**PHP Language Guide**

Learning a NEW language is easier if you compare its structure and syntax to a language you already are familiar with. In this guide we will explore the similarities and differences between PHP and Pascal

Lazarus/FreePascal = compiled and run on client; PHP = interpreted line at a time by server, run there and results sent to the client

|  |  |  |
| --- | --- | --- |
| Concept | Pascal | PHP **<?php** code\_here **?>** |
| Comments | // single line comment{ } or (\* \*) for blocks of commented text | // single line comment/\* \*/ for blocks of commented text |
| Variables | Pascal is STRONGLY TYPED, in that every variable you use has to be declared up front and assigned a data type.Start with a letter, can contain letters, digits, underscoresWe have a "declaration" and an "assignment". Eg: var x : integer;begin x := 42; | PHP is LOOSELY TYPED – you introduce variables as you need them and the language manages the data type.Start with a $, then letter or underscore, can contain letters, digits, underscoresWe "declare" our variables at the time we use them. Eg:$x = 42; |
| Syntax | Statements in Delphi are separated with semi-colons | Command lines in PHP end in a semi-colon |
| [Operators](http://www.w3schools.com/php/php_operators.asp) |

|  |  |
| --- | --- |
| Is the same as | = |
| Takes on the value of | := |
| Is different from | <> |
| String catenation | + |
| Add,subtract,multiply,divide | + - \* / |
| Modulus (div remainder) | Mod |
| Increment/decrement | inc(x); dec(x) |
| Logical | and or not |

 |

|  |  |
| --- | --- |
| Is the same as | == |
| Takes on the value of | = += -+ /= .= etc. combos |
| Is different from | != |
| String catenation | . |
| Add,subtract,multiply,divide | + - \* / |
| Modulus (div remainder) | % |
| Increment/decrement | ++x; --x; x++; x-- |
| LOGICAL | And && Or || Not ! |

 |
| Output | We are used to putting messages in captions, showmessages etcMost output is string in nature (with the possible exception of .value and .position properties being numeric) | The output from PHP is HTML (web stuff), there are a few simple commands to do this:**echo** "whatever"; **echo** $x . " is the value of x";[**printf**](http://www.w3schools.com/php/func_string_printf.asp)("a message **%s** with a value in the middle",**$x**);note: even though variables may be numeric, they are outputted as string (automatically) |
| Decisions | Simple binary selection**if** condition **then** task1 **else** task2these can be "nested"Multiple selection**case** variable **of** value1 : task1; value2 : task2 **else** taskn**end**;the *else* is optional, it catches all un-tested valueswhere “tasks” are either single Delphi statements of blocks contained within a **begin** … **end**  | Simple binary selection**if**(condition) { task1;} **else** { Task2;}Multiple selection

|  |  |
| --- | --- |
| **switch**(variable){**case** value1: task1; **break**;**case** value2: task2; **break**;**default**: defaulttask} | **if** (condition1) { task1;} **elseif** (condition2) { task2;} **elseif** (condition3) { task3;} **else** { task4;} |

 |
| Loops |

|  |  |
| --- | --- |
| ***Pre-tested definite***Use when you KNOW how many times something needs to be done | For to do loop:**for** thing := a **to** b **do** task;**for** thing := b **downto** a **do** task; |
| ***Pre-tested indefinite***Use when the task might need doing 0 to many times | While do loop:**while** condition\_is\_true **do** task; |
| ***Post tested indefinite***Use when the task needs doing 1 to many times | Repeat until loop:**repeat** task**until** condition\_is\_true; |

 |

|  |  |
| --- | --- |
| Pre-tested definite | For loop:**for** ($x=1; $x<=5; $x++) { Task;}*for (start value; keep looping condition; inc)* |
| Pre-tested indefinite | While loop:**while** condition\_is\_true { task;} |
| Post tested indefinite | Do While loop:**do** { task;} **while** condition\_is\_true; |

 |
| Strings | length(targetString) returns how many chars are in it | strlen(targetString) returns how many chars are in itltrim(targetstring) gets rid of leadingrtrim(targetString) gets rid of trailing spaces |
| Arrays | Defined (usually) as type, declared as a variable, user selects indexing systemType thingarray = array[x..y] of datatypeVar example : thingarray | $thing = array(‘val1’,’val2’,’val3’)$thing[0]Arrays zero indexed – can contained mixed types (more like a list)Multi-dimensioned arrays are arrays of arrays  |
| Randoms | random(x) delivers a random between o and x-1 | rand(min,max) makes a random number between min and max |
| Forms | Form container – 2 methods (post most common) | <form name="somename" method="post" action="webpage"></form> |
| Form elements | Submit button – causes "post" event | <input type="submit" name="somename" value="buttoncap"> |
| Text input box | <input type="text" name="somename"> |
| Password Input box | <input type="password" name="somename"> |
| Radio Group | <input type="radio" name="somename" value="option1"><input type="radio" name="somename" value="option2"> |
| Checkboxes | <input type="checkbox" name="somename" value="option1"><input type="checkbox" name="somename" value="option2"> |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |