

# PHP 4 Reference Card

by Steven R. Gould

## Escaping HTML

preferred format	<code>&lt;?php ...?&gt;</code>
verbose	<code>&lt;script language="php"&gt;...&lt;/script&gt;</code>

Less portable, may be disabled in php.ini:

short-form	<code>&lt;?...?&gt;</code>
short-form expression	<code>&lt;?=expr?&gt;</code>
ASP-style	<code>&lt;%...%&gt;</code>
ASP-style expression	<code>&lt;%=expr%&gt;</code>

## Basic syntax

statement terminator	<code>;</code>
C-style comments	<code>/*...*/</code>
C++-style comments	<code>//...</code>
shell-script-style comments	<code># ...</code>
block delimiters	<code>{ }</code>
octal integers (prefix zero)	<code>0</code>
hexadecimal integers (prefix zero-ex)	<code>0x</code> or <code>0X</code>
newline, cr, tab, backspace	<code>\n, \r, \t, \b</code>
special characters	<code>\\, \?, \', \", \\$</code>

## Data Types

boolean	<code>TRUE/FALSE</code>
integer	<code>-101, 23, 69</code>
floating point	<code>3.141592</code>
character string (parsed)	<code>"..."</code>
character string (unparsed)	<code>'...'</code>
class	<code>class name {...}</code>
resource (refer to PHP manual for details)	
array	<code>array([<i>index=&gt;</i>]<i>value</i>,...)</code> where <i>index</i> can be non-negative int or string

## Predefined PHP variables

Many variables are defined that are specific to the web server and OS. Run `phpinfo()` for a complete list of these.

<code>\$argv</code>	array of arguments passed to script
<code>\$argc</code>	number of arguments in <code>\$argv</code>
<code>\$PHP_SELF</code>	filename of currently executing script

The following are only available if `track_vars=0n` in `php.ini`

<code>\$HTTP_COOKIE_VARS</code>	array of variables passed via cookies
<code>\$HTTP_GET_VARS</code>	array of variables passed via GET
<code>\$HTTP_POST_VARS</code>	array of variables passed via POST
<code>\$HTTP_POST_FILES</code>	array of files uploaded via POST
<code>\$HTTP_ENV_VARS</code>	array of variables from parent environment
<code>\$HTTP_SERVER_VARS</code>	array of variables from HTTP server

## Control Structures

<code>include filename</code>	include named file (only if executed)
<code>include_once filename</code>	include named file (at most once)
<code>require filename</code>	include named file, like C/C++ <code>#include</code>
<code>require_once filename</code>	include named file, if not already included

## Flow of Control

exit from <code>switch</code> , <code>while</code> , <code>do</code> , <code>for</code>	<code>break</code>
next iteration of <code>while</code> , <code>do</code> , <code>for</code>	<code>continue</code>
go to (avoid if possible!)	<code>goto label</code>
label	<code>label:</code>
return value from function	<code>return expr</code>
terminate execution	<code>exit(arg)</code>

**Flow Constructions** (if/while/for/do/switch)

<code>if (expr) statement</code>
<code>else if (expr) statement</code>
<code>else statement</code>
<code>for (expr<sub>1</sub>; expr<sub>2</sub>; expr<sub>3</sub>) statement</code>
<code>while (expr) statement</code>
<code>do statement</code>
<code>while(expr);</code>
<code>switch (expr) {</code>
<code>0</code>
<code>case const<sub>1</sub>: statement<sub>1</sub> break;</code>
<code>case const<sub>2</sub>: statement<sub>2</sub> break;</code>
<code>default: statement</code>
<code>}</code>

## Functions

```
function name([arg,...,arg[=default]]) {  
    statement  
    return [value];  
}
```

## Classes and objects

A “class” is a collection of related variables and functions  
**Note<sub>1</sub>**: constructors in derived classes do not call constructors in base classes! You must do this, if you want this behavior.  
**Note<sub>2</sub>**: all members are public

**Class definition**

<code>class name [extends base]{</code>	
<code>var name;</code>	declare member variables
<code>function name {...}</code>	function declarations
<code>};</code>	end of class definition

**Using classes**

create new instance of class	<code>var = new name([arg,...])</code>
accessing member variable	<code>object-&gt;member</code>
calling member function	<code>object-&gt;fnc([arg,...])</code>
current object, from within class	<code>\$this</code>
to reference parent class from base class	<code>parent::</code>
to access members without class instance ::	

## Operators (decreasing precedence)

<u>new operator</u>	<u>new</u>
<u>array member accessor</u>	<u>[]</u>
not [logical operator]	!
ones compliment [bit operator]	~
increment, decrement	++, --
error control operator	@
multiply, divide, modulus (remainder)	*, /, %
addition, subtraction	+, -, .
left, right shift [bit operations]	<<, >>
comparison operators	>, >=, <, <=
equality operators	==, !=, ===, !==
bitwise and	&
bitwise exclusive-or (xor)	^
bitwise or (inclusive-or)	
logical and	&&
logical or	
conditional expression	<code>expr<sub>1</sub>?expr<sub>2</sub>:expr<sub>3</sub></code>
assignment operators	=, +=, -=, *=, ...
print operation	print
logical and	and
logical xor (exclusive-or)	xor
logical or (inclusive-or)	or
list operator	,

## Predefined Apache variables

More commonly used variables defined by Apache web server are listed below. Run `phpinfo()` for a complete list.

<code>\$SERVER_NAME</code>	name of the web server
<code>\$SERVER_SOFTWARE</code>	server ID string, used in HTTP response
<code>\$SERVER_PROTOCOL</code>	HTTP protocol used to request page
<code>\$REQUEST_METHOD</code>	request method: GET, HEAD, POST, PUT
<code>\$QUERY_STRING</code>	query string via which page was accessed
<code>\$DOCUMENT_ROOT</code>	root directory under which script is running
<code>\$HTTP_REFERER</code>	referring URL
<code>\$HTTP_USER_AGENT</code>	user's browser string
<code>\$HTTP_REMOTE_ADDR</code>	user's IP address
<code>\$HTTP_REMOTE_PORT</code>	port used on user's machine
<code>\$SCRIPT_FILENAME</code>	absolute path name of script
<code>\$SERVER_ADMIN</code>	server administrator's e-mail address
<code>\$SERVER_PORT</code>	port on server; e.g. HTTP 80, HTTPS 443
<code>\$PATH_TRANSLATED</code>	path of script relative to filesystem
<code>\$SCRIPT_NAME</code>	path of script relative to document root
<code>\$REQUEST_URI</code>	the requested URI; e.g. <code>'/index.html'</code>

# PHP 4 Reference Card

## String Functions <string>

return specific character	<code>chr(n)</code>
return ASCII value of character	<code>ord(c)</code>
length of string	<code>strlen(s)</code>
<b>String formatting/output</b>	
output string(s)	<code>echo(s[,...])</code>
output string	<code>print(s)</code>
output formatted string	<code>printf(s[,arg])</code>
return a formatted string	<code>sprintf(s[,arg])</code>

### String comparison

binary safe case-sensitive compare	<code>strcmp(s<sub>1</sub>,s<sub>2</sub>)</code>
binary safe case-sensitive compare	<code>strcmp(s<sub>1</sub>,s<sub>2</sub>,len)</code>
binary safe case-insensitive compare	<code>strcasecmp(s<sub>1</sub>,s<sub>2</sub>)</code>
binary safe case-insensitive compare	<code>strncasecmp(s<sub>1</sub>,s<sub>2</sub>,len)</code>

### Searching strings

find position of 1st occurrence of char.	<code>strpos(h,n[,offset])</code>
find position of last occurrence of char.	<code>strrpos(h,n)</code>
find first occurrence of string	<code>strstr(h,n)</code>
case-insensitive version of strstr	<code>striestr(h,n)</code>
find last occurrence of char.	<code>strrchr(h,n)</code>

### String manipulation

convert to upper/lower case	<code>strtoupper(s)/strtolower(s)</code>
trim whitespace from start of string	<code>ltrim(s[,w])</code>
trim whitespace from end of string	<code>rtrim(s[,w])</code>
trim whitespace from start & end	<code>trim(s[,w])</code>
strip HTML&PHP tags from string	<code>strip_tags(s[,allow])</code>
reverse a string	<code>strrev(s)</code>
replace s <sub>1</sub> with s <sub>2</sub> in str	<code>str_replace(s<sub>1</sub>,s<sub>2</sub>,str)</code>
translate characters	<code>strtr(str,s<sub>1</sub>,s<sub>2</sub>)</code>
extract part of a string	<code>substr(s,start[,len])</code>

## Filesystem Functions <filesystem>

open file	<code>fopen(filename,mode)</code>
modes: r (read from beginning), w (overwrite), a (append)	
modifiers: + (open for read & write), b (binary mode)	
close file	<code>fclose(fp)</code>
retrieve current position in file	<code>ftell(fp)</code>
jump to position in file	<code>fseek(fp,offset[,whence])</code>
get next character from file	<code>fgetc(fp)</code>
get line from file	<code>fgets(fp[,len])</code>
read entire file into array	<code>file(filename)</code>
test for End Of File	<code>feof(fp)</code>
binary-safe file read	<code>fread(fp,len)</code>
write to file	<code>fputs(fp,s,len)</code>
flush output buffer	<code>fflush(fp)</code>
parse input from file	<code>fscanf(fp,format[,var...])</code>
binary-safe write to file	<code>fwrite(fp,s,len)</code>
copy a file	<code>copy(src,dest)</code>
available disk space	<code>diskfreespace(dir)</code>
test for existence of file	<code>file_exists(filename)</code>
echo all remaining data	<code>fpassthru(fp)</code>
is file readable?	<code>is_readable(filename)</code>
is file writable?	<code>is_writeable(filename)</code>

## Mathematical Functions <math>

trig functions	<code>sin(x), cos(x), tan(x)</code>
inverse trig functions	<code>asin(x), acos(x), atan(x)</code>
<code>arctan(y/x)</code>	<code>atan2(y,x)</code>
hyperbolic trig functions	<code>sinh(x), cosh(x), tanh(x)</code>
exponentials & logs	<code>exp(x), log(x), log10(x)</code>
powers	<code>pow(x,y), sqrt(x)</code>
rounding	<code>ceil(x), floor(x), abs(x)</code>
minimum, maximum	<code>min(x,...), max(x,...)</code>
random number	<code>rand(),rand(min,max)</code>

## Unified ODBC Functions <odbc>

connect to data source	<code>odbc_connect(dsn,user,pwd)</code>
close connection(s)	<code>odbc_close(id),odbc_close_all()</code>
retrieve last error/msg	<code>odbc_error(),odbc_errormsg()</code>
prepare SQL statement	<code>odbc_prepare(id,query)</code>
execute prepared SQL statement	<code>odbc_execute(id[,arg])</code>
prepare & execute SQL statement	<code>odbc_exec(id,query)</code>
get row as an array	<code>odbc_fetch_into(id[,row,result])</code>
fetch a result row	<code>odbc_fetch_row(id[,row])</code>
get result from a field	<code>odbc_result(id,field)</code>
free result resources	<code>odbc_free_result(id)</code>
number of rows in result	<code>odbc_num_rows(id)</code>
output results in HTML table	<code>odbc_result_all(id[,format])</code>

### Transactions

toggle autocommit on/off	<code>odbc_autocommit(id)</code>
commit transaction	<code>odbc_commit(id)</code>
rollback transaction	<code>odbc_rollback(id)</code>

## Session Handling Functions <session>

initialize session data	<code>session_start()</code>
destroy current session data	<code>session_destroy()</code>
get/set session name	<code>session_name([s])</code>
get/set session ID	<code>session_id([s])</code>
register variables in session	<code>session_register(name[,...])</code>
unregister variable	<code>session_unregister(name)</code>
variable is registered?	<code>session_is_registered(name)</code>
get cookie parameters	<code>session_get_cookie_params()</code>
set cookie params	<code>session_set_cookie_params(l[p,s])</code>
write data & close session	<code>session.write_close()</code>

## Miscellaneous Functions <misc>

evaluate string as PHP code	<code>eval(s)</code>
terminate script	<code>exit(x),exit(s)</code>

### Date/Time functions

format a local date/time	<code>date(format[,timestamp])</code>
current time in secs. since Jan.1, 1970	<code>time()</code>
current time in microseconds	<code>microtime()</code>

### External program execution

The following can be used to execute an external program.

They differ in their handling of the output.

output returned in result array	<code>exec(prg[,result,status])</code>
output returned as string	<code>shell_exec(prg)</code>
display output	<code>system(prg[,status])</code>
display raw output	<code>passthru(prg[,status])</code>

## Reference

PHP web site	<a href="http://www.php.net/">http://www.php.net/</a>
Zend Technologies	<a href="http://www.zend.com/">http://www.zend.com/</a>
PHP Builder	<a href="http://www.phpbuilder.com/">http://www.phpbuilder.com/</a>
Knowledge Base	<a href="http://php.faqts.com/">http://php.faqts.com/</a>
Apache web server	<a href="http://httpd.apache.org/">http://httpd.apache.org/</a>

January 2002 v1.0. Copyright © 2002 Steven R. Gould

Permission is granted to make and distribute copies of this card provided the copyright notice and this permission notice are preserved on all copies.

Send comments and corrections to Steven R. Gould, Publishing Writes, Dallas, TX 75252, USA. (sgould@publishingwrites.com)