o enviou para uma pasta chamada Download-Produto23. Em seguida, então, será necessário enviar esta pasta com o arquivo zipado dentro para o servidor de hospedagem do seu Site (se tiver dúvida com relação a isso consulte os técnicos do provedor de hospedagem do seu site). A partir daí você monta o link para download que será enviado para o cliente. No exemplo citado acima ficaria assim: http://www.nomedoseusite.com/Download-Produto23/DinheiroCo

mComputador.zip

Pronto. Assim que seu cliente clicar no link acima o download começará automaticamente. É importante não esquecer de acrescentar .zip no link para que tudo funcione corretamente.

**Uma dica Valiosíssima:** Procure identificar o nome do seu mini site com o nome do seu produto. Por exemplo, o nome do seu ebook é: Como Montar um Site. Então um endereço perfeito para o seu Mini Site seria: www.comomontarumsite.com. Fazendo assim você aumenta a visibilidade do seu site nos mecanismos de buscas, que utilizam palavras chaves, que são palavras muito buscadas na internet e nos sites de busca.

**Outra dica Valiosíssima:** Como enquadrar o título do seu produto, o endereço do seu Mini Site com palavras chaves procuradas? Fazendo essa pergunta com outras palavras: Como saber quais são as palavras mais procuradas nos mecanismos de buscas por exemplo, como o Google, para poder inseri-las no Título do seu produto e no endereço do seu Mini Site?

A resposta está neste endereço: **www.google.com.br/adwords**. Após acessar esta página procure por "Obtenha sugestões de palavras chaves". Clique e outra página lhe abrirá. Então basta inserir uma palavra chave relacionada ao nicho do seu produto e clicar em “pesquisar”.

Este é um **SEGREDO** utilizado pelos **MARKETEIROS TOP** da internet! Colocando seu negócio em boas palavras chaves, que são aquelas que não possuem muita concorrência e que são ao mesmo tempo bem procuradas e solicitadas, você atrairá centenas e até milhares de pessoas ao seu Mini Site Gratuitamente.

Se o seu produto, por exemplo, tem a ver com emagrecimento,

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

escreva emagrecimento no campo pedido e depois clique em "pesquisar". Você verá as palavras chaves mais buscadas no google com relação a este tema. Na barra lateral esquerda dessa mesma página tem um link "Estimador de tráfego". Pesquise isso lá também.

Após ter associado o título do seu produto e do seu site a boas palavras chaves, então a primeira coisa a ser feita é registrar seu Website nos Sites de Busca, como Google, Yahoo, etc. Você pode usar o serviço gratuito do DataHosting (**www.datahosting.com.br/divulgador**) para registrar seu Mini Site simultaneamente em dezenas de Sites de Busca. Assim você economiza tempo.



Posteriormente a este passo principal e primordial você poderá divulgar seu produto de diversas formar como:

- Criando artigos e divulgando-os em diretórios de artigos;
- Gerando links para seu negócio nos agregadores de links;
- Fazendo anúncios classificados online gratuitos ou pagos;
- Fazendo divulgação nas redes sociais, twitter, orkut, facebook, etc.
- Fazendo propaganda paga em links patrocinados do Google, Yahoo, Uol, etc.
- Fazendo vídeo marketing (principalmente no youtube).



Com relação aos diretórios de links recomendo os seguintes:
**- www.webartigos.com**
**- www.artigonal.com**
**- www.superartigos.com**
**- www.artigos.com**

Com relação a agregadores de links recomendo os seguintes:
**- http://dihitt.com.br**
**- http://linkto.com.br**

**- http://linkk.com.br**
**- http://domelhor.net**

**Para encerrar outra Dica Valiosíssima:** Você precisa fazer uso do email marketing através de uma Lista de Captação de emails. Instale um formulário em seu site para captar emails de pessoas interessadas e programe mensagens falando de sua oportunidade ou produto utilizando um autoresponder. As mensagens são programadas e enviadas automaticamente para seus prospectos. É como uma Secretária Virtual. Isto é FUNDAMENTAL para o seu negócio evoluir porque quase ninguém compra um produto no primeiro contato com ele. 80% das vendas pela internet são feitas num segundo contato do interessado com o produto. Procure acessar o site: http://www.e-goi.com.br e procure sobre email marketing. Eles oferecem um plano gratuito para até 500 prospectos e fazem um bom trabalho.



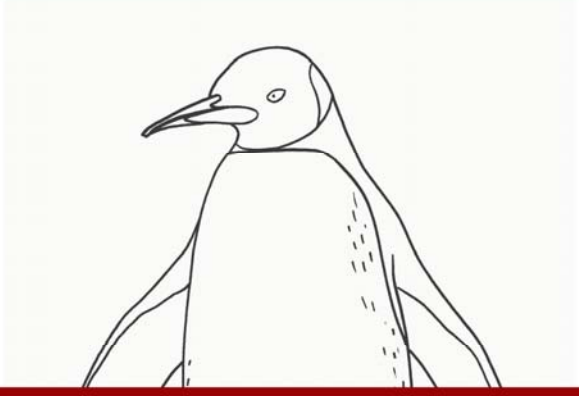


Se quiser uma ferramenta de autoresponder também profissional e com preço barato para **ilimitados** contatos e ainda por cima, que seja de lingua portuguesa acesse: http://www.ptsender.com/plauto

Se o seu nicho é renda extra pela internet, faça uso deste ebook que está lendo e ofereça-o como Brinde Grátis para todos aqueles que assinarem sua Newsletter. Assim você vai captando futuros clientes para o seu negócio.

Lembre-se: A forma mais fácil de iniciar um negócio lucrativo na internet é adquirindo produtos com direitos de revenda ou promovendo um link como afiliado. No decorrer do tempo crie o seu próprio produto e eleve o seu nível!
**LHE DESEJO MUITO SUCESSO!!**

Plauto Vesaro - www.rendacursos.com



# LINUX 101 HACKS

Practical Examples to Build a Strong Foundation in Linux

*Ramesh Natarajan*
*www.thegeekstuff.com*

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# Table of Contents

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# Introduction

*"There are only 10 types of people in the world — those who understand binary, those who don't, and those who understand gray code"*

*— Geek*

There are total of 101 hacks in this book that will help you build a strong foundation in Linux. All the hacks in this book are explained with appropriate Linux command examples that are easy to follow.

This book contains 12 chapters. Hacks mentioned in 6 chapters are based on the articles that I've already posted on my blog. Hacks mentioned in rest of the 6 chapters are brand new.

I'm Ramesh Natarajan, author of The Geek Stuff blog and this eBook.

I have done intensive programming on several languages and C is my favorite. I have done lot of work on the infrastructure side including Linux system administration, DBA, Networking, Hardware and Storage (EMC).

I have also developed Password Dragon — free, easy and secure password manager that runs on Windows, Linux and Mac.

If you have any feedback about this eBook, please use this contact form to get in touch with me.

# Foreword

Another collection of hacks? Yes! If you have just completed your first admin course or looking for better ways to get the job done the "Linux 101 Hacks" eBook is a good point to start. These useful tips are concise, well written and easy to read.

Well done - I will recommend this eBook to my students.

**--Prof. Dr. Fritz Mehner, FH Südwestfalen, Germany**

(Author of several Vim plugins, including bash-support vim plugin)

# Version

| Version | Date | Revisions |
|---|---|---|
| 1.0 | 12-Feb-2009 | First Edition |
| | | |

Download the latest version of the book here.

# Chapter 1: Powerful CD Command Hacks

cd is one of the most frequently used commands during a UNIX session. The 6 cd command hacks mentioned in this chapter will boost your productivity instantly and make it easier to navigate the directory structure from command line.

## Hack 1. Use CDPATH to define the base directory for cd command

If you are frequently performing cd to subdirectories of a specific parent directory, you can set the CDPATH to the parent directory and perform cd to the subdirectories without giving the parent directory path as explained below.

```
[ramesh@dev-db ~]# pwd
/home/ramesh

[ramesh@dev-db ~]# cd mail
-bash: cd: mail: No such file or directory

[Note: This is looking for mail directory under current
directory]

[ramesh@dev-db ~]# export CDPATH=/etc
[ramesh@dev-db ~]# cd mail
/etc/mail

[Note: This is looking for mail under /etc and not
under current directory]

[ramesh@dev-db /etc/mail]# pwd
/etc/mail
```

To make this change permanent, add export CDPATH=/etc to your ~/.bash_profile

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

Similar to the PATH variable, you can add more than one directory entry in the CDPATH variable, separating them with : , as shown below.

```
export CDPATH=.:~:/etc:/var
```

This hack can be very helpful under the following situations:

- Oracle DBAs frequently working under $ORACLE_HOME, can set the CDPATH variable to the oracle home
- Unix sysadmins frequently working under /etc, can set the CDPATH variable to /etc
- Developers frequently working under project directory /home/projects, can set the CDPATH variable to /home/projects
- End-users frequently accessing the subdirectories under their home directory, can set the CDPATH variable to ~ (home directory)

## Hack 2. Use cd alias to navigate up the directory effectively

When you are navigating up a very long directory structure, you may be using cd ..\..\ with multiple ..\'s depending on how many directories you want to go up as shown below.

```
# mkdir -p
/tmp/very/long/directory/structure/that/is/too/deep

# cd /tmp/very/long/directory/structure/that/is/too/deep

# pwd
/tmp/very/long/directory/structure/that/is/too/deep

# cd ../../../../
```

```
# pwd
/tmp/very/long/directory/structure
```

Instead of executing cd ../../../.. to navigate four levels up, use one of the following three alias methods:

### Method 1: Navigate up the directory using "..n"

In the example below, ..4 is used to go up 4 directory level, ..3 to go up 3 directory level, ..2 to go up 2 directory level. Add the following alias to your ~/.bash_profile and re-login.

```
alias ..="cd .."
alias ..2="cd ../.."
alias ..3="cd ../../.."
alias ..4="cd ../../../.."
alias ..5="cd ../../../../.."
```

```
# cd
/tmp/very/long/directory/structure/that/is/too/deep

# ..4
[Note: use ..4 to go up 4 directory level]

# pwd
/tmp/very/long/directory/structure/
```

### Method 2: Navigate up the directory using only dots

In the example below, ….. (five dots) is used to go up 4 directory level. Typing 5 dots to go up 4 directory structure is really easy to remember, as when you type the first two dots, you are thinking "going up one directory", after that every additional dot, is to go one level up. So, use …. (four dots) to go up 3 directory level and .. (two dots) to go up 1 directory level. Add the following alias to your ~/.bash_profile and re-login for the ….. (five dots) to work properly.

```
alias ..="cd .."
```

```
alias ...="cd ../.."
alias ....="cd ../../.."
alias .....="cd ../../../.."
alias ......="cd ../../../../.."
```

```
# cd /tmp/very/long/directory/structure/that/is/too/deep

# .....
[Note: use ..... (five dots) to go up 4 directory level]

# pwd
/tmp/very/long/directory/structure/
```

### Method 3: Navigate up the directory using cd followed by consecutive dots

In the example below, cd….. (cd followed by five dots) is used to go up 4 directory level. Making it 5 dots to go up 4 directory structure is really easy to remember, as when you type the first two dots, you are thinking "going up one directory", after that every additional dot, is to go one level up. So, use cd…. (cd followed by four dots) to go up 3 directory level and cd… (cd followed by three dots) to go up 2 directory level. Add the following alias to your ~/.bash_profile and re-login for the above cd….. (five dots) to work properly.

```
alias cd..="cd .."
alias cd...="cd ../.."
alias cd....="cd ../../.."
alias cd.....="cd ../../../.."
alias cd......="cd ../../../../.."
```

```
# cd /tmp/very/long/directory/structure/that/is/too/deep

# cd.....
[Note: use cd..... to go up 4 directory level]

# pwd
/tmp/very/long/directory/structure
```

### Method 5: Navigate up the directory using cd followed by number

In the example below, cd4 (cd followed by number 4) is used to go up 4 directory level.

```
alias cd1="cd .."
alias cd2="cd ../.."
alias cd3="cd ../../.."
alias cd4="cd ../../../.."
alias cd5="cd ../../../../.."
```

## Hack 3. Perform mkdir and cd using a single command

Sometimes when you create a new directory, you may cd to the new directory immediately to perform some work as shown below.

```
# mkdir -p /tmp/subdir1/subdir2/subdir3

# cd /tmp/subdir1/subdir2/subdir3

# pwd
/tmp/subdir1/subdir2/subdir3
```

Wouldn't it be nice to combine both mkdir and cd in a single command? Add the following to the .bash_profile and re-login.

```
$ vi .bash_profile

function mkdircd () { mkdir -p "$@" && eval cd
"\"\$$#\""; }
```

Now, perform both mkdir and cd at the same time using a single command as shown below:

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
# mkdircd /tmp/subdir1/subdir2/subdir3

[Note: This creates the directory and cd to it
automatically]

# pwd
/tmp/subdir1/subdir2/subdir3
```

## Hack 4. Use “cd -” to toggle between the last two directories

You can toggle between the last two current directories using cd - as shown below.

```
# cd /tmp/very/long/directory/structure/that/is/too/deep

# cd /tmp/subdir1/subdir2/subdir3

# cd -

# pwd
/tmp/very/long/directory/structure/that/is/too/deep


# cd -

# pwd
/tmp/subdir1/subdir2/subdir3

# cd -

# pwd
/tmp/very/long/directory/structure/that/is/too/deep
```

## Hack 5. Use dirs, pushd and popd to manipulate directory stack

You can use directory stack to push directories into it and later pop directory from the stack. Following three commands are used in this example.

- dirs: Display the directory stack
- pushd: Push directory into the stack
- popd: Pop directory from the stack and cd to it

Dirs will always print the current directory followed by the content of the stack. Even when the directory stack is empty, dirs command will still print only the current directory as shown below.

```
# popd
-bash: popd: directory stack empty

# dirs
~

# pwd
/home/ramesh
```

How to use pushd and popd? Let us first create some temporary directories and push them to the directory stack as shown below.

```
# mkdir /tmp/dir1
# mkdir /tmp/dir2
# mkdir /tmp/dir3
# mkdir /tmp/dir4

# cd /tmp/dir1
# pushd .

# cd /tmp/dir2
# pushd .

# cd /tmp/dir3
# pushd .

# cd /tmp/dir4
# pushd .
```

```
# dirs
/tmp/dir4 /tmp/dir4 /tmp/dir3 /tmp/dir2 /tmp/dir1

[Note: The first directory (/tmp/dir4) of the dir
command output is always the current directory and not
the content from the stack.]
```

At this stage, the directory stack contains the following directories:

```
/tmp/dir4
/tmp/dir3
/tmp/dir2
/tmp/dir1
```

The last directory that was pushed to the stack will be at the top. When you perform popd, it will cd to the top directory entry in the stack and remove it from the stack. As shown above, the last directory that was pushed into the stack is /tmp/dir4. So, when we do a popd, it will cd to the /tmp/dir4 and remove it from the directory stack as shown below.

```
# popd
# pwd
/tmp/dir4

[Note: After the above popd, directory Stack Contains:
/tmp/dir3
/tmp/dir2
/tmp/dir1]

# popd
# pwd
/tmp/dir3

[Note: After the above popd, directory Stack Contains:
/tmp/dir2
/tmp/dir1]

# popd
```

```
# pwd
/tmp/dir2

[Note: After the above popd, directory Stack Contains:
/tmp/dir1]

# popd
# pwd
/tmp/dir1

[Note: After the above popd, directory Stack is empty!]

# popd
-bash: popd: directory stack empty
```

## Hack 6. Use “shopt -s cdspell” to automatically correct mistyped directory names on cd

Use shopt -s cdspell to correct the typos in the cd command automatically as shown below. If you are not good at typing and make lot of mistakes, this will be very helpful.

```
# cd /etc/mall
-bash: cd: /etc/mall: No such file or directory

# shopt -s cdspell

# cd /etc/mall

# pwd
/etc/mail

[Note: By mistake, when I typed mall instead of mail,
          cd corrected it automatically]
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# Chapter 2: Date Manipulation

## Hack 7. Set System Date and Time

To change the system date use:

```
# date {mmddhhmiyyyy.ss}
```

- mm - Month
- dd - Date
- hh - 24 hour format
- mi - Minutes
- yyyy - Year
- ss - seconds

For example, to set system date to Jan 31st 2008, 10:19 p.m, 53 seconds

```
# date 013122192009.53
```

You can also change system date using set argument as shown below.

```
# date 013122192009.53

# date +%Y%m%d -s "20090131"

# date -s "01/31/2009 22:19:53"

# date -s "31 JAN 2009 22:19:53"

# date set="31 JAN 2009 22:19:53"
```

To set the time only:

```
# date +%T -s "22:19:53"
```

```
# date +%T%p -s "10:19:53PM"
```

## Hack 8. Set Hardware Date and Time

Before setting the hardware date and time, make sure the OS date and time is set appropriately as shown in the hack#7.

Set the hardware date and time based on the system date as shown below:

```
# hwclock –systohc

# hwclock --systohc –utc
```

Use hwclock without any parameter, to view the current hardware date and time:

```
# hwclock
```

Check the clock file to verify whether the system is set for UTC:

```
# cat /etc/sysconfig/clock

ZONE="America/Los_Angeles"
UTC=false
ARC=false
```

## Hack 9. Display Current Date and Time in a Specific Format

Following are different ways of displaying the current date and time in various formats:

```
$ date
Thu Jan  1 08:19:23 PST 2009

$ date --date="now"
Thu Jan  1 08:20:05 PST 2009

$ date --date="today"
Thu Jan  1 08:20:12 PST 2009

$ date --date='1970-01-01 00:00:01 UTC +5 hours' +%s
18001

$ date '+Current Date: %m/%d/%y%nCurrent Time:%H:%M:%S'
Current Date: 01/01/09
Current Time:08:21:41

$ date +"%d-%m-%Y"
01-01-2009

$ date +"%d/%m/%Y"
01/01/2009

$ date +"%A,%B %d %Y"
Thursday,January 01 2009
```

Following are the different format options you can pass to the date command:

- %D date (mm/dd/yy)
- %d day of month (01..31)
- %m month (01..12)
- %y last two digits of year (00..99)
- %a locale's abbreviated weekday name (Sun..Sat)
- %A locale's full weekday name, variable length (Sunday..Saturday)
- %b locale's abbreviated month name (Jan..Dec)

- %B locale's full month name, variable length (January..December)
- %H hour (00..23)
- %I hour (01..12)
- %Y year (1970...)

## Hack 10. Display Past Date and Time

Following are various ways to display a past date and time:

```
$ date --date='3 seconds ago'
Thu Jan  1 08:27:00 PST 2009

$ date --date="1 day ago"
Wed Dec 31 08:27:13 PST 2008

$ date --date="1 days ago"
Wed Dec 31 08:27:18 PST 2008

$ date --date="1 month ago"
Mon Dec  1 08:27:23 PST 2008

$ date --date="1 year ago"
Tue Jan  1 08:27:28 PST 2008

$ date --date="yesterday"
Wed Dec 31 08:27:34 PST 2008

$ date --date="10 months 2 day ago"
Thu Feb 28 08:27:41 PST 2008
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

## Hack 11. Display Future Date and Time

Following examples shows how to display a future date and time.

```
$ date
Thu Jan  1 08:30:07 PST 2009

$ date --date='3 seconds'
Thu Jan  1 08:30:12 PST 2009

$ date --date='4 hours'
Thu Jan  1 12:30:17 PST 2009

$ date --date='tomorrow'
Fri Jan  2 08:30:25 PST 2009

$ date --date="1 day"
Fri Jan  2 08:30:31 PST 2009

$ date --date="1 days"
Fri Jan  2 08:30:38 PST 2009

$ date --date="2 days"
Sat Jan  3 08:30:43 PST 2009

$ date --date='1 month'
Sun Feb  1 08:30:48 PST 2009

$ date --date='1 week'
Thu Jan  8 08:30:53 PST 2009

$ date --date="2 months"
Sun Mar  1 08:30:58 PST 2009

$ date --date="2 years"
Sat Jan  1 08:31:03 PST 2011

$ date --date="next day"
Fri Jan  2 08:31:10 PST 2009
```

```
$ date --date="-1 days ago"
Fri Jan  2 08:31:15 PST 2009

$ date --date="this Wednesday"
Wed Jan  7 00:00:00 PST 2009
```

# Chapter 3: SSH Client Commands

## Hack 12. Identify SSH Client Version

Sometimes it may be necessary to identify the SSH client that you are currently running and it's corresponding version number. Use ssh -V to identify the version number. Please note that Linux comes with OpenSSH.

The following example indicates that this particular system is using OpenSSH:

```
$ ssh -V
OpenSSH_3.9p1, OpenSSL 0.9.7a Feb 19 2003
```

The following example indicates that this particular system is using SSH2:

```
$ ssh -V
ssh: SSH Secure Shell 3.2.9.1 (non-commercial version)
on i686-pc-linux-gnu
```

## Hack 13. Login to Remote Host using SSH

The First time when you login to a remotehost from a localhost, it will display the host key not found message and you can give "yes" to continue. The host key of the remote host will be added under .ssh2/hostkeys directory of your home directory, as shown below.

```
localhost$ ssh -l jsmith remotehost.example.com

Host key not found from database.
Key fingerprint:
xabie-dezbc-manud-bartd-satsy-limit-nexiu-jambl-title-jarde-
tuxum
You can get a public key's fingerprint by running
% ssh-keygen -F publickey.pub
```

```
on the keyfile.
Are you sure you want to continue connecting (yes/no)? Yes

Host key saved to
/home/jsmith/.ssh2/hostkeys/key_22_remotehost.example.com.pub
host key for remotehost.example.com, accepted by jsmith Mon
May 26 2008 16:06:50 -0700
jsmith@remotehost.example.com password:

remotehost.example.com$
```

The Second time when you login to the remote host from the localhost, it will prompt only for the password as the remote host key is already added to the known hosts list of the ssh client.

```
localhost$ ssh -l jsmith remotehost.example.com
jsmith@remotehost.example.com password:

remotehost.example.com$
```

For some reason, if the host key of the remote host is changed after you logged in for the first time, you may get a warning message as shown below. This could be because of various reasons such as:

- Sysadmin upgraded/reinstalled the SSH server on the remote host
- Someone is doing malicious activity etc.,

The best possible action to take before saying “yes” to the message below, is to call your sysadmin and identify why you got the host key changed message and verify whether it is the correct host key or not.

```
localhost$ ssh -l jsmith remotehost.example.com

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@       WARNING: HOST IDENTIFICATION HAS CHANGED!         @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
middle attack)!
It is also possible that the host key has just been changed.
Please contact your system administrator.
Add correct host key to
"/home/jsmith/.ssh2/hostkeys/key_22_remotehost.example.com.pu
b"
to get rid of this message.
Received server key's fingerprint:
xabie-dezbc-manud-bartd-satsy-limit-nexiu-jambl-title-arde-
tuxum
You can get a public key's fingerprint by running
% ssh-keygen -F publickey.pub
on the keyfile.
Agent forwarding is disabled to avoid attacks by corrupted
servers.
Are you sure you want to continue connecting (yes/no)? yes

Do you want to change the host key on disk (yes/no)? yes

Agent forwarding re-enabled.
Host key saved to
/home/jsmith/.ssh2/hostkeys/key_22_remotehost.example.com.pub
host key for remotehost.example.com, accepted by jsmith Mon
May 26 2008 16:17:31 -0700

jsmith @remotehost.example.com's password:

remotehost$
```

## Hack 14. Debug SSH Client Session

Sometimes it is necessary to view debug messages to troubleshoot any SSH connection issues. pass -v (lowercase v) option to the ssh as shown below to view the ssh debug messages.

### Example without SSH client debug message:

```
localhost$ ssh -l jsmith remotehost.example.com

warning: Connecting to remotehost.example.com failed:
No address associated to the name
```

**Example with SSH client debug message:**

```
locaclhost$ ssh -v -l jsmith remotehost.example.com

debug:
SshConfig/sshconfig.c:2838/ssh2_parse_config_ext:
Metaconfig parsing stopped at line 3.

debug:
SshConfig/sshconfig.c:637/ssh_config_set_param_verbose:
Setting variable 'VerboseMode' to 'FALSE'.

debug:
SshConfig/sshconfig.c:3130/ssh_config_read_file_ext:
Read 17 params from config file.

debug: Ssh2/ssh2.c:1707/main: User config file not
found, using defaults. (Looked for
'/home/jsmith/.ssh2/ssh2_config')

debug: Connecting to remotehost.example.com, port 22…
(SOCKS not used)
warning: Connecting to remotehost.example.com failed:
No address associated to the name
```

## Hack 15. Toggle SSH Session using SSH Escape Character

When you've logged on to the remotehost using ssh from the localhost, you may want to come back to the localhost to perform some activity and go back to remote host again. In this case, you don't need to disconnect the ssh session to the remote host. Instead, follow the steps below.

1. Login to remotehost from localhost:

```
localhost$ ssh -l jsmith remotehost
```

2. Now you are connected to the remotehost:

```
remotehost$
```

3. To come back to the localhost temporarily, type the escape character ~ and Control-Z.

When you type ~ you will not see that immediately on the screen until you press <Control-Z> and press enter. So, on the remotehost in a new line enter the following key strokes for the below to work: ~<Control-Z>

```
remotehost$ ~^Z
[1]+  Stopped  ssh -l jsmith remotehost

localhost$
```

4. Now you are back to the localhost and the ssh remotehost client session runs as a typical UNIX background job, which you can check as shown below:

```
localhost$ jobs
[1]+  Stopped  ssh -l jsmith remotehost
```

5. You can go back to the remote host ssh without entering the password again by bringing the background ssh remotehost session job to foreground on the localhost.

```
localhost$ fg %1
ssh -l jsmith remotehost

remotehost$
```

## Hack 16. SSH Session Statistics using SSH Escape Character

To get some useful statistics about the current ssh session, do the following. This works only on SSH2 client.

1. Login to remotehost from localhost.

```
localhost$ ssh -l jsmith remotehost
```

2. On the remotehost, type ssh escape character ~ followed by s as shown below. This will display lot of useful statistics about the current SSH connection.

```
remotehost$  [Note: The ~s is not visible on the
command line when you type.]

        remote host: remotehost
        local host: localhost
        remote version: SSH-1.99-OpenSSH_3.9p1
        local version:  SSH-2.0-3.2.9.1 SSH Secure
Shell (non-commercial)
        compressed bytes in: 1506
        uncompressed bytes in: 1622
        compressed bytes out: 4997
        uncompressed bytes out: 5118
        packets in: 15
        packets out: 24
        rekeys: 0
        Algorithms:
        Chosen key exchange algorithm: diffie-hellman-
group1-sha1
        Chosen host key algorithm: ssh-dss
        Common host key algorithms: ssh-dss,ssh-rsa
        Algorithms client to server:
        Cipher: aes128-cbc
        MAC: hmac-sha1
        Compression: zlib
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
        Algorithms server to client:
        Cipher: aes128-cbc
        MAC: hmac-sha1
        Compression: zlib

localhost$
```

### Additional SSH Info

On a side note, to setup SSH key based authentication, refer openSSH and SSH2 tutorials.

# Chapter 4: Essential Linux Commands

## Hack 17. Grep Command

grep command is used to search files for a specific text. This is incredibly powerful command with lot of options.

```
Syntax: grep [options] pattern [files]
```

### How can I find all lines matching a specific keyword on a file?

In this example, grep looks for the text John inside /etc/passwd file and displays all the matching lines.

```
# grep John /etc/passwd

jsmith:x:1082:1082:John Smith:/home/jsmith:/bin/bash
jdoe:x:1083:1083:John Doe:/home/jdoe:/bin/bash
```

Option -v, will display all the lines except the match. In the example below, it displays all the records from /etc/password that doesn't match John.

**Note:** There are several lines in the /etc/password that doesn't contain the word John. Only the first line of the output is shown below.

```
# grep -v John /etc/passwd

jbourne:x:1084:1084:Jason Bourne:/home/jbourne:/bin/bash
```

### How many lines matched the text pattern in a particular file?

In the example below, it displays the total number of lines that contains the text John in /etc/passwd file.

```
# grep -c John /etc/passwd
2
```

You can also get the total number of lines that did not match the specific pattern by passing option -cv.

```
# grep -cv John /etc/passwd
39
```

### How to search a text by ignoring the case?

Pass the option -i (ignore case), which will ignore the case while searching.

```
# grep -i john /etc/passwd

jsmith:x:1082:1082:John Smith:/home/jsmith:/bin/bash
jdoe:x:1083:1083:John Doe:/home/jdoe:/bin/bash
```

### How do I search all subdirectories for a text matching a specific pattern?

Use option -r (recursive) for this purpose. In the example below, it will search for the text "John" by ignoring the case inside all the subdirectories under /home/users.

This will display the output in the format of "filename: line that matching the pattern". You can also pass the option -l, which will display only the name of the file that matches the pattern.

```
# grep -ri john /home/users

/home/users/subdir1/letter.txt:John, Thanks for your
contribution.
/home/users/name_list.txt:John Smith
```

```
/home/users/name_list.txt:John Doe

# grep -ril john /root

/home/users/subdir1/letter.txt
/home/users/name_list.txt
```

## Hack 18. Find Command

find is frequently used command to find files in the UNIX filesystem based on numerous conditions. Let us review some practice examples of find command.

```
Syntax: find [pathnames] [conditions]
```

### How to find files containing a specific word in its name?

The following command looks for all the files under /etc directory with mail in the filename.

```
# find /etc -name "*mail*"
```

### How to find all the files greater than certain size?

The following command will list all the files in the system greater than 100MB.

```
# find / -type f -size +100M
```

### How to find files that are not modified in the last x number of days?

The following command will list all the files that were modified more than 60 days ago under the current directory.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
# find . -mtime +60
```

### How to find files that are modified in the last x number of days?

The following command will list all the files that were modified in the last two days under the current directory.

```
# find . –mtime -2
```

### How to delete all the archive files with extension *.tar.gz and greater than 100MB?

Please be careful while executing the following command as you don't want to delete the files by mistake. The best practice is to execute the same command with ls -l to make sure you know which files will get deleted when you execute the command with rm.

```
# find / -type f -name *.tar.gz -size +100M -exec ls -l {} \;

# find / -type f -name *.tar.gz -size +100M -exec rm -f {} \;
```

### How to archive all the files that are not modified in the last x number of days?

The following command finds all the files not modified in the last 60 days under /home/jsmith directory and creates an archive files under /tmp in the format of ddmmyyyy_archive.tar.

```
# find /home/jsmith -type f -mtime +60 | xargs tar -cvf
/tmp/`date '+%d%m%Y'_archive.tar`
```

On a side note, you can perform lot of file related activities (including finding files) using midnight commander GUI, a powerful text based file manager for Unix.

## Hack 19. Suppress Standard Output and Error Message

Sometime while debugging a shell script, you may not want to see either the standard output or standard error message. Use /dev/null as shown below for suppressing the output.

### Suppress standard output using > /dev/null

This will be very helpful when you are debugging shell scripts, where you don't want to display the echo statement and interested in only looking at the error messages.

```
# cat file.txt > /dev/null

# ./shell-script.sh > /dev/null
```

### Suppress standard error using 2> /dev/null

This is also helpful when you are interested in viewing only the standard output and don't want to view the error messages.

```
# cat invalid-file-name.txt 2> /dev/null

# ./shell-script.sh 2> /dev/null
```

## Hack 20. Join Command

Join command combines lines from two files based on a common field.

In the example below, we have two files – employee.txt and salary.txt. Both have employee-id as common field. So, we can use join command to combine

the data from these two files using employee-id as shown below.

```
$ cat employee.txt

100 Jason Smith
200 John Doe
300 Sanjay Gupta
400 Ashok Sharma

$ cat bonus.txt

100 $5,000
200 $500
300 $3,000
400 $1,250

$ join employee.txt bonus.txt

100 Jason Smith $5,000
200 John Doe $500
300 Sanjay Gupta $3,000
400 Ashok Sharma $1,250
```

## Hack 21. Change the Case

### Convert a file to all upper-case

```
$ cat employee.txt

100 Jason Smith
200 John Doe
300 Sanjay Gupta
400 Ashok Sharma

$ tr a-z A-Z < employee.txt

100 JASON SMITH
200 JOHN DOE
300 SANJAY GUPTA
```

```
400 ASHOK SHARMA
```

### Convert a file to all lower-case

```
$ cat department.txt

100 FINANCE
200 MARKETING
300 PRODUCT DEVELOPMENT
400 SALES

$ tr A-Z a-z < department.txt

100 finance
200 marketing
300 product development
400 sales
```

## Hack 22. Xargs Command

xargs is a very powerful command that takes output of a command and pass it as argument of another command. Following are some practical examples on how to use xargs effectively.

1. When you are trying to delete too many files using rm, you may get error message: /bin/rm Argument list too long – Linux. Use xargs to avoid this problem.

```
find ~ -name '*.log' -print0 | xargs -0 rm -f
```

2. Get a list of all the *.conf file under /etc/. There are different ways to get the same result. Following example is only to demonstrate the use of xargs. The output of the find command in this example is passed to the ls -l one by one using xargs.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
# find /etc -name "*.conf" | xargs ls –l
```

3. If you have a file with list of URLs that you would like to download, you can use xargs as shown below.

```
# cat url-list.txt | xargs wget –c
```

4. Find out all the jpg images and archive it.

```
# find / -name *.jpg -type f -print | xargs tar -cvzf
images.tar.gz
```

5. Copy all the images to an external hard-drive.

```
# ls *.jpg | xargs -n1 -i cp {} /external-hard-
drive/directory
```

## Hack 23. Sort Command

Sort command sorts the lines of a text file. Following are several practical examples on how to use the sort command based on the following sample text file that has employee information in the format:

```
employee_name:employee_id:department_name.
```

```
$ cat names.txt

Emma Thomas:100:Marketing
Alex Jason:200:Sales
Madison Randy:300:Product Development
Sanjay Gupta:400:Support
Nisha Singh:500:Sales
```

### Sort a text file in ascending order

```
$ sort names.txt

Alex Jason:200:Sales
Emma Thomas:100:Marketing
Madison Randy:300:Product Development
Nisha Singh:500:Sales
Sanjay Gupta:400:Support
```

### Sort a text file in descending order

```
$ sort -r names.txt

Sanjay Gupta:400:Support
Nisha Singh:500:Sales
Madison Randy:300:Product Development
Emma Thomas:100:Marketing
Alex Jason:200:Sales
```

### Sort a colon delimited text file on 2nd field (employee_id)

```
$ sort -t: -k 2 names.txt

Emma Thomas:100:Marketing
Alex Jason:200:Sales
Madison Randy:300:Product Development
Sanjay Gupta:400:Support
Nisha Singh:500:Sales
```

### Sort a tab delimited text file on 3rd field (department_name) and suppress duplicates

```
$ sort -t: -u -k 3 names.txt
```

```
Emma Thomas:100:Marketing
Madison Randy:300:Product Development
Alex Jason:200:Sales
Sanjay Gupta:400:Support
```

### Sort the passwd file by the 3rd field (numeric userid)

```
$ sort -t: -k 3n /etc/passwd | more

root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
```

### Sort /etc/hosts file by ip-addres

```
$ sort -t . -k 1,1n -k 2,2n -k 3,3n -k 4,4n /etc/hosts

127.0.0.1 localhost.localdomain localhost
192.168.100.101 dev-db.thegeekstuff.com dev-db
192.168.100.102 prod-db.thegeekstuff.com prod-db
192.168.101.20  dev-web.thegeekstuff.com dev-web
192.168.101.21  prod-web.thegeekstuff.com prod-web
```

### Combine sort with other commands

- **ps –ef | sort** : Sort the output of process list
- **ls -al | sort +4n** : List the files in the ascending order of the file-size. i.e sorted by 5th filed and displaying smallest files first.
- **ls -al | sort +4nr** : List the files in the descending order of the file-size. i.e sorted by 5th filed and displaying largest files first.

## Hack 24. Uniq Command

Uniq command is mostly used in combination with sort command, as uniq removes duplicates only from a sorted file. i.e In order for uniq to work, all the duplicate entries should be in the adjacent lines. Following are some common examples.

1. When you have an employee file with duplicate entries, you can do the following to remove duplicates.

```
$ sort namesd.txt | uniq

$ sort –u namesd.txt
```

2. If you want to know how many lines are duplicates, do the following. The first field in the following examples indicates how many duplicates where found for that particular line. So, in this example the lines beginning with Alex and Emma were found twice in the namesd.txt file.

```
$ sort namesd.txt | uniq –c

      2 Alex Jason:200:Sales
      2 Emma Thomas:100:Marketing
      1 Madison Randy:300:Product Development
      1 Nisha Singh:500:Sales
      1 Sanjay Gupta:400:Support
```

3. The following displays only the entries that are duplicates.

```
$ sort namesd.txt | uniq –cd

      2 Alex Jason:200:Sales
      2 Emma Thomas:100:Marketing
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

## Hack 25. Cut Command

Cut command can be used to display only specific columns from a text file or other command outputs.

Following are some of the examples.

### Display the 1st field (employee name) from a colon delimited file

```
$ cut -d: -f 1 names.txt

Emma Thomas
Alex Jason
Madison Randy
Sanjay Gupta
Nisha Singh
```

### Display 1st and 3rd field from a colon delimited file

```
$ cut -d: -f 1,3 names.txt

Emma Thomas:Marketing
Alex Jason:Sales
Madison Randy:Product Development
Sanjay Gupta:Support
Nisha Singh:Sales
```

### Display only the first 8 characters of every line in a file

```
$ cut -c 1-8 names.txt

Emma Tho
Alex Jas
Madison
Sanjay G
Nisha Si
```

### Misc Cut command examples

- **cut -d: -f1 /etc/passwd** Displays the unix login names for all the users in the system.
- **free | tr -s ' ' | sed '/^Mem/!d' | cut -d" " -f2** Displays the total memory available on the system.

## Hack 26. Stat Command

Stat command can be used either to check the status/properties of a single file or the filesystem.

### Display statistics of a file or directory.

```
$ stat /etc/my.cnf

  File: `/etc/my.cnf'
  Size: 346 Blocks: 16 IO Block: 4096   regular file
Device: 801h/2049d      Inode: 279856      Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid:
(    0/    root)
Access: 2009-01-01 02:58:30.000000000 -0800
Modify: 2006-06-01 20:42:27.000000000 -0700
Change: 2007-02-02 14:17:27.000000000 -0800

$ stat /home/ramesh

  File: `/home/ramesh'
  Size: 4096            Blocks: 8          IO Block:
4096   directory
Device: 803h/2051d      Inode: 5521409     Links: 7
Access: (0755/drwxr-xr-x)  Uid: (  401/ramesh)   Gid: (
401/ramesh)
Access: 2009-01-01 12:17:42.000000000 -0800
Modify: 2009-01-01 12:07:33.000000000 -0800
Change: 2009-01-09 12:07:33.000000000 -0800
```

### Display the status of the filesystem using option –f

```
$ stat -f /

  File: "/"
    ID: 0        Namelen: 255     Type: ext2/ext3
Blocks: Total: 2579457    Free: 2008027    Available:
1876998    Size: 4096
Inodes: Total: 1310720    Free: 1215892
```

## Hack 27. Diff Command

diff command compares two different files and reports the difference. The output is very cryptic and not straight forward to read.

```
Syntax: diff [options] file1 file2
```

### What was modified in my new file when compare to my old file?

The option -w in the diff command will ignore the white space while performing the comparison.

In the following diff output:

- The lines above ---, indicates the changes happened in first file in the diff command (i.e name_list.txt).
- The lines below ---, indicates the changes happened to the second file in the diff command (i.e name_list_new.txt). The lines that belong to the first file starts with < and the lines of second file starts with >.

```
# diff -w name_list.txt name_list_new.txt
```

```
2c2,3
< John Doe
---
> John M Doe
> Jason Bourne
```

## Hack 28. Display total connect time of users

Ac command will display the statistics about the user's connect time.

### Connect time for the current logged in user

With the option –d, it will break down the output for the individual days. In this example, I've been logged in to the system for more than 6 hours today. On Dec 1st, I was logged in for about 1 hour.

```
$ ac –d

Dec   1  total        1.08
Dec   2  total        0.99
Dec   3  total        3.39
Dec   4  total        4.50
Today    total        6.10
```

### Connect time for all the users

To display connect time for all the users use –p as shown below. Please note that this indicates the cumulative connect time for the individual users.

```
$ ac -p

        john                                    3.64
        madison                               0.06
        sanjay                                  88.17
        nisha                                   105.92
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
        ramesh                                  111.42
        total 309.21
```

### Connect time for a specific user

To get a connect time report for a specific user, execute the following:

```
$ ac -d sanjay

Jul  2  total       12.85
Aug 25  total        5.05
Sep  3  total        1.03
Sep  4  total        5.37
Dec 24  total        8.15
Dec 29  total        1.42
Today   total        2.95
```

# Chapter 5: PS1, PS2, PS3, PS4 and PROMPT_COMMAND

## Hack 29. PS1 - Default Interaction Prompt

The default interactive prompt on your Linux can be modified as shown below to something useful and informative. In the following example, the default PS1 was "\s-\v\$", which displays the shell name and the version number. Let us change this default behavior to display the username, hostname and current working directory name as shown below.

```
-bash-3.2$  export PS1="\u@\h \w> "

ramesh@dev-db ~> cd /etc/mail

ramesh@dev-db /etc/mail>

[Note: Prompt changed to "username@hostname current-
dir>" format]
```

Following PS1 codes are used in this example:

- \u - Username
- \h - Hostname
- \w - Full pathname of current directory. Please note that when you are in the home directory, this will display only ~ as shown above

Note that there is a space at the end in the value of PS1. Personally, I prefer a space at the end of the prompt for better readability.

Make this setting permanent by adding export PS1="\u@\h \w> " to either .bash_profile (or) .bashrc as shown below.

```
ramesh@dev-db ~> vi ~/.bash_profile

ramesh@dev-db ~> vi ~/.bashrc

[Note: Add export PS1="\u@\h \w> " to one of the above
files]
```

Refer to the next chapter for several practical examples of PS1 usage in detail.

## Hack 30. PS2 - Continuation Interactive Prompt

A very long command can be broken down to multiple lines by giving \ at the end of the line. The default interactive prompt for a multi-line command is "> ". Let us change this default behavior to display "continue->" by using PS2 environment variable as shown below.

```
ramesh@dev-db ~> myisamchk --silent --force --fast --
update-state \
> --key_buffer_size=512M --sort_buffer_size=512M \
> --read_buffer_size=4M --write_buffer_size=4M \
> /var/lib/mysql/bugs/*.MYI

[Note: This uses the default ">" for continuation
prompt]

ramesh@dev-db ~> export PS2="continue-> "

ramesh@dev-db ~> myisamchk --silent --force --fast --
update-state \
continue-> --key_buffer_size=512M --
sort_buffer_size=512M \
continue-> --read_buffer_size=4M --write_buffer_size=4M
\
continue-> /var/lib/mysql/bugs/*.MYI

[Note: This uses the modified "continue-> " for
```

```
continuation prompt]
```

I found it very helpful and easy to read, when I break my long commands into multiple lines using \. I have also seen others who don't like to break-up long commands.

## Hack 31. PS3 - Prompt used by "select" inside shell script

You can define a custom prompt for the select loop inside a shell script, using the PS3 environment variable, as explained below.

**Shell script and output WITHOUT PS3:**

```
ramesh@dev-db ~> cat ps3.sh
select i in mon tue wed exit
do
  case $i in
    mon) echo "Monday";;
    tue) echo "Tuesday";;
    wed) echo "Wednesday";;
    exit) exit;;
  esac
done

ramesh@dev-db ~> ./ps3.sh
1) mon
2) tue
3) wed
4) exit
#? 1
Monday
#? 4

[Note: This displays the default "#?" for select
command prompt]
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

**Shell script and output WITH PS3:**

```
ramesh@dev-db ~> cat ps3.sh
PS3="Select a day (1-4): "
select i in mon tue wed exit
do
  case $i in
    mon) echo "Monday";;
    tue) echo "Tuesday";;
    wed) echo "Wednesday";;
    exit) exit;;
  esac
done

ramesh@dev-db ~> ./ps3.sh
1) mon
2) tue
3) wed
4) exit
Select a day (1-4): 1
Monday
Select a day (1-4): 4

[Note: This displays the modified "Select a day (1-4):"
for select command prompt]
```

## Hack 32. PS4 - Used by “set -x” to prefix tracing output

The PS4 shell variable defines the prompt that gets displayed, when you execute a shell script in debug mode as shown below.

**Shell script and output WITHOUT PS4:**

```
ramesh@dev-db ~> cat ps4.sh

set -x
echo "PS4 demo script"
ls -l /etc/ | wc -l
du -sh ~

ramesh@dev-db ~> ./ps4.sh
       ++ echo 'PS4 demo script'
PS4 demo script
++ ls -l /etc/
++ wc -l
243
++ du -sh /home/ramesh
48K     /home/ramesh
```

[**Note**: This displays the default "++" while tracing the output using set -x]

**Shell script and output WITH PS4:**

The PS4 defined below in the ps4.sh has the following two codes:

- $0 - indicates the name of script
- $LINENO - displays the current line number within the script

```
ramesh@dev-db ~> cat ps4.sh

export PS4='$0.$LINENO+ '
set -x
echo "PS4 demo script"
ls -l /etc/ | wc -l
du -sh ~

ramesh@dev-db ~> ./ps4.sh
../ps4.sh.3+ echo 'PS4 demo script'
PS4 demo script
../ps4.sh.4+ ls -l /etc/
```

```
../ps4.sh.4+ wc -l
243
../ps4.sh.5+ du -sh /home/ramesh
48K     /home/ramesh

[Note: This displays the modified "{script-name}.{line-
number}+" while tracing the output using set -x]
```

## Hack 33. PROMPT_COMMAND

Bash shell executes the content of the PROMPT_COMMAND just before displaying the PS1 variable.

```
ramesh@dev-db ~> export PROMPT_COMMAND="date +%k:%m:%S"

22:08:42
ramesh@dev-db ~>

[Note: This displays the PROMPT_COMMAND and PS1 output
on different lines]
```

If you want to display the value of PROMPT_COMMAND in the same line as the PS1, use the echo -n as shown below.

```
ramesh@dev-db ~> export PROMPT_COMMAND="echo -n [$(date
+%k:%m:%S)]"

[22:08:51]ramesh@dev-db ~>

[Note: This displays the PROMPT_COMMAND and PS1 output
on the same line]
```

# Chapter 6: Colorful and Functional Shell Prompt Using PS1

## Hack 34. Display username, hostname and basename of directory in the prompt

The PS1 in this example displays following three information in the prompt:

- \u - Username
- \h - Hostname
- \W - Base name of the current working directory

```
-bash-3.2$ export PS1="\u@\h \W> "

ramesh@dev-db ~> cd /etc/mail

ramesh@dev-db mail>
```

## Hack 35. Display current time in the prompt

In the PS1 environment variable, you can directly execute any Linux command, by specifying in the format $(linux_command). In the following example, the command $(date) is executed to display the current time inside the prompt.

```
ramesh@dev-db ~> export PS1="\u@\h [\$(date
+%k:%m:%S)]> "

ramesh@dev-db [11:09:56]>
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

You can also use \t to display the current time in the hh:mm:ss format as shown below:

```
ramesh@dev-db ~> export PS1="\u@\h [\t]> "

ramesh@dev-db [12:42:55]>
```

You can also use \@ to display the current time in 12-hour am/pm format as shown below:

```
ramesh@dev-db ~> export PS1="[\@] \u@\h> "

[04:12 PM] ramesh@dev-db>
```

## Hack 36. Display output of any command in the prompt

You can display output of any Linux command in the prompt. The following example displays three items separated by | (pipe) in the command prompt:

- \!: The history number of the command
- \h: hostname
- $kernel_version: The output of the uname -r command from $kernel_version variable
- \$?: Status of the last command

```
ramesh@dev-db ~> kernel_version=$(uname -r)

ramesh@dev-db ~> export PS1="\!|\h|$kernel_version|\$?> "

473|dev-db|2.6.25-14.fc9.i686|0>
```

## Hack 37. Change foreground color of the prompt

Display prompt in blue color, along with username, host and current directory information

```
$ export PS1="\e[0;34m\u@\h \w> \e[m "

[Note: This is for light blue prompt]

$ export PS1="\e[1;34m\u@\h \w> \e[m "

[Note: This is for dark blue prompt]
```

- \e[ - Indicates the beginning of color prompt
- x;ym - Indicates color code. Use the color code values mentioned below.
- \e[m - indicates the end of color prompt

**Color Code Table:**

```
Black 0;30
Blue 0;34
Green 0;32
Cyan 0;36
Red 0;31
Purple 0;35
Brown 0;33

[Note: Replace 0 with 1 for dark color]
```

Make the color change permanent by adding the following lines your ~/.bash_profile or ~/.bashrc

```
$ vi ~/.bash_profile
```

```
STARTCOLOR='\e[0;34m';
ENDCOLOR="\e[0m"
export PS1="$STARTCOLOR\u@\h \w> $ENDCOLOR"
```

## Hack 38. Change background color of the prompt

Change the background color by specifying \e[{code}m in the PS1 prompt as shown below.

```
$ export PS1="\e[47m\u@\h \w> \e[m "

[Note: This is for Light Gray background]
```

Combination of background and foreground.

```
$ export PS1="\e[0;34m\e[47m\u@\h \w> \e[m "

[Note: This is for Light Blue foreground and Light Gray
background]
```

Add the following to your ~/.bash_profile or ~/.bashrc to make the above background and foreground color permanent.

```
$ vi ~/.bash_profile
STARTFGCOLOR='\e[0;34m';
STARTBGCOLOR="\e[47m"
ENDCOLOR="\e[0m"
export PS1="$STARTFGCOLOR$STARTBGCOLOR\u@\h \w>
$ENDCOLOR"
```

Play around by using the following background color and choose the one that match your taste:

- \e[40m

- \e[41m
- \e[42m
- \e[43m
- \e[44m
- \e[45m
- \e[46m
- \e[47m

## Hack 39. Display multiple colors in the prompt

You can also display multiple colors in the same prompt. Add the following function to your ~/.bash_profile

```
function prompt {
  local BLUE="\[\033[0;34m\]"
  local DARK_BLUE="\[\033[1;34m\]"
  local RED="\[\033[0;31m\]"
  local DARK_RED="\[\033[1;31m\]"
  local NO_COLOR="\[\033[0m\]"
  case $TERM in
    xterm*|rxvt*)
      TITLEBAR='\[\033]0;\u@\h:\w\007\]'
      ;;
    *)
      TITLEBAR=""
      ;;
  esac
  PS1="\u@\h [\t]> "
  PS1="${TITLEBAR}\
  $BLUE\u@\h $RED[\t]>$NO_COLOR "
  PS2='continue-> '
  PS4='$0.$LINENO+ '
}
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

You can re-login for the changes to take effect or source the .bash_profile as shown below.

```
$. ./.bash_profile

$ prompt

ramesh@dev-db [13:02:13]>
```

## Hack 40. Change the prompt color using tput

You can also change color of the PS1 prompt using tput as shown below:

```
$ export PS1="\[$(tput bold)$(tput setb 4)$(tput setaf
7)\]\u@\h:\w $ \[$(tput sgr0)\]“
```

tput Color Capabilities:

- tput setab [1-7] - Set a background color using ANSI escape
- tput setb [1-7] - Set a background color
- tput setaf [1-7] - Set a foreground color using ANSI escape
- tput setf [1-7] - Set a foreground color

tput Text Mode Capabilities:

- tput bold - Set bold mode
- tput dim - turn on half-bright mode
- tput smul - begin underline mode
- tput rmul - exit underline mode
- tput rev - Turn on reverse mode

- tput smso - Enter standout mode (bold on rxvt)
- tput rmso - Exit standout mode
- tput sgr0 - Turn off all attributes

Color Code for tput:

- 0 – Black
- 1 – Red
- 2 – Green
- 3 – Yellow
- 4 – Blue
- 5 – Magenta
- 6 – Cyan
- 7 - White

## Hack 41. Create your own prompt using the available codes for PS1 variable

Use the following codes and create your own personal PS1 Linux prompt that is functional and suites your taste.

- \a an ASCII bell character (07)
- \d the date in “Weekday Month Date” format (e.g., “Tue May 26”)
- \D{format} - the format is passed to strftime(3) and the result is inserted into the prompt string; an empty format results in a locale-specific time representation. The braces are required
- \e an ASCII escape character (033)

- \h the hostname up to the first part
- \H the hostname
- \j the number of jobs currently managed by the shell
- \l the basename of the shell's terminal device name
- \n newline
- \r carriage return
- \s the name of the shell, the basename of $0 (the portion following the final slash)
- \t the current time in 24-hour HH:MM:SS format
- \T the current time in 12-hour HH:MM:SS format
- \@ the current time in 12-hour am/pm format
- \A the current time in 24-hour HH:MM format
- \u the username of the current user
- \v the version of bash (e.g., 2.00)
- \V the release of bash, version + patch level (e.g., 2.00.0)
- \w the current working directory, with $HOME abbreviated with a tilde
- \W the basename of the current working directory, with $HOME abbreviated with a tilde
- \! the history number of this command
- \# the command number of this command
- \$ if the effective UID is 0, a #, otherwise a $
- \nnn the character corresponding to the octal number nnn
- \\ a backslash
- \[ begin a sequence of non-printing characters, which could be used to embed a terminal control sequence into the prompt

- \] end a sequence of non-printing character

## Hack 42. Use bash shell function inside PS1 variable

You can also invoke a bash shell function in the PS1 as shown below.

```
ramesh@dev-db ~> function httpdcount {
>  ps aux | grep httpd | grep -v grep | wc -l
> }

ramesh@dev-db ~> export PS1="\u@\h [`httpdcount`]> "

ramesh@dev-db [12]>

[Note: This displays the total number of running httpd
processes]
```

You can add the following line to your ~/.bash_profile or ~/.bashrc to make this change permanent:

```
$ vi .bash_profile
function httpdcount {
  ps aux | grep httpd | grep -v grep | wc -l
}
export PS1='\u@\h [`httpdcount`]> '
```

## Hack 43. Use shell script inside PS1 variable

You can also invoke a shell script inside the PS1 variable. In the example below, the ~/bin/totalfilesize.sh, which calculates the total filesize of the current directory, is invoked inside the PS1 variable.

```
ramesh@dev-db ~> cat ~/bin/totalfilesize.sh
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
for filesize in $(ls -l . | grep "^-" | awk '{print
$5}')
do
  let totalsize=$totalsize+$filesize
done
echo -n "$totalsize"

ramesh@dev-db ~> export PATH=$PATH:~/bin

ramesh@dev-db ~> export PS1="\u@\h
[\$(totalfilesize.sh) bytes]> "

ramesh@dev-db [534 bytes]> cd /etc/mail

ramesh@dev-db [167997 bytes]>
```

[**Note**: This executes the totalfilesize.sh to display the total file size of the current directory in the PS1 prompt]

# Chapter 7: Archive and Compression

## Hack 44. Zip command basics

**How to zip multiple files?**

```
syntax: zip {.zip file-name} {file-names}
```

```
# zip var-log-files.zip /var/log/*
  adding: var/log/acpid (deflated 81%)
  adding: var/log/anaconda.log (deflated 79%)
  adding: var/log/anaconda.syslog (deflated 73%)
  adding: var/log/anaconda.xlog (deflated 82%)
  adding: var/log/audit/ (stored 0%)
  adding: var/log/boot.log (stored 0%)
  adding: var/log/boot.log.1 (deflated 40%)
  adding: var/log/boot.log.2 (deflated 42%)
  adding: var/log/boot.log.3 (deflated 40%)
  adding: var/log/boot.log.4 (deflated 40%)
```

**How to zip a directory and it's files recursively?**

```
# zip -r var-log-dir.zip /var/log/
updating: var/log/ (stored 0%)
  adding: var/log/wtmp (deflated 78%)
  adding: var/log/scrollkeeper.log (deflated 94%)
  adding: var/log/rpmpkgs.3 (deflated 68%)
  adding: var/log/spooler (stored 0%)
  adding: var/log/cron.2 (deflated 90%)
  adding: var/log/spooler.1 (stored 0%)
  adding: var/log/spooler.4 (stored 0%)
  adding: var/log/httpd/ (stored 0%)
  adding: var/log/rpmpkgs.1 (deflated 68%)
  adding: var/log/anaconda.log (deflated 79%)
  adding: var/log/secure.2 (deflated 93%)
```

### How to unzip a *.zip compressed file?

```
# unzip var-log.zip
Archive:  var-log.zip
  inflating: var/log/acpid
  inflating: var/log/anaconda.log
  inflating: var/log/anaconda.syslog
  inflating: var/log/anaconda.xlog
   creating: var/log/audit/
```

To see a detailed output during unzip pass the -v option as shown below.

```
# unzip -v var-log.zip

Archive:  var-log.zip
 Length   Method    Size  Ratio   Date   Time   CRC-32
Name
--------  ------  ------- -----   ----   ----   ------    ---
-
    1916  Defl:N      369  81%  02-08-08 14:27  e2ffdc0c
var/log/acpid
   13546  Defl:N     2900  79%  02-02-07 14:25  34cc03a1
var/log/anaconda.log

skip..

    7680  Defl:N      411  95%  12-30-08 10:55  fe876ee9
var/log/wtmp.1
   40981  Defl:N     7395  82%  02-08-08 14:28  6386a95e
var/log/Xorg.0.log
--------          -------  ---                            ---
----
41406991          2809229  93%                            56
files
```

### How to list a content of zip file with uncompressing it?

```
# unzip -l var-log.zip

Archive:  var-log.zip
  Length     Date   Time    Name
 --------    ----   ----    ----
     1916  02-08-08 14:27   var/log/acpid
```

```
    13546  02-02-07 14:25   var/log/anaconda.log

..skip..

    40981  02-08-08 14:28   var/log/Xorg.0.log
    40981  02-08-07 14:56   var/log/Xorg.0.log.old
 --------                   -------
 41406991                   56 files
```

## Hack 45. Advanced compression using zip command.

There are 10 levels of compression provided by zip command.

- Level 0 is the lowest level, where it just archives the file without any compression.
- Level 1 will perform little compression. But, will be very fast.
- Level 6 is the default level of compression.
- Level 9 is the maximum compression. This will be slower when compared to default level. In my opinion, unless you are compressing a huge file, you should always use level 9.

In the example below, I used Level 0, default Level 6, and Level 9 compression on a same directory. See the compressed file size yourself.

```
# zip var-log-files-default.zip /var/log/*

# zip -0 var-log-files-0.zip /var/log/*

# zip -9 var-log-files-9.zip /var/log/*

# ls -ltr
-rw-r--r--   1 root       root        2817248 Jan  1 13:05
var-log-files-default.zip
-rw-r--r--   1 root       root       41415301 Jan  1 13:05
var-log-files-0.zip
-rw-r--r--   1 root       root        2582610 Jan  1 13:06
var-log-files-9.zip
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

## Hack 46. Password Protection of Zip files

Pass the option -P to the zip command to assign a password to the zip file.

```
# zip -P mysecurepwd var-log-protected.zip /var/log/*
```

The above option is good if you are using the command inside a shell-script for background jobs. However, when you are performing the compression interactively on the command-line, you don't want the password to be visible in the history. So, use the option -e as shown below to assign the password.

```
# zip -e var-log-protected.zip /var/log/*
Enter password:
Verify password:
updating: var/log/acpid (deflated 81%)
updating: var/log/anaconda.log (deflated 79%)
```

When you are uncompressing a password protected file, it will ask for the password as shown below.

```
# unzip  var-log-protected.zip
Archive:  var-log-protected.zip
[var-log-protected.zip] var/log/acpid password:
```

## Hack 47. Validate a zip archive

Sometime you may want to validate a zip archive without extracting it. To test the validity of the zip file, pass option -t as shown below.

```
# unzip -t var-log.zip

     Archive:  var-log.zip
    testing: var/log/acpid            OK
    testing: var/log/anaconda.log     OK
```

```
    testing: var/log/anaconda.syslog   OK

skip...

    testing: var/log/wtmp              OK
    testing: var/log/wtmp.1            OK
    testing: var/log/Xorg.0.log        OK

No errors detected in compressed data of var-log.zip.
```

## Hack 48. Tar Command Basics

tar command (tape archive) is used to convert a group of files into an archive.

```
Syntax: tar [options] [tar-archive-name] [other-file-
names]
```

**How can I create a single backup file of all files and subdirectories under my home directory?**

The following command creates a single archive backup file called my_home_directory.tar under /tmp. This archive will contain all the files and subdirectories under /home/jsmith.

- Option c, stands for create an archive.
- Option v stands for verbose mode, displays additional information while executing the command.
- Option f indicates the archive file name mentioned in the command.

```
# tar cvf /tmp/my_home_directory.tar /home/jsmith
```

### How do I view all the files inside the tar archive?

Option t will display all the files from the tar archive.

```
# tar tvf /tmp/my_home_directory.tar
```

### How do I extract all the files from a tar archive?

Option x will extract the files from the tar archive as shown below. This will extract the content to the current directory location from where the command is executed.

```
# tar xvf /tmp/my_home_directory.tar
```

### How do I extract tar.gz files to a specific directory?

```
# tar xvfz /tmp/my_home_directory.tar.gz –C
/home/ramesh
```

## Hack 49. Combine gzip, bzip2 with tar

### How to use gzip with tar?

Add *option z* to the tar command when dealing with tar.gz compressed file.

```
# tar cvfz /tmp/my_home_directory.tar.gz /home/jsmith

# tar xvfz /tmp/my_home_directory.tar.gz

# tar tvfz /tmp/my_home_directory.tar.gz
```

**Note:** Using gzip is faster when compared to bzip2.

### How to use bzip2 with tar?

Add *option j* to the tar command when dealing with tar.bz2 compressed file.

```
# tar cvfj /tmp/my_home_directory.tar.bz2 /home/jsmith

# tar xvfj /tmp/my_home_directory.tar.bz2

# tar tvfj /tmp/my_home_directory.tar.bz2
```

**Note:** Using bizp2 gives higher level of compression when compared to gzip.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

# Chapter 8: Command Line History

When you are using Linux command line frequently, using the history effectively can be a major productivity boost. In fact, once you have mastered the 15 examples that I've provided here, you'll find using command line more enjoyable and fun.

## Hack 50. Display TIMESTAMP in history using HISTTIMEFORMAT

Typically when you type history from command line, it displays the command# and the command. For auditing purpose, it may be beneficial to display the timestamp along with the command as shown below.

```
# export HISTTIMEFORMAT='%F %T '

# history | more
      1  2008-08-05 19:02:39 service network restart
      2  2008-08-05 19:02:39 exit
      3  2008-08-05 19:02:39 id
      4  2008-08-05 19:02:39 cat /etc/redhat-release
```

**Note**: You can also setup the following alias to view the recent history commands.

```
alias h1='history 10'
alias h2='history 20'
alias h3='history 30'
```

## Hack 51. Search the history using Control+R

I strongly believe that this may be your most frequently used feature of history. When you've already executed a very long command, you can simply

search history using a keyword and re-execute the same command without having to type it fully. Press Control+R and type the keyword.

In the following example, I searched for red, which displayed the previous command “cat /etc/redhat-release” in the history that contained the word red.

```
# [Note: Press Ctrl+R from the command prompt, which
will display the reverse-i-search prompt as shown
below]

(reverse-i-search)`red': cat /etc/redhat-release
[Note: Press enter when you see your command, which
will execute the command from the history]

# cat /etc/redhat-release
Fedora release 9 (Sulphur)
```

Sometimes you want to edit a command from history before executing it. For e.g. you can search for httpd, which will display service httpd stop from the command history, select this command and change the stop to start and re-execute it again as shown below.

```
# [Note: Press Ctrl+R from the command prompt, which
will display the reverse-i-search prompt]

(reverse-i-search)`httpd': service httpd stop
[Note: Press either left arrow or right arrow key when
you see your command, which will display the command
for you to edit, before executing it]

# service httpd start
```

## Hack 52. Repeat previous command quickly using 4 different methods

Sometime you may end up repeating the previous commands for various reasons. Following are the 4 different ways to repeat the last executed command.

1. Use the **up arrow** to view the previous command and press enter to execute it.
2. **Type !!** and press enter from the command line
3. **Type !-1** and press enter from the command line.
4. **Press Control+P** will display the previous command, press enter to execute it

## Hack 53. Execute a specific command from history

In the following example, If you want to repeat the command #4, execute !4 as shown below.

```
# history | more
       1  service network restart
       2  exit
       3  id
       4  cat /etc/redhat-release

# !4
cat /etc/redhat-release
Fedora release 9 (Sulphur)
```

## Hack 54. Execute previous command that starts with a specific word

Type ! followed by the starting few letters of the command that you would like to re-execute. In the following example, typing !ps and enter, executed the previous command starting with ps, which is 'ps aux | grep yp'.

```
# !ps
ps aux | grep yp
       root     16947  0.0  0.1  36516  1264 ?
Sl   13:10   0:00 ypbind
       root     17503  0.0  0.0   4124   740 pts/0
S+   19:19   0:00 grep yp
```

## Hack 55. Control the total number of lines in the history using HISTSIZE

Append the following two lines to the .bash_profile and relogin to the bash shell again to see the change. In this example, only 450 command will be stored in the bash history.

```
# vi ~/.bash_profile

HISTSIZE=450
HISTFILESIZE=450
```

## Hack 56. Change the history file name using HISTFILE

By default, history is stored in ~/.bash_history file. Add the following line to the .bash_profile and relogin to the bash shell, to store the history command in .commandline_warrior file instead of .bash_history file. I'm yet to figure out a practical use for this. I can see this getting used when you want to track commands executed from different terminals using different history file name.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
# vi ~/.bash_profile
HISTFILE=/root/.commandline_warrior
```

## Hack 57. Eliminate the continuous repeated entry from history using HISTCONTROL

In the following example pwd was typed three times, when you do history, you can see all the 3 continuous occurrences of it. To eliminate duplicates, set HISTCONTROL to ignoredups as shown below.

```
# pwd

# pwd

# pwd

# history | tail -4
       44  pwd
       45  pwd
       46  pwd
       47  history | tail -4

[Note: There are three pwd commands in history, after
executing pwd 3 times as shown above]

# export HISTCONTROL=ignoredups

# pwd

# pwd

# pwd

# history | tail -3
       56  export HISTCONTROL=ignoredups
       57  pwd
       58  history | tail -4
```

```
[Note: There is only one pwd command in the history,
even after executing pwd 3 times as shown above]
```

## Hack 58. Erase duplicates across the whole history using HISTCONTROL

The ignoredups shown above removes duplicates only if they are consecutive commands. To eliminate duplicates across the whole history, set the HISTCONTROL to erasedups as shown below.

```
# export HISTCONTROL=erasedups

# pwd

# service httpd stop

# history | tail -3
       38  pwd
       39  service httpd stop
       40  history | tail -3

# ls -ltr

# service httpd stop

# history | tail -6
       35  export HISTCONTROL=erasedups
       36  pwd
       37  history | tail -3
       38  ls –ltr
       39  service httpd stop
       40  history | tail -6

[Note: The previous service httpd stop after pwd got
erased]
```

## Hack 59. Force history not to remember a particular command using HISTCONTROL

When you execute a command, you can instruct history to ignore the command by setting HISTCONTROL to ignorespace AND typing a space in front of the command as shown below. I can see lot of junior sysadmins getting excited about this, as they can hide a command from the history.

It is good to understand how ignorespace works. But, as a best practice, don't hide purposefully anything from history.

```
# export HISTCONTROL=ignorespace

# ls –ltr

# pwd

#  service httpd stop

[Note:  There is a space at the beginning of service,
to ignore this command from history]

# history | tail -3
       67  ls –ltr
       68  pwd
       69  history | tail -3
```

## Hack 60. Clear all the previous history using option -c

Sometime you may want to clear all the previous history. However you may still want to keep the history moving forward.

```
# history -c
```

## Hack 61. Substitute words from history commands

When you are searching through history, you may want to execute a different command but use the same parameter from the command that you've just searched.

In the example below, the !!:$ next to the vi command gets the argument from the previous command to the current command.

```
# ls anaconda-ks.cfg
anaconda-ks.cfg

# vi !!:$
vi anaconda-ks.cfg
```

In the example below, the !^ next to the vi command gets the first argument from the previous command (i.e cp command) to the current command (i.e vi command).

```
# cp anaconda-ks.cfg anaconda-ks.cfg.bak
anaconda-ks.cfg

# vi !^
vi anaconda-ks.cfg
```

## Hack 62. Substitute a specific argument for a specific command

In the example below, !cp:2 searches for the previous command in history that starts with cp and takes the second argument of cp and substitutes it for the ls -l command as shown below.

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
# cp ~/longname.txt /really/a/very/long/path/long-
filename.txt

# ls -l !cp:2
ls -l /really/a/very/long/path/long-filename.txt
```

In the example below, !cp:$ searches for the previous command in history that starts with cp and takes the last argument (in this case, which is also the second argument as shown above) of cp and substitutes it for the ls -l command as shown below.

```
# ls -l !cp:$
ls -l /really/a/very/long/path/long-filename.txt
```

## Hack 63. Disable the usage of history using HISTSIZE

If you want to disable history all together and don't want bash shell to remember the commands you've typed, set the HISTSIZE to 0 as shown below.

```
# export HISTSIZE=0

# history

# [Note:  History did not display anything]
```

## Hack 64. Ignore specific commands from the history using HISTIGNORE

Sometimes you may not want to clutter your history with basic commands such as pwd and ls. Use HISTIGNORE to specify all the commands that you want to ignore from the history.

Please note that adding ls to the HISTIGNORE ignores only ls and not ls -l. So, you have to provide the exact command that you would like to ignore from the history.

```
# export HISTIGNORE="pwd:ls:ls –ltr:"

# pwd

# ls

# ls -ltr

# service httpd stop

# history | tail -3
       79  export HISTIGNORE="pwd:ls:ls -ltr:"
       80  service httpd stop
       81  history

[Note: History did not display pwd and ls]
```

# Chapter 9: System Administration Tasks

## Hack 65. Partition using fdisk

After you've installed brand new disks on your server, you have to use tools like fdisk to partition it accordingly.

Following are the 5 typical actions (commands) that you can execute inside fdisk.

- n - New Partition creation
- d - Delete an existing partition
- p - Print Partition Table
- w - Write the changes to the partition table. i.e save.
- q - Quit the fdisk utility

### Create a partition

In the following example, I created a /dev/sda1 primary partition.

```
# fdisk /dev/sda

Device contains neither a valid DOS partition table,
nor Sun, SGI or OSF disklabel Building a new DOS
disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course,
the previous content won't be recoverable.

The number of cylinders for this disk is set to 34893.
There is nothing wrong with that, but this is larger
than 1024, and could in certain setups cause problems
with:
```

```
1) software that runs at boot time (e.g., old versions
of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
Warning: invalid flag 0x0000 of partition table 4 will
be corrected by w(rite)

Command (m for help): p

Disk /dev/sda: 287.0 GB, 287005343744 bytes
255 heads, 63 sectors/track, 34893 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot  Start   End      Blocks   Id  System

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-34893, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-34893,
default 34893):
Using default value 34893

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

### Verify that the partition got created successfully

```
# fdisk /dev/sda

The number of cylinders for this disk is set to 34893.
There is nothing wrong with that, but this is larger
than 1024, and could in certain setups cause problems
with:
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
1) software that runs at boot time (e.g., old versions
of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): p

Disk /dev/sda: 287.0 GB, 287005343744 bytes
255 heads, 63 sectors/track, 34893 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot  Start         End      Blocks   Id  System
/dev/sda1        1       34893   280277991   83  Linux

Command (m for help): q
```

## Hack 66. Format a partition using mke2fsk

After partitioning the disks, it is still not ready for usage, as we need to format the disk. At this stage, if you try to view the disk information, it will give the following error message indicating that no valid superblock is present.

```
# tune2fs -l /dev/sda1

tune2fs 1.35 (28-Feb-2004)
tune2fs: Bad magic number in super-block while trying
to open /dev/sda1

Couldn't find valid filesystem superblock.
```

To format the disk, use mke2fs as shown below.

```
# mke2fs /dev/sda1
```

You can also pass the following optional parameter to the mke2fs.

- -m 0 : reserved-blocks-percentage – This indicates the percentage of the filesystem blocks reserved for the root user. Default is 5%. In the following example, it is set to 0.
- -b 4096 : block-size specified in bytes. Valid values are 1024, 2048 and 4096 bytes per block.

```
# mke2fs -m 0 -b 4096 /dev/sda1

mke2fs 1.35 (28-Feb-2004)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
205344 inodes, 70069497 blocks
0 blocks (0.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=71303168
2139 block groups
32768 blocks per group, 32768 fragments per group
96 inodes per group
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736,
1605632, 2654208, 4096000, 7962624, 11239424, 20480000,
23887872

Writing inode tables: done
Writing superblocks and filesystem accounting
information: done

This filesystem will be automatically checked every 32
mounts or 180 days, whichever comes first.  Use tune2fs
-c or -i to override.
```

The above command will create an ext2 filesystem. To create an ext3 file system do the following:

```
# mkfs.ext3 /dev/sda1

# mke2fs –j /dev/sda1
```

## Hack 67. Mount the partition

After creating a partition and formatting, you can mount it to a mount point.

First create a directory where the partition should be mounted.

```
# mkdir /home/database
```

Mount the file system.

```
# mount /dev/sda1 /home/database
```

To automatically mount the filesystem after the reboot, add the following entry to the /etc/fstab

```
/dev/sdaa /home/database ext3 defaults 0 2
```

## Hack 68. Fine tune the partition using tune2fs

Use the tune2fs -l /dev/sda1 to view the filesystem information as shown below.

```
# tune2fs -l /dev/sda1

tune2fs 1.35 (28-Feb-2004)
Filesystem volume name:   /home/database
Last mounted on:          <not available>
Filesystem UUID:          f1234556-e123-1234-abcd-
bbbbaaaaae11
Filesystem magic number:  0xEF44
Filesystem revision #:    1 (dynamic)
Filesystem features:      resize_inode filetype
sparse_super
Default mount options:    (none)
Filesystem state:         not clean
```

```
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              1094912
Block count:              140138994
Reserved block count:     0
Free blocks:              16848481
Free inodes:              1014969
First block:              0
Block size:               2048
Fragment size:            2048
Reserved GDT blocks:      512
Blocks per group:         16384
Fragments per group:      16384
Inodes per group:         128
Inode blocks per group:   8
Filesystem created:       Tue Jul  1 00:06:03 2008
Last mount time:          Thu Aug 21 05:58:25 2008
Last write time:          Fri Jan  2 15:40:36 2009
Mount count:              2
Maximum mount count:      20
Last checked:             Tue Jul  1 00:06:03 2008
Check interval:           15552000 (6 months)
Next check after:         Sat Dec 27 23:06:03 2008
Reserved blocks uid:      0 (user root)
Reserved blocks gid:      0 (group root)
First inode:              11
Inode size:               128
Default directory hash:   tea
Directory Hash Seed:      12345829-1236-4123-9aaa-
ccccc123292b
```

You can also use the tune2fs to tune the ex2/ext3 filesystem parameter. For example, if you want to change the Filesystem volume name, you can do it as shown below.

```
# tune2fs -l /dev/sda1 | grep volume
Filesystem volume name:   /home/database

# tune2fs -L database-home /dev/emcpowera1
tune2fs 1.35 (28-Feb-2004)
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
# tune2fs -l /dev/sda1 | grep volume
Filesystem volume name:   database-home
```

## Hack 69. Create a swap file system.

Create a file for swap usage as shown below.

```
# dd if=/dev/zero of=/home/swap-fs bs=1M count=512
512+0 records in
512+0 records out

# ls -l /home/swap-fs
-rw-r--r--  1 root root 536870912 Jan  2 23:13
/home/swap-fs
```

Use mkswap to setup a Linux swap area in the /home/swap-fs file that was created above.

```
# mkswap /home/swap-fs

Setting up swapspace version 1, size = 536866 kB
```

Once the file is created and has been setup for Linux swap area, it is time to enable the swap using swapon as shown below.

```
# swapon /home/swap-fs
```

Add the following line to /etc/fstab and reboot the system for the swap to take into effect.

```
/home/swap-fs swap swap defaults 0 0
```

# Hack 70. Create a new user

### Add a new user – Basic method

Specify only the user name.

```
# useradd jsmith
```

### Add a new user with additional Parameter

You can also specify the following parameter to the useradd

- -c : Description about the user.
- -e : expiry date of the user in mm/dd/yy format

```
# adduser -c "John Smith - Oracle Developer" -e
12/31/09 jsmith
```

Verify that the user got added successfully.

```
# grep jsmith /etc/passwd
jsmith:x:510:510:John Smith - Oracle
Developer:/home/jsmith:/bin/bash
```

### Change the user password.

```
# passwd jsmith

Changing password for user jsmith.
New UNIX password:
BAD PASSWORD: it is based on a dictionary word
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

**Note:** Make sure to follow these best practices to create a strong password for the user.

### How to identify the default values used by useradd?

Following are the default values that will be used when an user is created.

```
# useradd -D

GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
```

## Hack 71. Create a new group and assign to an user

### Create a new developer group.

```
# groupadd developers
```

### Validate that the group was created successfully.

```
# grep developer /etc/group
developers:x:511:
```

### Add an user to an existing group.

You cannot use useradd to modify an existing user, as you'll get the following error message.

```
# useradd -G developers jsmith
useradd: user jsmith exists

# usermod -g developers jsmith
```

**Validate the users group was modified successfully.**

```
# grep jsmith /etc/passwd
jsmith:x:510:511:Oracle
Developer:/home/jsmith:/bin/bash

# id jsmith
uid=510(jsmith) gid=511(developers)
groups=511(developers)

# grep jsmith /etc/group
jsmith:x:510:
developers:x:511:jsmith
```

## Hack 72. Setup SSH passwordless login in OpenSSH

You can login to a remote Linux server without entering password in 3 simple steps using ssky-keygen and ssh-copy-id as explained in this example.

ssh-keygen creates the public and private keys. ssh-copy-id copies the local-host's public key to the remote-host's authorized_keys file. ssh-copy-id also assigns proper permission to the remote-host's home, ~/.ssh, and ~/.ssh/authorized_keys.

**Step 1: Create public and private keys using ssh-key-gen on local-host**

```
jsmith@local-host$ ssh-keygen

Generating public/private rsa key pair.
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
Enter file in which to save the key
(/home/jsmith/.ssh/id_rsa):[Enter key]
Enter passphrase (empty for no passphrase): [Press
enter key]
Enter same passphrase again: [Pess enter key]
Your identification has been saved in
/home/jsmith/.ssh/id_rsa.
Your public key has been saved in
/home/jsmith/.ssh/id_rsa.pub.
The key fingerprint is:
33:b3:fe:af:95:95:18:11:31:d5:de:96:2f:f2:35:f9
jsmith@local-host
```

### Step 2: Copy the public key to remote-host using ssh-copy-id

```
jsmith@local-host$ ssh-copy-id -i ~/.ssh/id_rsa.pub
remote-host

jsmith@remote-host's password:
Now try logging into the machine, with "ssh 'remote-
host'", and check in:
.ssh/authorized_keys to make sure we haven't added
extra keys that you weren't expecting.
```

**Note:** ssh-copy-id appends the keys to the remote-host's .ssh/authorized_key.

### Step 3: Login to remote-host without entering the password

```
jsmith@local-host$ ssh remote-host

Last login: Sun Nov 16 17:22:33 2008 from 192.168.1.2

[Note: SSH did not ask for password.]

jsmith@remote-host$ [Note: You are on remote-host here]
```

# Hack 73. Use ssh-copy-id along with ssh-agent

### Using ssh-copy-id along with the ssh-add/ssh-agent

When no value is passed for the option -i and If ~/.ssh/identity.pub is not available, ssh-copy-id will display the following error message.

```
jsmith@local-host$ ssh-copy-id -i remote-host

/usr/bin/ssh-copy-id: ERROR: No identities found
```

If you have loaded keys to the ssh-agent using the ssh-add, then ssh-copy-id will get the keys from the ssh-agent to copy to the remote-host. i.e, it copies the keys provided by ssh-add -L command to the remote-host, when you don't pass option -i to the ssh-copy-id.

```
jsmith@local-host$ ssh-agent $SHELL

jsmith@local-host$ ssh-add -L
The agent has no identities.

jsmith@local-host$ ssh-add
Identity added: /home/jsmith/.ssh/id_rsa
(/home/jsmith/.ssh/id_rsa)

jsmith@local-host$ ssh-add -L
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAsJIEILxftj8aSxMa3d8t6JvM79D
aHrtPhTYpq7kIEMUNzApnyxsHpH1tQ/Ow==
/home/jsmith/.ssh/id_rsa

jsmith@local-host$ ssh-copy-id -i remote-host
jsmith@remote-host's password:
Now try logging into the machine, with "ssh 'remote-
host'", and check in: .ssh/authorized_keys to make sure
we haven't added extra keys that you weren't expecting.
[Note: This has added the key displayed by ssh-add -L]
```

### Three Minor Annoyances of ssh-copy-id

Following are few minor annoyances of the ssh-copy-id.

1. Default public key: ssh-copy-id uses ~/.ssh/identity.pub as the default public key file (i.e when no value is passed to option -i). Instead, I wish it uses id_dsa.pub, or id_rsa.pub, or identity.pub as default keys. i.e If any one of them exist, it should copy that to the remote-host. If two or three of them exist, it should copy identity.pub as default.
2. The agent has no identities: When the ssh-agent is running and the ssh-add -L returns "The agent has no identities" (i.e no keys are added to the ssh-agent), the ssh-copy-id will still copy the message "The agent has no identities" to the remote-host's authorized_keys entry.
3. Duplicate entry in authorized_keys: I wish ssh-copy-id validates duplicate entry on the remote-host's authorized_keys. If you execute ssh-copy-id multiple times on the local-host, it will keep appending the same key on the remote-host's authorized_keys file without checking for duplicates. Even with duplicate entries everything works as expected. But, I would like to have my authorized_keys file clutter free.

## Hack 74. Crontab

Using cron you can execute a shell-script or Linux commands at a specific time and date. For example a sysadmin can schedule a backup job that can run every day.

### How to add a job to the cron?

```
# crontab –e
0 5 * * * /root/bin/backup.sh
```

This will execute /root/bin/backup.sh at 5 a.m every day.

### Description of Cron fields.

Following is the format of the crontab file.

{minute} {hour} {day-of-month} {month} {day-of-week} {full-path-to-shell-script}

- minute: Allowed range 0 – 59
- hour: Allowed range 0 – 23
- day-of-month: Allowed range 0 – 31
- month: Allowed range 1 – 12. 1 = January. 12 = December.
- Day-of-week: Allowed range 0 – 7. Sunday is either 0 or 7.

### Crontab examples

1. Run at 12:01 a.m. 1 minute after midnight everyday. This is a good time to run backup when the system is not under load.

```
1 0 * * * /root/bin/backup.sh
```

2. Run backup every weekday (Mon – Fri) at 11:59 p.m.

```
59 11 * * 1,2,3,4,5 /root/bin/backup.sh
```

Following will also do the same.

```
59 11 * * 1-5 /root/bin/backup.sh
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

3. Execute the command every 5 minutes.

```
*/5 * * * * /root/bin/check-status.sh
```

4. Execute at 1:10 p.m on 1st of every month

```
10 13 1 * * /root/bin/full-backup.sh
```

5. Execute 11 p.m on weekdays.

```
0 23 * * 1-5 /root/bin/incremental-backup.sh
```

### Crontab Options

Following are the available options with crontab:

- crontab -e : Edit the crontab file. This will create a crontab, if it doesn't exist
- crontab -l : Display the crontab file.
- crontab -r : Remove the crontab file.
- crontab -ir : This will prompt user before deleting a crontab.

## Hack 75. Safe Reboot Of Linux Using Magic SysRq Key

The magic SysRq key is a key combination in the Linux kernel which allows the user to perform various low level commands regardless of the system's state.

It is often used to recover from freezes, or to reboot a computer without corrupting the filesystem. The key combination consists of

Alt+SysRq+commandkey. In many systems the SysRq key is the printscreen key.

First, you need to enable the SysRq key, as shown below.

```
echo "1" > /proc/sys/kernel/sysrq
```

### List of SysRq Command Keys

Following are the command keys available for Alt+SysRq+commandkey.

- 'k' – Kills all the process running on the current virtual console.
- 's' – This will attempt to sync all the mounted file system.
- 'b' - Immediately reboot the system, without unmounting partitions or syncing.
- 'e' – Sends SIGTERM to all process except init.
- 'm' – Output current memory information to the console.
- 'i' - Send the SIGKILL signal to all processes except init
- 'r' - Switch the keyboard from raw mode (the mode used by programs such as X11), to XLATE mode.
- 's' – sync all mounted file system.
- 't' - Output a list of current tasks and their information to the console.
- 'u' - Remount all mounted filesystems in readonly mode.
- 'o' – Shutdown the system immediately.
- 'p' – Print the current registers and flags to the console.
- '0-9′ - Sets the console log level, controlling which kernel messages will be printed to your console.

- ‘f’ - Will call oom_kill to kill process which takes more memory.
- ‘h’ – Used to display the help. But any other keys than the above listed will print help.

We can also do this by echoing the keys to the /proc/sysrq-trigger file. For example, to re-boot a system you can perform the following.

```
echo "b" > /proc/sysrq-trigger
```

### Perform a Safe reboot of Linux using Magic SysRq Key

To perform a safe reboot of a Linux computer which hangs up, do the following. This will avoid the fsck during the next re-booting. i.e Press Alt+SysRq+letter highlighted below.

- un**R**aw (take control of keyboard back from X11,
- t**E**rminate (send SIGTERM to all processes, allowing them to terminate gracefully),
- **k**ill (send SIGILL to all processes, forcing them to terminate immediately),
- **S**ync (flush data to disk),
- **U**nmount (remount all filesystems read-only),
- re**B**oot.

# Chapter 10: Apachectl and Httpd Examples

After you have installed Apache2, if you want to use apachectl and httpd to it's maximum potential, you should go beyond using start, stop and restart. The 9 practical examples provided in this chapter will help you to use apachectl and httpd very effectively.

Apachectl acts as SysV init script, taking arguments like start, stop, restart and status. It also acts as front-end to httpd command, by simply passing the command line arguments to httpd. So, all the commands you execute using apachectl, can also be executed directly by calling httpd.

If you don't have Apache, refer to the tutorials: install apache from source or install LAMP stack using yum.

## Hack 76. Pass different httpd.conf filename to apachectl

Typically you'll modify the original httpd.conf to try out different Apache directives. If something doesn't work out, you'll revert back the changes. Instead of playing around with the original httpd.conf, copy it to a new httpd.conf.debug and use this new httpd.conf.debug file with Apache for testing purpose as shown below using option -f.

```
# apachectl -f conf/httpd.conf.debug

# httpd -k start -f conf/httpd.conf.debug

[Note: you can use either apachectl or httpd as shown
above]

# ps -ef | grep http
root   25080     1  0 23:26 00:00:00 /usr/sbin/httpd -f
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
conf/httpd.conf.debug
apache 25099 25080  0 23:28 00:00:00 /usr/sbin/httpd -f
conf/httpd.conf.debug

[Note: ps shows the httpd running with httpd.conf.debug
file]
```

Once you are satisfied with the changes and Apache runs without any problem with httpd.conf.debug, you can copy the changes to httpd.conf and start the Apache normally as shown below.

```
# cp httpd.conf.debug httpd.conf

# apachectl stop

# apachectl start

# ps -ef | grep httpd
root     25114     1  0 23:28 00:00:00 /usr/sbin/httpd
-k start
daemon   25115 25114  0 23:28 00:00:00 /usr/sbin/httpd
-k start

[Note: ps indicates that the httpd is running using the
default config file]
```

## Hack 77. Use a temporary DocumentRoot without modifying httpd.conf

This is very helpful, when you are trying out different layout for your website and don't want to modify the original files under the default DocumentRoot.

Take a copy of your original DocumentRoot directory (/var/www/html) to a new temporary DocumentRoot directory (/var/www/html_debug). Make all your changes under this temporary DocumentRoot directory (/var/www/html_debug) and start the Apache with this temporary directory as shown below using option -c.

```
# httpd -k start -c "DocumentRoot /var/www/html_debug/"
```

If you want to go back to original configuration using the default DocumentRoot (/var/www/html), simply restart the Apache as shown below.

```
# httpd -k stop
# apachectl start
```

## Hack 78. Increase the Log Level temporarily

While you are debugging an issue, you can change the LogLevel of the Apache temporarily, without modifying the LogLevel directive in the httpd.conf as shown below using option -e. In this example, the LogLevel is set to debug.

```
# httpd -k start -e debug

[Sun Aug 17 13:53:06 2008] [debug] mod_so.c(246):
loaded module auth_basic_module
[Sun Aug 17 13:53:06 2008] [debug] mod_so.c(246):
loaded module auth_digest_module
```

Possible values you can pass to option -e are:

- debug
- info
- notice
- warn
- error
- crit
- alert
- emerg

## Hack 79. Display the modules inside Apache

### Display the modules compiled inside Apache

```
# httpd -l

Compiled in modules:
core.c
prefork.c
http_core.c
mod_so.c
```

### Display both static and dynamic module loaded by Apache

When you pass option -l, to httpd, it will display only the static modules. Passing option -M, will display both static and shared modules as shown below.

```
# httpd –M

Loaded Modules:
core_module (static)
mpm_prefork_module (static)
http_module (static)
so_module (static)
auth_basic_module (shared)
auth_digest_module (shared)
authn_file_module (shared)
authn_alias_module (shared)
Syntax OK
```

## Hack 80. Show all accepted directives inside httpd.conf

This is like an extended help for httpd, which will display all the httpd.conf directives and the places where they are valid. For a specific directive, it tells all the possible values and where it can be used inside the httpd.conf. This can be very helpful, when you want to quickly know about a particular Apache directive.

```
# httpd -L

HostnameLookups (core.c)
“on” to enable, “off” to disable reverse DNS lookups,
or “double” to enable double-reverse DNS lookups
Allowed in *.conf anywhere

ServerLimit (prefork.c)
Maximum value of MaxClients for this run of Apache
Allowed in *.conf only outside <Directory>, <Files> or
<Location>

KeepAlive (http_core.c)
Whether persistent connections should be On or Off
Allowed in *.conf only outside <Directory>, <Files> or
<Location>

LoadModule (mod_so.c)
a module name and the name of a shared object file to
load it from
Allowed in *.conf only outside <Directory>, <Files> or
<Location>
```

## Hack 81. Validate the httpd.conf after making changes

Use option -t to validate whether there are any issues with a specific Apache configuration file. In the example shown below, it displays that there is a

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

problem at line 148 in the httpd.conf.debug. mod_auth_basicso is missing a . (period) before the so.

```
# httpd -t -f conf/httpd.conf.debug

httpd: Syntax error on line 148 of
/etc/httpd/conf/httpd.conf.debug:
Cannot load /etc/httpd/modules/mod_auth_basicso into
server:
/etc/httpd/modules/mod_auth_basicso: cannot open shared
object file: No such file or directory
```

Once you fix the issue, it will display Syntax OK.

```
# httpd -t -f conf/httpd.conf.debug

Syntax OK
```

## Hack 82. Display the httpd build parameters

Use option -V (upper-case V), to display Apache version number and all the parameters that are used while building the Apache.

```
# httpd –V

Server version: Apache/2.2.9 (Unix)
Server built:   Jul 14 2008 15:36:56
Server’s Module Magic Number: 20051115:15
Server loaded:  APR 1.2.12, APR-Util 1.2.12
Compiled using: APR 1.2.12, APR-Util 1.2.12
Architecture:   32-bit
Server MPM:     Prefork
threaded:     no
forked:     yes (variable process count)
Server compiled with….
-D APACHE_MPM_DIR=”server/mpm/prefork”
-D APR_HAS_SENDFILE
-D APR_HAS_MMAP
```

```
-D APR_HAVE_IPV6 (IPv4-mapped addresses enabled)
-D APR_USE_SYSVSEM_SERIALIZE
-D APR_USE_PTHREAD_SERIALIZE
-D SINGLE_LISTEN_UNSERIALIZED_ACCEPT
-D APR_HAS_OTHER_CHILD
-D AP_HAVE_RELIABLE_PIPED_LOGS
-D DYNAMIC_MODULE_LIMIT=128
-D HTTPD_ROOT="/etc/httpd"
-D SUEXEC_BIN="/usr/sbin/suexec"
-D DEFAULT_PIDLOG="logs/httpd.pid"
-D DEFAULT_SCOREBOARD="logs/apache_runtime_status"
-D DEFAULT_LOCKFILE="logs/accept.lock"
-D DEFAULT_ERRORLOG="logs/error_log"
-D AP_TYPES_CONFIG_FILE="conf/mime.types"
-D SERVER_CONFIG_FILE="conf/httpd.conf"
```

If you want display only the Apache version number, use the option -v (lower-case v) as shown below.

```
# httpd -v

Server version: Apache/2.2.9 (Unix)
Server built:   Jul 14 2008 15:36:56
```

## Hack 83. Load a specific module only on demand

Sometimes you may not want to load all the modules in the Apache. For e.g. You may want to load ldap related modules to Apache, only when you are testing LDAP. This can be achieved as shown below.

Modify the httpd.conf and add IfDefine directive called load-ldap (you can name this anything you want).

```
<IfDefine load-ldap>
LoadModule ldap_module modules/mod_ldap.so
LoadModule authnz_ldap_module
```

```
modules/mod_authnz_ldap.so
</IfDefine>
```

When you are testing ldap and would like to Load the ldap related modules, pass the load-ldap to Option -D, as shown below:

```
# httpd -k start -e debug -Dload-ldap -f
/etc/httpd/conf/httpd.conf.debug

[Sun Aug 17 14:14:58 2008] [debug] mod_so.c(246):
loaded module ldap_module
[Sun Aug 17 14:14:58 2008] [debug] mod_so.c(246):
loaded module authnz_ldap_module
[Note: Pass -Dload-ldap, to load the ldap modules into
Apache]

# apachectl start

[Note: Start the Apache normally, if you don't want to
load the ldap modules.
```

# Chapter 11: Bash Scripting

## Hack 84. Execution Sequence of .bash_* files

What is the sequence in which the following files are executed?

- /etc/profile
- ~/.bash_profile
- ~/.bashrc
- ~/.bash_login
- ~/.profile
- ~/.bash_logout

### Execution sequence for interactive login shell

Following pseudo code explains the sequence of execution of these files.

```
execute /etc/profile
IF ~/.bash_profile exists THEN
    execute ~/.bash_profile
ELSE
    IF ~/.bash_login exist THEN
        execute ~/.bash_login
    ELSE
        IF ~/.profile exist THEN
            execute ~/.profile
        END IF
    END IF
END IF
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

When you logout of the interactive shell, following is the sequence of execution:

```
IF ~/.bash_logout exists THEN
    execute ~/.bash_logout
END IF
```

Please note that /etc/bashrc is executed by ~/.bashrc as shown below:

```
# cat ~/.bashrc
if [ -f /etc/bashrc ]; then
. /etc/bashrc
Fi
```

### Execution sequence for interactive non-login shell

While launching a non-login interactive shell, following is the sequence of execution:

```
IF ~/.bashrc exists THEN
    execute ~/.bashrc
END IF
```

**Note:** When a non-interactive shell starts up, it looks for ENV environment variable, and executes the file-name value mentioned in the ENV variable.

## Hack 85. How to generate random number in bash shell

Use the $RANDOM bash built-in function to generate random number between 0 - 32767 as shown below.

```
$ echo $RANDOM
```

```
22543

$ echo $RANDOM
25387

$ echo $RANDOM
647
```

## Hack 86. Debug a shell script

To debug a shell script use set -xv inside the shell script at the top.

**Shell script with no debug command:**

```
$ cat filesize.sh
#!/bin/bash
for filesize in $(ls -l . | grep "^-" | awk '{print
$5}')
do
  let totalsize=$totalsize+$filesize
done
echo "Total file size in current directory: $totalsize"
```

**Output of Shell script with no debug command:**

```
$ ./filesize.sh
Total file size in current directory: 652
```

**Shell script with Debug command inside:**

Add set -xv inside the shell script now to debug the output as shown below.

```
$ cat filesize.sh
#!/bin/bash
set -xv
for filesize in $(ls -l . | grep "^-" | awk '{print
```

```
$5}')
do
  let totalsize=$totalsize+$filesize
done
echo "Total file size in current directory: $totalsize"
```

**Output of Shell script with Debug command inside:**

```
$ ./fs.sh
++ ls -l .
++ grep '^-'
++ awk '{print $5}'
+ for filesize in '$(ls -l . | grep "^-" | awk
'\''{print $5}'\'')'
+ let totalsize=+178
+ for filesize in '$(ls -l . | grep "^-" | awk
'\''{print $5}'\'')'
+ let totalsize=178+285
+ for filesize in '$(ls -l . | grep "^-" | awk
'\''{print $5}'\'')'
+ let totalsize=463+189
+ echo 'Total file size in current directory: 652'
Total file size in current directory: 652
```

**Execute Shell script with debug option:**

Instead of giving the set -xv inside the shell script, you can also provide that while executing the shell script as shown below.

```
$ bash -xv filesize.sh
```

## Hack 87. Quoting

echo statement without any special character.

```
$ echo The Geek Stuff
```

```
The Geek Stuff
```

Echo statement with a special character ; . semi-colon is a command terminator in bash. In the following example, “The Geek” works for the echo and “Stuff” is treated as a separate Linux command and gives command not found.

```
$ echo The Geek; Stuff
The Geek
-bash: Stuff: command not found
```

To avoid this you can add a \ in front of semi-colon, which will remove the special meaning of semi-colon and just print it as shown below.

```
$ echo The Geek\; Stuff
The Geek; Stuff
```

### Single Quote

Use single quote when you want to literally print everything inside the single quote. Even the special variables such as $HOSTNAME will be print as $HOSTNAME instead of printing the name of the Linux host.

```
$ echo 'Hostname=$HOSTNAME ;  Current User=`whoami` ;
Message=\$ is USD'

Hostname=$HOSTNAME ;  Current User=`whoami` ;
Message=\$ is USD
```

### Double Quote

Use double quotes when you want to display the real meaning of special variables.

```
$ echo "Hostname=$HOSTNAME ;  Current User=`whoami` ;
Message=\$ is USD"
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
Hostname=dev-db ;  Current User=ramesh ; Message=$ is
USD
```

Double quotes will remove the special meaning of all characters except the following:

- $ Parameter Substitution.
- ` Backquotes
- \$ Literal Dollar Sign.
- \´ Literal Backquote.
- \" Embedded Doublequote.
- \\ Embedded Backslashes.

## Hack 88. Read data file fields inside a shell script

This example shows how to read a particular field from a data-file and manipulate it inside a shell-script. For example, let us assume the employees.txt file is in the format of {employee-name}:{employee-id}:{department-name}, with colon delimited file as shown below.

```
$ cat employees.txt
Emma Thomas:100:Marketing
Alex Jason:200:Sales
Madison Randy:300:Product Development
Sanjay Gupta:400:Support
Nisha Singh:500:Sales
```

The following shell script explains how to read specific fields from this employee.txt file.

```
$ vi read-employees.sh
#!/bin/bash
IFS=:
echo "Employee Names:"
echo "---------------"
while read name empid dept
```

```
do
  echo "$name is part of $dept department"
done < ~/employees.txt
```

Assign execute privilege to the shell script and execute it.

```
$ chmod u+x read-employees.sh

$ ./read-employees.sh
Employee Names:
---------------
Emma Thomas is part of Marketing department
Alex Jason is part of Sales department
Madison Randy is part of Product Development department
Sanjay Gupta is part of Support department
Nisha Singh is part of Sales department
```

# Chapter 12: System Monitoring and Performance

## Hack 89. Free command

free command displays all the necessary information about system physical (RAM) and swap memory.

```
Syntax: free [options]
```

**What is the total RAM on my system?**

In the example below, the total physical memory on this system is 1GB. The values displayed below are in KB.

```
# free
       total   used    free   shared  buffers  cached
Mem: 1034624   1006696 27928  0       174136   615892
-/+ buffers/cache:     216668      817956
Swap:    2031608       0    2031608
```

**What is the total memory on my system including RAM and Swap?**

In the following command:

- option m displays the values in MB
- option t displays the “Total” line, which is sum of physical and swap memory values
- option o is to hide the buffers/cache line from the above example.

# **free -mto**

```
             total       used       free     shared    buffers     cached
Mem:          1010        983         27          0        170        601
Swap:         1983          0       1983
Total:        2994        983       2011
```

## Hack 90. Top Command

top command displays real time information about various performance metrics of the system such as CPU Load, Memory Usage, Processes list etc.

```
Syntax: top [options]
```

### How to view my current system status including CPU usage?

Execute top without any option from the command line, which will display the output shown below. The top command output will keep displaying the real-time values, until you press “Control + c” or q to exit from the command output.

# **top**

```
top - 13:10:13 up 171 days, 20:21,  3 users,  load average: 0.01, 0.05, 0.00

Tasks: 194 total,   1 running, 193 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.6% us,  0.7% sy,  0.0% ni, 98.7% id,  0.0% wa,  0.0% hi,  0.0% si
Mem:   1034624k total,  1007420k used,    27204k free,   174540k buffers
Swap:  2031608k total,        0k used,  2031608k free,   615904k cached

PID  USER    PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
11912 apache  15   0 31828  13m  3916 S    1    0.2   0:46.35  httpd
19299  oracle  19   0  279m  18m   17m S    1    0.2   0:00.03  oracle
11398 jsmith  16   0  107m  28m  6404 S    0    0.4   0:03.07  perl
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

### How to read the output of the top command shown above?

- Line 1 “top”, indicates that the system has been up and running for 171 days.
- Line 2 “Tasks”, displays the total number of processes along with a breakdown of running, sleeping, stopped and zombie processes count.
- Line 3 “Cpu(s)” displays the current CPU utilization of the system. In this example, CPU is 98.7% idle
- Line 4 “Mem” and line 5 “Swap” provides the memory information. This is the same information from the free command.
- The rest of the lines display all the active processes on the system, sorted default by CPU usage (%CPU column). i.e the most CPU intensive processes will be displayed on the top by default.

There are several command line options and interactive options available for top commands. Let us review couple of essential options for top command.

### How to identify the most memory intensive processes?

While the output of the top command displayed, press F, which will display the following message and show all fields available for sorting, press n (which is for sorting the processes by Memory) and press enter. This will display the processes in the top output sorted by memory usage.

```
Current Sort Field:  K  for window 1:Def

Select sort field via field letter, type any other key to return
```

### How to add additional fields (for e.g. CPU Time) to the top output?

While the top command is running, press f, which will display the following message and show all fields available for display, press l, which will add the CPU Time to the display columns in the top output.

```
Current Fields:  AEHIOQTWKNMbcdfgjplrsuvyzX  for window 1:Def

Toggle fields via field letter, type any other key to return
```

### How to get the full path name and parameters of the running processes?

While the top command is running, press c, which will display full pathname of running processes as shown below in the command column. i.e Instead of httpd, it displays /usr/local/apache2/bin/httpd.

```
PID   USER    PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND

11912 apache  15   0 31828  13m  3916 S    1    0.2   0:46.35
/usr/local/apache2/bin/httpd
```

### How to view the individual CPUs in the top command?

While the top command is running, press 1 (number one), which will display the performance data of the individual CPUs on that machine as shown below.

```
top - 13:10:13 up 171 days, 20:21,  3 users,  load average: 0.01, 0.05, 0.00
Tasks: 194 total,   1 running, 193 sleeping,   0 stopped,   0 zombie
Cpu0  : 10.2% us,  2.6% sy,  0.0% ni, 86.8% id,  0.3% wa,  0.0% hi,  0.0% si
Cpu1  :  9.6% us,  8.0% sy,  0.0% ni, 82.4% id,  0.0% wa,  0.0% hi,  0.0% si
Cpu2  :  1.3% us,  1.3% sy,  0.0% ni, 95.0% id,  2.3% wa,  0.0% hi,  0.0% si
Cpu3  :  0.0% us,  0.0% sy,  0.0% ni, 100.0% id,  0.0% wa,  0.0% hi,  0.0% si
```

```
Mem:   1034624k total,  1007420k used,     27204k free,   174540k buffers
Swap:  2031608k total,        0k used,  2031608k free,   615904k cached
```

## Hack 91. Ps Command

ps command (process status) will display snapshot information of all active processes.

```
Syntax: ps [options]
```

### How to display all the processes running in the system?

Use "ps aux", as shown below.

```
# ps aux | more

USER       PID %CPU %MEM    VSZ    RSS TTY  STAT START  TIME COMMAND
root         1   0.0   0.0   2044    588 ?       Ss       Jun27
0:00  init [5]
apache   31186      0.0       1.6  23736 17556 ?       S        Jul26
0:40  /usr/local/apache2/bin/httpd
apache   31187      0.0       1.3  20640 14444 ?       S        Jul26
0:37  /usr/local/apache2/bin/httpd
```

You can also use "ps -ef | more", to get a similar output

### Print the Process Tree

You can use either **ps axuf** or **ps –ejH** to display processes in a tree format. The tree structure will help to visualize the process and it's parent process immediately. For clarity purpose, few columns have been cut-off in the output below.

```
# ps axuf
```

```
root     Oct14   0:00 /opt/VRTSralus/bin/beremote
root     Oct14   0:00  \_ /opt/VRTSralus/bin/beremote
root     Oct14   0:00      \_ /opt/VRTSralus/bin/beremote
root     Oct14   0:00      \_ /opt/VRTSralus/bin/beremote
root     Oct14   0:01      \_ /opt/VRTSralus/bin/beremote
root     Oct 14  0:00      \_ /opt/VRTSralus/bin/beremote
root     Dec03   0:01 /usr/local/sbin/sshd
root     Dec22   1:08 /usr/local/sbin/sshd
root     23:35   0:00  \_ /usr/local/sbin/sshd
511      23:35   0:00      \_ -bash
511                           \_ ps axuf
```

Note: You can also use pstree command to display process in tree structure.

### View Processes Owned by a Particular User

The following command displays all the process owned by Linux user-name: oracle.

```
$ ps U oracle

PID TTY      STAT   TIME COMMAND
5014 ?        Ss     0:01 /oracle/bin/tnslsnr
7124 ?        Ss     0:00 ora_q002_med
8206 ?        Ss     0:00 ora_cjq0_med
8852 ?        Ss     0:01 ora_pmon_med
8854 ?        Ss     0:00 ora_psp0_med
8911 ?        Ss     0:02 oraclemed (LOCAL=NO)
```

### View Processes Owned by Current User

Following command displays all the process owned by the current user.

```
$ ps U $USER

PID TTY      STAT   TIME COMMAND
10329 ?        S      0:00 sshd: ramesh@pts/1,pts/2
10330 pts/1    Ss     0:00 -bash
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
10354 pts/2    Ss+    0:00 -bash
10530 pts/1    R+     0:00 ps U ramesh
```

## Hack 92. Df Command

df command (disk free) displays the amount of total and free disk space available on the mounted filesystems.

```
Syntax: df [options] [name]
```

### How much GB of disk space is free on my system?

Use df -h as shown below. Option -h displays the values in human readable format (for example: K for Kb, M for Mb and G for Gb). In the sample output below, / filesystem has 17GB of disk space available and /home/user filesystem has 70GB available.

```
# df –h

Filesystem            Size  Used Avail Use% Mounted on
/dev/sda1              64G   44G   17G  73%   /
/dev/sdb1             137G   67G   70G  49%   /home/user
```

### What type of filesystem do I have on my system?

Option -T will display the information about the filesystem Type. In this example / and /home/user filesystems are ext2. Option -a will display all the filesystems, including the 0 size special filesystem used by the system.

```
# df -Tha

Filesystem    Type    Size  Used  Avail Use%  Mounted on
```

```
/dev/sda1      ext2    64G   44G   17G  73%   /
/dev/sdb1      ext2   137G   67G   70G  49%   /home/user
none           proc      0     0     0    -      /proc
none           sysfs     0     0     0    -      /sys
none           devpts    0     0     0    -      /dev/pts
none           tmpfs  2.0G     0  2.0G   0%   /dev/shm
```

## Hack 93. Kill Command

kill command can be used to terminate a running process. Typically this command is used to kill processes that are hanging and not responding.

```
Syntax: kill [options] [pids|commands]
```

**How to kill a hanging process?**

First, identify the process id of the particular process that you would like to kill using the ps command. Once you know the process id, pass it as a parameter to the kill command. The example below shows how to kill the hanging apache httpd process. Please note that typically you should use “apachectl stop” to stop apache.

```
# ps aux | grep httpd
USER       PID %CPU %MEM    VSZ   RSS TTY  STAT START  TIME COMMAND
apache   31186  0.0  1.6  23736 17556 ?    S    Jul26
0:40  /usr/local/apache2/bin/httpd
apache   31187  0.0  1.3  20640 14444 ?    S    Jul26
0:37  /usr/local/apache2/bin/httpd

# kill 31186 31187
```

Please note that the above command tries to terminate the process graciously by sending a signal called SIGTERM. If the process does not get terminated, you can forcefully terminate the process by passing a signal called SIGKILL, using the option -9 as shown below. You should either be the owner of the process or a privileged user to kill a process.

```
# kill -9 31186 31187
```

Another way to kill multiple processes easily is by adding the following two functions to the .bash_profile.

```
function psgrep ()
{
 ps aux | grep "$1" | grep -v 'grep'
}

function psterm ()
{
 [ ${#} -eq 0 ] && echo "usage: $FUNCNAME STRING" && return 0
 local pid
 pid=$(ps ax | grep "$1" | grep -v grep | awk '{ print $1 }')
 echo -e "terminating '$1' / process(es):\n$pid"
 kill -SIGTERM $pid
}
```

Now do the following, to identify and kill all httpd processes.

```
# psgrep http

USER       PID %CPU %MEM    VSZ    RSS TTY  STAT START  TIME
COMMAND
apache   31186       0.0        1.6  23736 17556 ?        S
Jul26       0:40  /usr/local/apache2/bin/httpd
apache   31187       0.0        1.3  20640 14444 ?        S
Jul26       0:37  /usr/local/apache2/bin/httpd

# psterm httpd

terminating 'httpd' / process(es):
31186
```

```
31187
```

## Hack 94. Du Command

du command (disk usage) will print the file space usage for a particular directory and its subdirectories.

### How much space is taken by my home directory and all its subdirectories?

In the following example, option -s stands for summary only. i.e it displays only the total size of /home/jsmith and not the individual sizes of all the subdirectories inside the /home/jsmith. Option -h displays the information in a human readable format. i.e K for KB, M for MB and G for GB. The ~ indicates the user home directory. This command is same as “du -sh /home/jsmith”

```
# du -sh ~
320M    /home/jsmith
```

To get the subdirectories under /home/jsmith listed, execute the above command without the s option.

## Hack 95. lsof commands.

Lsof stands for ls open files, which will list all the open files in the system. The open files include network connection, devices and directories. The output of the lsof command will have the following columns:

- COMMAND process name.
- PID process ID
- USER Username

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

- FD file descriptor
- TYPE node type of the file
- DEVICE device number
- SIZE file size
- NODE node number
- NAME full path of the file name.

### View all open files of the system

Execute the lsof command without any parameter as shown below.

```
# lsof | more
COMMAND     PID      USER   FD      TYPE     DEVICE     SIZE      NODE
NAME
init          1      root  cwd       DIR       8,1     4096        2 /
init          1      root  rtd       DIR       8,1     4096        2 /
init          1      root  txt       REG       8,1    32684   983101 /sbin/init
init          1      root  mem       REG       8,1   106397   166798 /lib/ld-
2.3.4.so
init          1      root  mem       REG       8,1  1454802   166799
/lib/tls/libc-2.3.4.so
init          1      root  mem       REG       8,1    53736   163964
/lib/libsepol.so.1
init          1      root  mem       REG       8,1    56328   166811
/lib/libselinux.so.1
init          1      root   10u      FIFO      0,13              972 /dev/initctl
migration     2      root  cwd       DIR       8,1     4096        2 /
skipped...
```

The lsof command by itself without may return lot of records as output, which may not be very meaningful except to give you a rough idea about how

many files are open in the system at any given point of view as shown below.

```
# lsof | wc -l

3093
```

### View open files by a specific user

Use lsof –u option to display all the files opened by a specific user.

```
# lsof –u ramesh

vi      7190 ramesh  txt    REG        8,1   474608
475196 /bin/vi

sshd    7163 ramesh    3u  IPv6   15088263
TCP dev-db:ssh->abc-12-12-12-12.socal.res.rr.com:2631
(ESTABLISHED)
```

A system administrator can use this command to get some idea on what users are executing on the system.

### List Users of a particular file

If you like to view all the users who are using a particular file, use lsof as shown below. In this example, it displays all users who are currently using vi.

```
# lsof /bin/vi

COMMAND  PID  USER    FD   TYPE DEVICE   SIZE   NODE NAME
vi      7258  root   txt    REG    8,1 474608 475196 /bin/vi
vi      7300  ramesh txt    REG    8,1 474608 475196 /bin/vi
```

## Hack 96. Sar Command

Sar commands comes with the sysstat package. Make sure sysstat is installed. If you don't have sar installed on your system, get it from Sysstat project.

Sar is an excellent monitoring tool that displays performance data of pretty much every resource of the system including CPU, memory, IO, paging, networking, interrupts etc.,

Sar Collects, Reports (displays) and Saves the performance data. Let us look at all the three aspects separately

### Sadc - System activity data collector

/usr/lib/sadc (System activity data collector) command collects the system data at a specified time interval. This uses the daily activity data file that is located under /va/log/sa/sa[dd], where dd is the current day.

### Sa1 shell-script

/usr/lib/sa1 in-turn calls the /usr/lib/sadcs. sa1 is invoked from the crontab as shown below. Run this every 5 minutes or 15 minutes depending on your need. I prefer to schedule it for every 5 minutes in the cron tab as shown below.

```
*/5 * * * * root /usr/lib/sa/sa1 1 1
```

### Sa2 shell-script

/usr/lib/sa2 is a shell script that will write a daily report in the /var/log/sa/sa[dd] file, where dd is the current day. Invoke the sa2 from the crontab once a day at midnight.

```
# 59 23 * * * root /usr/lib/sa/sa2 –A
```

**Note:** /etc/cron.d/sysstat files comes with the sysstat package that includes some default value for the sa1 and sa2, which you can change accordingly.

### Display CPU Statistics using Sar Command

```
# sar –u

Linux 2.6.9-42.ELsmp (dev-db)        01/01/2009
12:00:01 AM  CPU  %user  %nice   %system   %iowait     %idle
12:05:01 AM  all  3.70    0.00      0.85      0.00     95.45
12:10:01 AM  all  4.59    0.00      1.19      0.06     94.16
12:15:01 AM  all  3.90    0.00      0.95      0.04     95.11
12:20:01 AM  all  4.06    0.00      1.00      0.01     94.93
12:25:01 AM  all  3.89    0.00      0.87      0.00     95.23
12:30:01 AM  all  3.89    0.00      0.87      0.00     95.23

Skipped..

Average: all     4.56     0.00      1.00      0.15     94.29
```

**Note:** If you need a break down of the performance data for the individual CPU's, execute the following command.

```
# sar -u -P ALL
```

### Display Disk IO Statistics using sar command

```
# sar –d

Linux 2.6.9-42.ELsmp (dev-db)        01/01/2009
12:00:01 AM    DEV              tps    rd_sec/s  wr_sec/s
12:05:01 AM    dev2-0           1.65      1.28     45.43
12:10:01 AM    dev8-1          4.08      8.11     21.81

Skipped..

Average:       dev2-0           4.66    120.77     69.45
Average:       dev8-1          1.89      3.17      8.02
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

### Display networking Statistics using sar command

```
# sar -n DEV | more

Linux 2.6.9-42.ELsmp (dev-db)        01/01/2009
12:00:01 AM     IFACE   rxpck/s   txpck/s   rxbyt/s   txbyt/s
rxcmp/s   txcmp/
s  rxmcst/s
12:05:01 AM        lo      0.17      0.16     25.31     23.33
0.00      0.0
0      0.00
12:10:01 AM      eth0     52.92     53.64  10169.74  12178.57
0.00      0.0
0      0.00


# sar -n SOCK |more

Linux 2.6.9-42.ELsmp (dev-db)        01/01/2009
12:00:01 AM    totsck    tcpsck    udpsck    rawsck   ip-frag
12:05:01 AM        50        13         3         0         0
12:10:01 AM        50        13         4         0         0
12:15:01 AM        53        13         5         0         0
```

## Hack 97. vmstat Command

For a typical performance monitoring all you need is only vmstat command. This display memory, swap, IO, system and cpu performance information.

The following command executes vmstat every 1 second for 100 times.

```
# vmstat 1 100
procs -----------memory---------- ---swap-- -----io---- --system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us sy id wa
 0  0      0 282120 134108 5797012    0    0     0     2    0     0  0  0 100  0
 0  0      0 282120 134108 5797012    0    0     0     0 1007   359  0  0 100  0
```

```
0  0      0 282120 134108 5797012    0    0     0     0 1117   577  0  0 100  0
0  0      0 282120 134108 5797012    0    0     0     0 1007   366  0  0 100  0
```

### Vmstat procs Section

- r field: Total number of runnable process
- b field: Total number of blocked process

### Memory section

- Swpd field: Used swap space
- Free field: Available free RAM
- Buff field: RAM used for buffers
- Cache field: RAM used for filesystem cache

### Swap Section

- Si field: Amount of memory swapped from disk per second
- So field: Amount of memory swapped to disk per second

### IO Section

- Bi field: Blocks received from disk
- Bo field: Blocks sent to disk.

### System Section

- In field: Number of interrupts per second.
- Cs field: Number of context switches per second.

### CPU Section

- Us field: Time spend running user code. (non-kernel code)
- Sy field: Time spent running kernel code.
- Id field: Idle time.
- Wa field: Time spent waiting for the IO

## Hack 98. Netstat Command

Netstat command displays the network related information such as network connections, routing tables, interface statistics. Following are few examples on how to use netstat command.

### Display Active Internet Connections and domain sockets using netstat

```
# netstat –an

Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign
Address             State
tcp        0      0 0.0.0.0:5666                0.0.0.0:*
LISTEN
tcp        0      0 0.0.0.0:111                 0.0.0.0:*
LISTEN
tcp        0      0 0.0.0.0:4086                0.0.0.0:*
LISTEN
skipped..
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node Path
unix  2      [ ACC ]     STREAM     LISTENING     7894
/tmp/.font-unix/fs7100
unix  2      [ ACC ]     STREAM     LISTENING     9662
/tmp/.gdm_socket
unix  2      [ ACC ]     STREAM     LISTENING     10897
@/tmp/fam-root-
```

### Display Active Connections with Process ID and Program Name

This could be very helpful to identify which program has initiated a specific network connection.

```
# netstat -tap

Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign
Address             State       PID/Program name
tcp        0      0 *:nrpe                      *:*
LISTEN      16277/xinetd
tcp        0      0 localhost.localdomain:smtp  *:*
LISTEN      7263/sendmail: acce
tcp       34      0 localhost.localdomain:54221
localhost.localdomain:4089  CLOSE_WAIT  29881/httpd
tcp        0   3216 dev-db:ssh                cpe-76-
94-215-154.soca:4682 ESTABLISHED 11717/sshd: ramesh
```

### Display Routing Table

```
# netstat --route

Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS
Window  irtt Iface
192.168.1.0   *               255.255.255.0   U         0 0
0 eth0
162.244.0.0     *               255.255.0.0     U         0 0
0 eth0
default         192.168.1.1   0.0.0.0         UG        0 0
0 eth0
```

### Display RAW network statistics

```
# netstat --statistics --raw

Ip:
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
    11080343 total packets received
    0 forwarded
    1 with unknown protocol
    0 incoming packets discarded
    11037744 incoming packets delivered
    11199763 requests sent out
Icmp:
    577135 ICMP messages received
    64 input ICMP message failed.
    ICMP input histogram:
        destination unreachable: 537
        timeout in transit: 65
        source quenches: 2
        echo requests: 576476
        echo replies: 12
        timestamp request: 3
        address mask request: 3
    581558 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 5079
        echo replies: 576476
        timestamp replies: 3
```

### Misc Netstat Commands

- **# netstat --tcp --numeric** List of TCP connection to and from the machine.
- **# netstat --tcp --listening --programs** Display TCP port that the server is listening on along with the program that is listening on that particular port.
- **# netstat -rnC** Display the routing cache

## Hack 99. Sysctl Command

Linux kernel parameter can be changed on the fly using sysctl command.

Sysctl helps to configure the Linux kernel parameters during runtime.

```
# sysctl –a

dev.cdrom.autoclose = 1
fs.quota.writes = 0
kernel.ctrl-alt-del = 0
kernel.domainname = (none)
kernel.exec-shield = 1
net.core.somaxconn = 128
net.ipv4.tcp_window_scaling = 1
net.ipv4.tcp_wmem = 4096        16384   131072
net.ipv6.route.mtu_expires = 600
sunrpc.udp_slot_table_entries = 16
vm.block_dump = 0
```

### Modify Kernel parameter in /etc/sysctl.conf for permanent change

After modifying the kernel parameter in the /etc/sysctl.conf, execute sysctl -p to commit the changes. The changes will still be there after the reboot.

```
# vi /etc/sysctl.conf

# sysctl –p
```

### Modify kernel parameter temporarily

To temporarily modify a kernel parameter, execute the following command. Please note that after reboot these changes will be lost.

```
# sysctl –w {variable-name=value}
```

## Hack 100. Nice Command

Kernel decides how much processor time is required for a process based on the nice value. Possible nice value range is: -20 to 20. A process that has a nice value of -20 is very high priority. The process that has a nice value of 20 is very low priority.

Use ps axl to display the nice value of all running process as shown below.

```
# ps axl

F   UID   PID  PPID PRI  NI   VSZ  RSS WCHAN  STAT TTY
TIME COMMAND
4     0     1     0  16   0  2172  552 -      S    ?
0:17 init [5]
1     0     3     1  34  19     0    0 ksofti SN   ?
3:18 [ksoftirqd/0]
1     0    10     1   5 -10     0    0 worker S<   ?
0:01 [events/0]
4     0  5145     1  25  10 32124 18592 -      SNs  ?
0:08 /usr/bin/python /usr/bin/rhn-applet-gui --sm-client-id
default4
4     0  5147  5142  16   0  3528  604 -      S    ?
0:00 /sbin/pam_timestamp_check -d root
1   503 17552  4180  16   0 14208 3920 -      S    ?
0:01 /home/www/apache2/bin/httpd -f
/home/www/apache2/conf/httpd.conf -k start
```

### How to assign a low priority to a shell-script? (higher nice value)

In the example below, when I started the nice-test.sh script in the background, it took the nice value of 0.

```
$ ./nice-test.sh &
[3] 13009

$ ps axl | grep nice-test
0   509 13009 12863  17   0  4652  972 wait   S
```

```
pts/1      0:00 /bin/bash ./nice-test.sh

[Note: 6th column with value 0 is the nice.]
```

Now, let us execute the same shell script with a different nice value as shown below.

```
$ nice -10 ./nice-test.sh &
[1] 13016

$ ps axl | grep nice-test
0   509 13016 12863  30  10  4236  968 wait   SN
pts/1      0:00 /bin/bash ./nice-test.sh

[Note: 6th column with value 10 is the nice value for
the shell-script.]
```

### How to assign a high priority to a shell-script? (Lower nice value)

In the following example, let us assign a nice value of -10 (minus 10) to the nice-test.sh shellscript.

```
$ nice --10 ./nice-test.sh &
[1] 13021
$ nice: cannot set priority: Permission denied
```

**Note:** Only root user can set a negative nice value. Login as root and try the same. Please note that there is a double dash before the 10 in the nice command below.

```
# nice --10 ./nice-test.sh &
[1] 13060

# ps axl | grep nice-test
4     0 13060 13024  10 -10  5388  964 wait   S<
pts/1      0:00 /bin/bash ./nice-test.sh
```

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

```
[Note: 6th column with value -10 is the nice value of
the shell-script.]
```

## Hack 101. Renice Command

Renice alters the scheduling priority of a running process.

### How to decrease the priority of a running process? (Increase nice)

In the example below, an existing shell-script is running at nice value of 10. (6th column in the ps output)

```
$ ps axl | grep nice-test
0   509 13245 13216  30  10  5244  968 wait   SN
pts/1      0:00 /bin/bash ./nice-test.sh
```

To increase the nice value (thus reducing the priority), execute the renice command as shown below.

```
$ renice 16 -p 13245
13245: old priority 10, new priority 16

$ ps axl | grep nice-test
0   509 13245 13216  36  16  5244  968 wait   SN
pts/1      0:00 /bin/bash ./nice-test.sh

[Note: Now, the 6th column of the nice-test.sh (PID
13245) shows the new nice value of 16.]
```

### How to increase the priority of a running process? (Decrease nice)

In the example below, an existing shell-script is running at a nice value of 10.

(6th column in the ps output)

```
$ ps axl | grep nice-test
0   509 13254 13216  30  10  4412  968 wait   SN
pts/1      0:00 /bin/bash ./nice-test.sh
```

In increase the priority, give a lower nice value as shown below. However, only root can increase the priority of a running process, else you'll get the following error message.

```
$ renice 5 -p 13254
renice: 13254: setpriority: Permission denied
Login as root to increase the priority of a running
process

$ su -

# renice 5 -p 13254
13254: old priority 10, new priority 5

# ps axl | grep nice-test
0   509 13254 13216  25   5  4412  968 wait   SN
pts/1      0:00 /bin/bash ./nice-test.sh

[Note: The 6th column now shows a lower nice value of 5
(increased priority)]
```

# 12 Amazing and Essential Linux Books

For further reading on Linux, I recommend the following books. The 12 Linux books mentioned here by no means are comprehensive or authoritative list. But, these 12 Books are few of my favorites that I enjoyed reading over the years and I strongly believe will enhance your technical abilities on Linux, if you have not read them yet.

1. **Sed and Awk, by Dale Dougherty and Arnold Robbins**. Sed and Awk have transformed the way I worked on Linux command line. This book is the only material you would ever need on Sed and Awk. Once you've mastered even the basics of Sed and Awk, you'll be amazed with the amount of complex tasks you can perform very quickly and elegently. For my day-to-day quick reference of sed and awk examples, I use the Sed and Awk Pocket Reference, written by the same author.

2. **Learning the Vi and Vim Editors, by Arnold Robbins**. I'm a command-line junkie. So, naturally I'm a huge fan of Vi and Vim editors. Several years back, when I wrote lot of C code on Linux, I used to carry the Vi editor pocket reference with me all the times. Even if you've been using Vi and Vim Editors for several years and have not read this book, please do yourself a favor and read this book. You'll be amazed with the capabilities of Vim editor.

3. **Bash Cookbook, by Carl Albing, JP Vossen and Cameron Newham**. Whether you are a sysadmin, DBA or a developer, you have to write shell script at some point. A wise sysadmin knows that once you've mastered the shell-scripting techniques, you can put your servers on auto-pilot mode by letting the shell-scripts do the grunt work. To get to the auto-pilot mode of sysadmin, you definitely need to master the examples provided in this cookbook. There are quiet few Bash shell books out there. But, this books tops them all by giving lot of detailed examples.

4. **SSH, The Secure Shell, by Daniel J. Barrett, Richard E. Silverman and Robert G. Byrnes**. This is hands-down the best book on SSH. This book explains both theoretical and practical aspects of SSH. Using

SSH as an end-user is fairly straight forward . But, configuring SSH as an administrator is complex and involves a detailed understanding of SSH. This is a must read for any system administrator. The examples in this book show exactly what needs to be done differently for the different flavors of SSH such as SSH1, SSH2 and OpenSSH.

5. **Essential System Administration, by Æleen Frisch.** This is an excellent book for those who like to become a Unix System Administrator. This book covers all the typical system administration tasks. This is a perfect companion when you are dealing with multiple flavors of Unix, as it has examples for AIX, FreeBSD, HP-UX, Linux, Solaris and Tru64. I've used the pocket version of this book — Essential System Administration Pocket Reference, when I was managing multiple flavors of Unix systems at the same time.

6. **Linux Server Hacks, Volume One, by Rob Flickenger**. 100 awesome practical hacks packed in one book. Setup a Linux test bed and try out all these hacks. These hacks are neatly grouped into different sections — Server Basics, Revision Control, Backups, Networking, Monitoring, SSH, Scripting, and Information Servers. Once you've mastered these hacks, you should absolutely read Linux Server Hacks, Volume Two, by William von Hagen and Brian Jones, which has 100 Linux hacks focussed on authentication, monitoring, security, performance and connectivity.

7. **DNS and BIND, by Cricket Liu and Paul Albitz**. Several years ago, I configured my first DNS by reading online documentation. I brought this book to understand how DNS and BIND works. I've already upgraded this book twice when a newer edition was released. This should definitely be in your library, if you are a serious system administrator.

8. **Understanding the Linux Kernel, by Daniel Bovet and Marco Cesati.** If you are a serious developer on Linux environment or a sysadmin, this is a must read. This books explains the inner workings of the Linux Kernel 2.6 in a structured and logical way. This talks about how Kenel handles the Memory Management, Process scheduling, I/O architecture and Block devices. Overall this book is a treat for geeks who are curious to explore what is under the hood of Linux.

9. **Linux Cookbook, by Carla Schroder**. This book covers Linux features from both users and system administrators point of view. There are

# 2 Cartorio de Resgistro de Documentos e Titulos e Pessoas Fisicas de Sao Paulo

Central IRTD -
Rua XV, n 40, Centro - Sao Paulo,SP

two chapters dedicated for installing and managing software on RPM-based system and Debian. If you use RedHat, the Linux Pocket Guide, by Daniel J. Barrett is an excellent addition to your library, which covers all the essential Linux command with a sample usage.

10. **Linux Firewalls, by Michael Rash**. To build a secure Linux system, you must read this book. There are quiet few books out there for iptables. But, this one talks specifically about the fundamentals of how to configure an Intrusion Detection System using iptables, psad and fwsnort. If you want a comprehensive handy reference of all the things iptables can do with specific examples, Linux Iptables Pocket Reference, by Gregor N. Purdy is the best.

11. **Linux Administration Handbook, by Evi Nemeth, Garth Snyder and Trent R. Hein**. During my early days of system administration, I've referred this book frequently. This is pretty detailed book with close to 1000 pages and 30 chapters that are nicely grouped together in three high level sections — Basic Administration, Networking and Bunch O' Stuff.

12. **Beginning Ubuntu Linux, by Keir Thomas and Jaime Sicam**. For those who like to transition from Windows to Linux, install Ubuntu Linux on one of your old laptop or desktop and get this book. I strongly believe in spreading the news about Linux to those who don't use it. If you want any of your loved ones or friends to learn Linux, install Ubuntu on an old laptop and give this book as a gift to them. They'll definitely be very thankful to you.

# Extended Reading

Following are few articles from The Geek Stuff blog for your extended reading. Check out Best Of The Blog section for more articles.

- Turbocharge PuTTY with 12 Powerful Add-Ons
- Nagios - Enterprise Monitoring Solution
  - Nagios Jumpstart Guide
  - Monitor Window Server
  - Monitor Linux Server
  - Monitor Network Switch
  - Monitor VPN Device
- Perform SSH and SCP without entering password:
  - From openSSH to openSSH
  - From openSSH to SSH2
  - From SSH2 to SSH2
- Vi / Vim Tips and Tricks
  - Vim Macro Tutorial: How To Record and Play
  - How To Use Vim as Perl IDE and C/C++ IDE
  - Automatic Word Completion in Vim
  - 3 Steps to Add Custom Header to a File Using Vim
- The Ultimate Guide for Creating Strong Passwords
- Firefox Add-On: Hire 7 Personal Bodyguards to Browse Internet Securely
- Tripwire Tutorial: Linux Host Based Intrusion Detection System
- Midnight Commander (mc) Guide: Powerful Text based File Manager for Unix

# Your Feedback and Support

I hope you found **Linux 101 Hacks** eBook helpful. Thanks for reading. I sincerely appreciate all the support given by the regular readers of my blog. Without your tremendous support, it would've been difficult to find the motivation to write this eBook.

## Subscribe to TGS

To get Linux Tips, HowTos, Guides and Tutorials on an on-going basis, please subscribe to The Geek Stuff blog. If you subscribe, you will get new articles posted on TGS website directly to your inbox or to your RSS reader.

## Contact TGS

Please use this contact form to send me your feedback, question, or clarification on any of the 101 hacks mentioned in this eBook.