

## The t3d programming language

- 1) t3d programming language includes C programming language (C99), except its famous "avoid-using-this-functions".
- 2) t3d programming language adds to C modernization in terms of; (a) support for environmental things, (b) use of libraries and external utilities, (c) its evolutive development by the community, (d) safe strings, (e) garbage collection, (f) networking, (g) improved datatype definitions, (h) GUI, (i) multi-threading, (j) support for handling standardized file types, (k) the use of t3d function prototype, (l) t3d can be used in applications, (m) scripts and (n) web-programming. But anyway most C is also t3d and most of t3d is also C.
- 3) At the beginning GCC can be used as t3d compiler. Portability is to be achieved with the #ifdef - #ifndef preprocessor defines. C++ support is to be obtained with the use of "extern C" definitions.

## The structure of t3d function prototype

\*\*\*\*\*

```
langsign_mainverb_aaaaaa_bbbbb_bbbbb_cccccc_dddddd_Reeeee_ADJECTIVE_PART1_n_ADJECTIVE_PART2_ARG1XX
X_ARG2XXX_ARG3XXX
```

\*\*\*\*\*

- 1) Almost all t3d - functions are expected to have many alias names (even hundreds). The user is expected and required to find one of suitable aliases, using logic and the help of IDE editor, personal memory / routine or a function / internet search engine or native language.
- 2) t3d is not a rigid language. Any legal formulation of a function name, which does help the user to do tasks, is ok.
- 3) t3d programming can be performed using most of the worlds written languages. Stable translations are to be used for reserved datatype words and main verbs within each language. t3d function names can be written using Unicode.
- 4) When t3d-structured function prototypes are to be used within other languages than t3d, the use of t3d function prototypes may require using some additional input parameters.
- 5) t3d programming language and the structure of t3d function prototype are trademarks of Juuso Hukkanen. Fair use of both trademarks is allowed within all programs which are licensed using Charity Open Source License.

**Training examples 1 - What does:** 1) t3d\_measure\_barray\_LENGTH 2) t3d\_convert\_file\_Rfile\_GSM2WAV 3) t3d\_convert\_Rfile\_READ\_ONLY 4) t3d\_calculate\_iarray\_Rdouble\_STD\_DEVIATION 5) t3d\_add\_byte\_Rbarray\_RANDOM\_n\_END 6) t3d\_environment\_barray\_SET\_KEYBOARD\_n\_LANGUAGE

**Training examples 2 - How to:** 1) Change the text contents of a text file to uppercase (English text) 2) Compress a file using RLE compression algorithm and put the compressed content to another file 3) Change a file format from JPG to GIF 4) Check if a number 234524 is a prime number 5) Calculate how many times a word "hello" occurs in a file independently of the case of letters 6) Check whether a given text string contains a valid internet domain name

### **Return value:**

All t3d functions only return a 64bit signed integer number. All negative return values indicate errors each with respective error numbers. A returned zero means zero, all returned positive values do mean a meaningful correctly calculated numeric answer. In function which answers to yes/no kind of questions returned 0 = NO and 1 = YES.

### **Language identifier (langsign):**

It is possible to implement t3d programming within programs written using many other programming languages; such as C , C++ , PHP , Pascal , Python , Lisp , Cobol , Perl etc. The language identifier (langsign) is used in identifying the host programming language used inside the t3d structured function. The function contents must naturally be written using a particular host language, but one t3d structured function written with Ada should perform the same task as a similarly (except the langsign) named with host language PHP. Examples of used language identifiers:

```
t3d => t3d, C (also C++) => c3d, Pascal => p3d, PHP => ph3d,
Lisp => l3d, COBOL => CO3d, Ada => a3d, Perl => pe3d
```

### **Main verb (mainverb):**

There are 15 main verbs in t3d language. The mainverb defines the main category of action which the function will perform. Possible mainverbs are: add, ai, calculate, close, convert, create, crypto, environment, find, measure, move, open, read, remove and write.

### **Inputting (datatype) elements-part (aaaaa\_bbbbb\_ccccc\_ddddd):**

Each of the six letter strings (a-d) represents a data-parameter (of a certain datatype), which carries a parameter into the function. Order of inputting parameters is decided (in priority order) as follows (1) obvious direction <=> param1 goes-to param2, (2) age <=> first\_young\_param then\_old\_param, (3) size <=> first\_smaller\_data(type) then\_bigger\_data(type), (4) alphabetical order (aaa\_param before bbb\_param).

There are 20 different datatypes in t3d; byte, wbyte, int, long, double, bignum, barray, warray, iarray, larray, darray, bigarray, time, table, url, file, dirpath, gtable, gobject. Each datatype can carry different kind of data to / from function.

Datatype	Description:
byte	Size 8-bits can contain values 0-255
wbyte	Size 32-bits can contain Unicode text (UTF-32)
int	Size 32 bits can contain signed integers
long	Size 64 bits can contain signed integers
double	Size 64 bits can contain floating point numbers
bignum	Size unlimited. Bignums are barrays containing a number as text
barray	An array consisting of bytes, (optional use of delimiters included)
warray	An array (for Unicode text(UTF-32)) consisting of wbytes (use of delimiters allowed)
iarray	An array consisting of integers (use of delimiters included)
larray	An array consisting of longs (use of delimiters included)
darray	An array consisting of doubles (use of delimiters included)
bigarray	An array consisting of bignums (use of delimiters included)
time	High-resolution floating point presentation of seconds since Jan 1 <sup>st</sup> year 0 (00:00am)
table	Like a SQL- table(rows, columns, headers); any datatype fits into each cells
url	Unique Resource Locator (internet address or any other exact location)
file	A (binary) file, max file length name is 4095 bytes
dirpath	Max size 4095 bytes, identifier path to a directory/folder
gtable	GUI-window with build-in menu-tree structures. Carrier of gobjects
gobject	GUI-object (like an unit in CSS);defines appearance and reflex-reactions of UI-units
process	Semi-independent program within a program, can also be a thread or ext. program

#### **Result (data from function) - part (Reeeeeee):**

Also known as the *R-parameter*. The return value of t3d-functions is always an integer type number, so when a t3d-function needs to things to data in other datatypes, the results can not be returned as a return value. The result field of a function is described by a capital R. The datatype following the R will contain the result-data which is returned back from the function. Result data will be written to a datatype which is first given to a function using *pass by reference* or equivalent techniques.

#### **How to do (the main verb) - part (ADJECTIVE\_PART1\_n\_ADJECTIVE\_PART2):**

Also known as the *n-parameters*. As earlier said there are only 15 main verbs in t3d. The main verbs of t3d do only define the rough action but when that rough action is fine adjusted with some extra parameters, the function is able to achieve very precise actions.

There can be max two adjective parts in each t3d structured function. A character combination *n* <underscore+lower case n +underscore> is used in separating the two possible adjective parts. Words in adjective parts are written using capital letters. Both adjective parts can contain one main verb action defining parameter. Each defining parameter is to be expressed using one or two words separated by underscore-characters. As an example following are to be considered as legal single defining parameters: SHARPEN\_IMAGE, TRS\_ROUTE, USER\_SESSION, SPACE\_SEPARATED, EXTRACT\_HIDDEN, RAM\_AVAILABLE, THIS\_USER, CPU, SET, LANGUAGE, LOCAL. If a defining parameter contains the word "IS" (e.g. IS\_VALID) the function returns a YES or NO answer.

There are two exception 'words' which will not be calculated as a single word, first is "IS" (as a yes/no question IS\_something), a second word not-counted as a word is "TO" (meaning "to something"). TO is to be always replaced with a number 2, in order to highlight its special meaning. There may not be an underscore letter between the number 2 and the other parts of SAME verb defining parameter. Thus the following are legal single defining parameters: INT2BARRAY, TEXT2IMAGE, TRANSLATE2, RENAME2, WAV2OGG, MP32WAV, 2BASE, INT2ROMAN, IS\_FILE\_ACCESS, IS\_SWEDISH

The words which are to be used within the verb defining parameters shall be in their simplest singular form, but abbreviations which are familiar to most of the programmers may be used such as INT,AES256,TAB,ID.

The adjective parts are priority arranged; 1) to follow similar order as used for input datatypes, 2) secondly according to apparent significance and 3) by alphabetical order. Examples of arranging order (according to order given by datatypes) FILENAME\_n\_FIRST, (according to significance) IS\_ACCESS\_n\_WRITE (According to alphabetical order) CPU\_n\_SPEED\_MHZ.

#### **Additional parameters a.k.a the X-parameters (ARG1XXX\_ARG2XXX\_ARG3XXX)**

In some cases the fine-adjustment of defining the main verb is not enough or the parameters to function are too numerous for creating easy to use function names. In those cases the X- parameters can be used in feeding data accurately into the function. The max number of X- parameters per function is 10. Each individual X-parameter is identifiable by having XXX at the end. When using the function, the X-parameters are to be inserted after the R-part (result parameter outputting data from function). The order of these additional parameters is the same as they are listed in function prototype. For example the following are legal X-parameters LEVELXXX, PASSWORDXXX, WIDTHXXX\_HEIGHTXXX, BITXXX, SENDERXXX\_RECIPIENTSXXX\_SUBJECTXXX\_BODYXXX, PRESSKEYXXX. The inner mechanism for inputting the X-parameters into functions are like the typical mechanism for inputting variable arguments (varargs) into functions. X-parameters should not be utilized using pointer- references to variables.

Now look again those previously shown training examples. Answers to exercises, more exercises, code and discussion forums are to be found from [www.tele3d.com](http://www.tele3d.com) .