Forth Server Pages

Word and variable list

Author: Igor A. Maznitsa
Version: v2.0b
Date: 11-May-2001

Data types

An information and stack unit has width 64 bit with sign

Stacks

There are two stacks in the FSP, the arithmetical stack and the return stack.

User definited words

New words are defined by a user with ":" and ";" words. Name of new words must not concur with names of predefined words, in the case the new defined words will be ignored.

Variables

There are two kinds of variables: global variables and local variables. The global variables exist during all work period of program and accessible in any part of the program. The local variables can be used in user-defined words only.

Word	Arith. Stack	Return stack	Reference
1+	Int_1 ✍ int_2		The increment of the AS top.
1-	Int_1 ✍ int_2		The decrement of the AS top.
+	Int_1 int_2 ✍ int_3		To add two top values from AS and put the result onto the top.
-	Int_1 int_2 ✍ int_3		To subtract the AS top from the second value and the result to put onto the AS.
*	Int_1 int_2 ✍ int_3		To multiply two top values from AS and put the result onto the top.
/	Int_1 int_2 ✍ int_3		To divide the second AS value by the AS top and to put the result onto the AS.
AND	Int_1 int_2 ✍ int_3		The bitwise AND operation on the first and the second values of the AS.
OR	Int_1 int_2 ✍ int_3		The bitwise OR operation on the first and the second values of the AS.
XOR	Int_1 int_2 ✍ int_3		The bitwise XOR operation on the first and the second values of the AS.
NOT	Int_1 ✍ int_2		The bitwise NOT operation on the top value of the AS.
LOCAL	--		To reserve a place for a local variable with the specified name.
L!	Int addr_int ✍		To write the second AS value to the local variable with the name on the AS top.
L@	Addr_int ✍ int		To place the value from a local variable onto the AS stack.
DATE_BREAK	Date_Int ✍ dd mm yyyy		To extract day-month-year value from the packed date value on the top of AS.
DATE_PACK	Dd mm yyyy ✍ date_int		To pack day-mont-year data in the AS top to a packed date value and to

			place it onto the AS top.
DATETIME	**✍ date_int**		**To put on the AS top current date-time value.**
TIME_BREAK	**Int ✍ hh mm ss**		**To extract hour-minute-seconds from a packed date value in the AS top.**
SETHH	**Date_int int ✍ date_int**		**To change the hour value in the packed date value in the second element. New hour value in the top of the AS.**
SETMM	**Date_int int ✍ date_int**		**To change the minute value in the packed date value in the second element. New minute value in the top of the AS.**
SETSS	**Date_int int ✍ date_int**		**To change the seconds value in the packed date value in the second element. New seconds value in the top of the AS.**
WEEKDAY	**Date_int ✍ int**		**To extract the weekday number onto the AS top. (0 – the Sunday, 6 – the Saturday)**
EMIT	**Int ✍**		**Out the byte from the AS top to the current stream.**
SETINPSTREAM	**Int ✍**		**Set the current input stream: 0 – the socket, 1 – a opened input file. 2 – current input stream from a database. 3 – the user socket.**
SETOUTSTREAM	**Int ✍**		**Set the current output stream: 0 – the buffer (default), 1 – the socket, 2 – the server console, 3 – the opened output file, 4 – an user socket.**
RD_BYTE	**waittime ✍ int**		**Read a byte from current input stream and put it onto the As top. Waittime – time out of waiting of data. If the stream is empty, -1 will be placed onto the AS top.**
RD_STR	**Waittime str_indx ✍ 0 |-1**		**Read a string from current input stream to the string with the number on the AS top. Waittime – time out of waiting of data. If the stream is empty, -1 will be placed onto the AS top, else 0.**
HEREI	**✍ int**		**Place onto the AS top current size of global variable area.**
S2HTML	**Str_indx ✍ str_indx**		**Format a string to HTML compatible format.**
S2SQL	**Str_indx ✍ str_indx**		**Format a string to SQL compatible format.**
I,	**Int ✍**		**Increase global memory area and write the AS top to the area.**
CHR+	**Str_indx int ✍**		**Add the symbol (with the code on the AS top) to the string with the index in the second AS value.**
TKN_SET	**Str_indx int ✍ tkn_indx**		**To create a tokenizer for a string with the index in the second AS value. The code of delimiter char in the first value. After the operation you have the delimiter index in the top AS.**

TKN_NXT	**Tkn_indx str_indx ✍**		**Place next token from a tokenizer with the index in the second value to the string on the top.**
TKN_?	**Tkn_indx ✍ [-1|0]**		**Check tokenizer. If the tokenizer is empty then 0 will be placed, else –1.**
1+!	**Addr_int ✍**		**Increment of a cell in global memory area with the index on the AS top.**
S!	**Str_src str_dst ✍**		**Copy a string to other string.**
S_PS	**Str_indx substr_indx ✍ str_indx [pos_int|-1]**		**Find a substring in a string. If the substring has been found then its position is placed onto the AS top else –1.**
CHR	**Int str_indx ✍ str_indx**		
S_UC	**Str_indx ✍ str_indx**		**Uppercase operation for the string in the AS top.**
S_LC	**Str_indx ✍ str_indx**		**Lowercase operation for the string in the AS top.**
S_AT	**Str_indx int ✍ str_indx [-1|int]**		**Put a character code from the string onto the AS top. If the position is non in the string borders then –1 will be placed.**
S_LN	**Str_indx ✍ int**		**Put the string length onto the AS top.**
S_TR	**Str_indx ✍ str_indx**		**Trim operation for the string in the AS top.**
S_SB	**Str_src str_dest start_pos length ✍**		**Copy a substring from a string to other string. If the substring length is less than 0 then the string will be copied to the end.**
LEAVE	**--**		
IF	**Int ✍**		**The beginning of an IF-ELSE-THEN construction. If the value in the AS top is equivalent 0 then jump to ELSE else to process words after IF.**
THEN	**--**		**The ending of an IF-ELSE-THEN construction.**
ELSE	**--**		
.	**Int ✍**		**Put the value from the AS top into the current stream.**
S.	**Str_indx ✍**		**Put the string from the AS top into the current stream.**
S+	**Str_indx str_indx2 ✍ str_indx**		**Concatenate two strings and to place the result in the first string.**
DROPALL	**[a] ✍**		**Clear all the AS stack**
DROPALLR		**[a] ✍**	**Clear the RS stack.**
DROP	**Int ✍**		**Remove the AS top.**
DROPR		**Int ✍**	**Remove the RS top.**
DUP	**Int_1 ✍ int_1 int_1**		**Duplicate of the AS top.**
DUPR		**Int_1 ✍ int_1 int_1**	**Duplicate of the RS top.**
SWAP	**Int_1 int_2 ✍ int_2 int_1**		**Swap the two top values in the AS.**
OVER	**Int_1 int_2 ✍ int_1 int_2 int_1**		
ROT	**Int_1 int_2 int_3 ✍ int_2 int_3 int_1**		
ROTR		**Int_1 int_2 int_3 ✍ int_2 int_3 int_1**	
>R	**Int_1 ✍**	**✍ int_1**	
R>	**✍ int_1**	**Int_1 ✍**	
R@	**Int_1 ✍ int_1**	**✍ int_1**	**Copy the RS top to the AS top.**
0=	**Int_1 ✍ [0|-1]**		**Compare the AS top with zero. If the top is zero then**

			–1 will be placed, else 0.
0>	Int_1 ✍ [0|-1]		
0<	Int_1 ✍ [0|-1]		
>	Int_1 int_2 ✍ [0|-1]		
<	Int_1 int_2 ✍ [0|-1]		
<>	Int_1 int_2 ✍ [0|-1]		
=	Int_1 int_2 ✍ [0|-1]		
2*	Int_1 ✍ int_2		
2/	Int_1 ✍ int_2		
!	Int_val int_addr ✍		Place the value to the global memory area.
@	Int_addr ✍ int_val		Place the value from the global memory area onto the AS top.
VARIABLE	[int]✍		Create a global variable if the AS has a value then the variable will be initialized with the value.
:	--		The beginning of a new word definition. Switch on the compilation mode.
;	--		The ending of a new word definition and switch the system in the interpretation mode.
NEXT	--	Int_1 int_2 ✍	To end the processing of current word.
DEPTH	✍ int		Place the depth of the AS onto AS.
DEPTHR	✍ int		Place the depth of the RS onto AS.
S=	Str_indx1 str_indx2 ✍ [0|-1]		Compare two strings.
S==	Str_indx1 str_indx2 ✍ [0|-1]		Compare two strings without case sensitive.
DO	Int_1 int_2 ✍	✍ int_2 int_1	The beginning of a DO LOOP cycle or DO +LOOP cycle. The AS top is the limit and the second value is the start value.
LOOP			Increment the counter of a DO-LOOP cycle and if it is more than the limit, the cycle will be ended.
+LOOP	Int ✍		Increment the counter of a DO-LOOP cycle bye the AS top and if it is more than the limit, the cycle will be ended.
I	✍ int		Place the current DO-LOOP cycle value onto the AS stack.
J	✍ int		Place the parent DO-LOOP cycle value onto the AS stack.
BEGIN	--		The beginning of a BEGIN UNTIL cycle
UNTIL	Int ✍		Check the value in the AS top and if it is not equivalent 0 then the cycle will be ended.
ALLOT	Int ✍ addr_int		Reserve a memory area with the length in the AS top, in the global memory area and place the first address in the AS top.
?FIND	✍ [0|-1]		Check existence of next word in the stream to the dictionary, if it exists then –1 else 0.
?	--		Compile next word in the stream if the word exists.
QUIT	--		Quit from the program.
S_ENC	Str_indx ✍ str_indx		Encode a string to the URL format.

S_DEC	**Str_indx ✍ str_indx**		**Decode a string from URL format.**
DB_AUTOCOMMIT	**Con_indx mode_int ✍**		**Set auto transaction mode for a DB connection. 0 – turn off, else turn on.**
DB_CLOSE	**Con_indx ✍**		**Close a DB connection.**
DB_CONNECT	**url_indx un_indx psw_indx codepage_indx ✍ [con_indx| err_code]**		**Create a DB connection. url_indx - the alias of the DB un_indx – the user name psw_indx – the password codepage_indx – the code page Example: "DataBase" "name" "password" "Cp1251" DB_CONNECT If all is ok then the index of new DB connection has been placed onto the AS top else will be generated an exception with the error code on the AS top (0 – timeout error).**
DB_EXQ	**Con_indx str_indx ✍ rs_indx**		**Create a ResultSet based on a SQL query for a DB connection. The SQL string should be on the AS top. (the word is used for SELECT usually)**
DB_EXU	**Con_indx str_indx ✍ int**		**Process SQL query for a DB connection, the result of the query will be placed onto the AS top. (Usually used for DELETE, UPDATE and other)**
DB_GETINT	**Rs_indx int ✍ int**		**Read the integer value from the column with the specified number from a RecordSet and to place it onto the AS top.**
DB_GETSTR	**Rs_indx int str_indx ✍**		**Read the string value from the column with the specified number from a RecordSet and to place it to a string.**
DB_LOAD	**Str_indx ✍ [-1|0]**		**To load a DB driver for the name. If all is ok then –1 else 0.**
DB_NEXT	**Rs_indx ✍ [0|-1]**		**Move the cursor for a RecordSet to next string. If the record set is empty then 0 else –1.**
DB_NGETINT	**Rs_indx str_indx ✍ int**		**Read the integer value from the column with the specified name from a RecordSet and to place it onto the AS top.**
DB_NGETSTR	**Rs_indx column_str_indx str_dst_indx ✍**		**Read the string value from the column with the specified name from a RecordSet and to place it to a string.**
DB_COMMIT	**Con_indx ✍**		**To end a successful transaction for a DB connection.**
DB_ROLLBACK	**Con_indx ✍**		**Rollback a transaction for a DB connection.**
DB_TRANS	**Con_indx mode ✍**		**Set transaction isolation mode for a DB connection.**
DB_NGETBLOB	**Rs_indx str_indx ✍**		**To open a binary stream from the name specified column of a Resultset.**

DB_GETBLOB	**Rs_indx int ✍**		**To open a binary stream from the index specified column of a Resultset.**
FILELIST	**Serv_alias dir_name ✍ tkn_indx**		**To create a string tokeinzer contains file list in a subdirectory of a virtual server. Directory names start with "/".**
DB_RSCLOSE	**Rs_indx ✍**		**Close a resultset.**
LOAD	**Str_index ✍**		**Execute a FSP file and return back to the FSP script**
PLAYFILE	**Str_indx ✍**		**Execute a FSP file but don't return back.**
I2S	**Int str_indx ✍**		**To convert a number to a string.**
I2H	**Int str_indx ✍**		**To convert a number to a hex string representation.**
S2I	**Str_indx ✍ int**		**To convert a string to a number.**
SEXECUTE	**Str_indx ✍**		**Execute the word with the name on the AS top.**
SFIND	**Str_indx ✍ [-1|0]**		**To find the word with the name on the AS top, if the word has been found then –1 else 0.**
IMG_CRT	**w h i ✍ img_indx**		**To create an image with W (width) and H (height) and fill it by I (background color).**
IMG_GPNT	**Img_indx x y ✍ img_indx color_indx**		**To get the color of a point of an image in X Y cords.**
IMG_SPNT	**Img_indx x y ✍ img_indx**		**Draw a point on an image in X Y cords by current color.**
IMG_SPAL	**Img_indx palette_indx ✍ img_indx**		**Set the palette to an image.**
IMG_DPAL	**Img_indx ✍ img_indx**		**Set default palette to an image.**
IMG_LINE	**Img_indx x1 y1 x2 y2 ✍ img_indx**		**Draw a line on an image with current pen color.**
IMG_LINEF	**Img_linef x y ✍ img_indx**		**Draw a line on an image with current pen color from the last work point**
IMG_CIRCLE	**Img_indx x y r ✍ img_indx**		**Draw a circle with the pen color in X Y cords with R radius.**
IMG_FILL	**Img_indx x y b ✍ img_indx**		**Fill the area with the pen color from X Y point . B – color of the border.**
IMG_ELPS	**Img_indx x1 y1 x2 y2 ✍ img_indx**		**Draw an ellipse what is inscribed into a rectangle with corner cords x1 y1 x2 y2**
IMG_RCTNG	**Img_indx x1 y1 x2 y2 ✍ img_indx**		**Draw a rectangle with the pen color. x1 y1 and x2 y2 – the corner cords.**
IMG_FRCTNG	**Img_indx x1 y1 x2 y2 ✍ img_indx**		**The same as IMG_RCTNG but fills the inside area with the pen color.**
IMG_RRCTNG	**Img_indx x1 y1 x2 y2 r ✍ img_indx**		**Draw a rounded rectangle. R – radius of corners.**
IMG_SETC	**Img_indx color_indx ✍ img_indx**		**Set the pen color.**
IMG_LDIMG	**Img_indx str_indx ✍ img_indx**		**Load an image from disk (format Rimage) to an image.**
IMG_DRW	**Img_dst img_src x y ✍ img_dst**		**Draw an image to other image in X Y cords.**
IMG_DRWT	**Img_dst img_src x y ✍ img_dst**		**Draw an image to other image in X Y cords with transparent color what is equivalent to the pen color.**
IMG_SENDWBMP	**Img_dst ✍**		**Send an image to output**

			stream as WBMP image. The format is black white so the image will be converted before sending. The pen color is used as border color during the covert time.
IMG_SENDGIF	Img_dst ✍		Send an image to the output stream as a GIF image.
IMG_LDPAL	String_index ✍ palette_index		Load the palette from a file.
IMG_SETPS	Img_indx point_size ✍ img_indx		Set the point size for an image.
IMG_GETMTX	Img_indx ✍ img_indx w h		Get width and height of an image.
IMG_SETINDXC	img_indx r g b color_indx ✍ img_indx		Set RGB for the specified color.
IMG_DRWTXT	Img_indx str_indx bck_color x y ✍ img_indx		Draw text to an image with the pen color to X Y position. If bck_color < 0 then background will be transparent.
IMG_GETINDXC	img_indx indx_color ✍ img_indx r g b		Get RGB for the specified color.
SENDHTTPHDR	Ext_indx ✍		Send to current stream the HTTP header with code 200 OK and with MIME type for extending on the AS top.
<%%	--		Start of FSP script
%%>	--		Next text goes to current stream.
CLRBUF	--		Clear inside text buffer.
PRIORITY	Int ✍		Set the priority for the script (1 – min , 10 –max)
DELAY	Int ✍		Wait the number of milliseconds and continue the script processing.
TRY	--		The beginning of TRY-EXCEPT-TRYEND block
RASE	--		Throw an exception
EXCEPT	--		The beginning of the exception processing block
TRYEND	--		The end of exception processing block.
OPENSCKT	Host_addr_str port_num time_wait ✍		To open the user socket. Host_addr – URL of the host Port_num – the port number Time_wait – the max time of waiting in milleseconds (0-unspecified).
OPENFILE	Alias_str file_str mode ✍		Open a file in a server directory. Alias_str – a server alias File_str – the file name Mode – the mode of opening: 0 – to open with rewriting, 1 – to open for applying, 2 – to open for reading, 3 – removing.
CLOSEOUTSTREAM			Close current output stream
CLOSEINPSTREAM			Close current input stream
BUFF2EMAIL	Addr_str ✍ 0 | -1		Send the buffer to email address. –1 – success, 0 – error.
GV?	Var_name ✍ 0|-1		Check existence of a global server variable with the specified name. –1 - exists
GVS!	Str_indx var_name ✍		Write a string to a server global value, if the value non exists, it will be

			created, If the value exists but has different type, an exception will be throw.
GVI!	**Int Var_name ✍**		**Write an integer number to a server global variable. It is like to GVS!**
GVS@	**Str_indx var_name ✍**		**Get string from a server global variable.**
GVI@	**Var_name ✍ int**		**Get integer from a server global variable.**

Information variables

1) For HTTP scripts

1	Name	Type	Reference
1	**FSP-AUTHSCHEME**	**Str**	Authentication scheme name
2	**FSP-AUTHSTR**	**Str**	String has been gotten from the user.If the AuthScheme is Base64 then the string automatically converts to the "Name:Password" representation.
3	**FSP-AUTHNAME**	**Str**	The user name from the authentication string.
4	**FSP-AUTHPSSWD**	**Str**	The user password from the authentication string.
5	**FSP-CIP0**	**Int**	The variables contain IP address parts of the user address: CIP3.CIP2.CIP1.CIP0
6	**FSP-CIP1**	**Int**	
7	**FSP-CIP2**	**Int**	
8	**FSP-CIP3**	**Int**	
9	**FSP-CNM**	**Str**	The computer name of the user
10	**FSP-QUERY**	**Str**	The variable contains the query to a resource, else the part of the query after '?'
11	**FSP-POSTBODY**	**Str**	Contains POST command content
12	**FSP-CNTTYPE**	**Str**	The MIME type of the content what is formed by the script.
14	**FSP-USERAGENT**	**Str**	The user agent information.

2) For temporary server scripts

1	Name	Type	Reference
1	**FS-STARTCOUNTER**	**Int**	**Contains the start number of the script during current server work session.**
3	**FS-SCRIPTNAME**	**Str**	**Contains the script alias**
4	**FS-FILENAME**	**Str**	**Contains the file name of the script.**