

TPS

(Tiny PHP with SQL)

Jonathan T. Cotton

The primary purpose that Tiny PHP with SQL (TPS) serves is to provide a straight forward, non-cluttered, and highly efficient Hypertext Preprocessor based upon the commonly used programming language known as PHP.

“PHP is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML.” (PHP.net) The most recent versions of PHP contain many features that are unused by the vast majority of the developers that utilize the language. For example, there are 14 specialized functions included in the base PHP language that are specifically for interfacing with Lotus Notes. There are also many other un-needed function sets as well, totaling well into the tens of thousands of built in functions, of which, most developers only use a handful of. These additional functions, many of which cause additional processing requirements, adversely affect the speed of the PHP scripts that are run.

TPS solves this problem by including only the most commonly used programming elements, as well as a very basic SQL function set that allows the developer to interact with a MySQL server. By only including these very basic programming elements, the speed of the Preprocessor (also known as an interpreter) can be drastically increased.

PHP is commonly known for it's ability to accomplish most web based programming challenges, however, it's speed is a major drawback at times which results in the use of other, faster languages, such as Perl, even though they are less intuitive than PHP.

The choice of utilizing a MySQL server as the default SQL server interface was done because of its popularity and availability. As a freely available and widely adopted SQL server, MySQL is used by a large majority of developers.

The full syntax of TPS has been outlined below using BNF Rules. The entire syntax takes up less than two pages, demonstrating the languages simplicity, and succinctness.

Syntax:

The → character in each BNF Rule represents “the meaning of ___ is”. For example, “Example → Stuff” stands for “The meaning of Example is Stuff”.

Anywhere text is displayed in **bold** on the right hand side of one of the BNF Rules, that text is an actual string that is included in the tps file. For example, the BNF Rule for Open below specifies that in the actual TPS script file, the string “<?tps” should be included.

BNF Rules:

TPS → CodeSet VarChar CodeSet

CodeSet → Open Code Quit

Open → **<?tps**

Quit → **?>**

Code → Code Code

Code → Sub

Code → Test

Code → Assignment

Code → Display

Code → While

Code → SQL;

Code → FuncCall;

Code → Func

Func → FuncName (VarList) Sub

VarList → VarList VarList

VarList → Var

FuncCall → FuncName (VarList)

FuncName → **[a-zA-Z][a-zA-Z0-9]***

Sub → { Code }

Test → **if (Expr)** Sub

Test → Test **else** Sub

Test → Test **else** Test

Expr → (Expr **&&** Expr)

Expr → (Expr **||** Expr)

Expr → Var **==** Var

Expr → Var != Var
Expr → Var < Var
Expr → Var > Var
Expr → Var <= Var
Expr → Var >= Var
Expr → Var

Display → **echo** String;

Assignment → VarName = Var;

VarName → $\$[a-zA-Z]$ String

Var → VarName
Var → Num
Var → String
Var → (Var + Var)
Var → (Var - Var)
Var → (Var * Var)
Var → (Var / Var)
Var → (Var % Var)
Var → SQL
Var → **TRUE**
Var → **FALSE**

String → "VarChar"
String → String . String
String → Num

VarChar = $[a-zA-Z0-9<>:;'"[{}]\|+=_-)(*\&^\%\$#\@!~`.,/?]*$

Num → $[0-9.]^*$

While → **while** (Expr) Sub

SQL → **sql_query**(String)
SQL → **sql_query**(Var)
SQL → **sql_result**(VarName)
SQL → **sql_num**(VarName)

"PHP.net." PHP: Hypertext Preprocessor. 21 April 2006. The PHP Group. 21 April 2006 <<http://www.php.net>>.