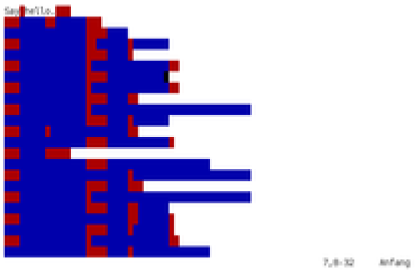
WIKIPEDIA
The Free Encyclopedia

Whitespace (programming language)



Whitespace hello world program with syntax highlighting

🟦 tabs

🟥 spaces

Whitespace is an esoteric programming language developed by Edwin Brady and Chris Morris at the University of Durham (also developers of the Kaya and Idris programming languages). It was released on 1 April 2003 (April Fool's Day). Its name is a reference to whitespace characters. Unlike most programming languages, which ignore or assign little meaning to most whitespace characters, the Whitespace interpreter ignores any non-whitespace characters. Only spaces, tabs and linefeeds have meaning.[1][2]

A consequence of this property is that a Whitespace program can easily be contained within the whitespace characters of a program written in a language which ignores whitespace, making the text a polyglot.[2]

The language itself is an imperative stack-based language. The virtual machine on which the programs run has a stack and a heap. The programmer is free to push arbitrary-width integers onto the stack (currently there is no implementation of floating point numbers) and can also access the heap as a permanent store for variables and data structures.

History

Whitespace was created by Edwin Brady and Chris Morris in 2002. Slashdot published a review of the programming language on 1 April 2003.[3]

The idea of using whitespace characters as operators for the C++ language had been facetiously suggested five years earlier by Bjarne Stroustrup.[4]

Syntax

Commands are composed of sequences of spaces, tab stops and linefeeds. All other characters are ignored and thus can be used for comments. For example, tab-space-space-space performs arithmetic addition of the top two elements on the stack.

Code is written as an *Instruction Modification Parameter* (IMP) followed by the operation.[1] The table below shows a list of all the IMPs in Whitespace.

IMP	Meaning
[Space]	Stack Manipulation
[Tab][Space]	Arithmetic
[Tab][Tab]	Heap Access
[LineFeed]	Flow Control
[Tab][LineFeed]	I/O

Each IMP is followed by one operation defined for that IMP, and a parameter if needed. The list of operations supported in Whitespace is:[1]

IMP	Command	Parameter	Meaning
[Space]	[Space]	Number	Push the number onto the stack
[Space]	[LF][Space]	-	Duplicate the top item on the stack
[Space]	[Tab][Space]	Number	Copy the nth item on the stack (given by the argument) onto the top of the stack
[Space]	[LF][Tab]	-	Swap the top two items on the stack
[Space]	[LF][LF]	-	Discard the top item on the stack
[Space]	[Tab][LF]	Number	Slide n items off the stack, keeping the top item
[Tab][Space]	[Space][Space]	-	Addition
[Tab][Space]	[Space][Tab]	-	Subtraction
[Tab][Space]	[Space][LF]	-	Multiplication
[Tab][Space]	[Tab][Space]	-	Integer Division
[Tab][Space]	[Tab][Tab]	-	Modulo
[Tab][Tab]	[Space]	-	Store in heap
[Tab][Tab]	[Tab]	-	Retrieve from heap
[LF]	[Space][Space]	Label	Mark a location in the program
[LF]	[Space][Tab]	Label	Call a subroutine
[LF]	[Space][LF]	Label	Jump to a label
[LF]	[Tab][Space]	Label	Jump to a label if the top of the stack is zero
[LF]	[Tab][Tab]	Label	Jump to a label if the top of the stack is negative
[LF]	[Tab][LF]	-	End a subroutine and transfer control back to the caller
[LF]	[LF][LF]	-	End the program
[Tab][LF]	[Space][Space]	-	Output the character at the top of the stack
[Tab][LF]	[Space][Tab]	-	Output the number at the top of the stack
[Tab][LF]	[Tab][Space]	-	Read a character and place it in the location given by the top of the stack
[Tab][LF]	[Tab][Tab]	-	Read a number and place it in the location given by the top of the stack

The "copy" and "slide" operations were added in Whitespace 0.3 and may not be supported by all implementations.[1]

Numbers

Numbers are composed of spaces (0) and tabs (1), and they are terminated by a linefeed. The first space/tab in the number represents the sign of the number, if it's a space the number is positive, if it's a tab the number is negative. The rest of the trailing spaces and tabs represent the rest of the binary number.

Examples

- space-tab-space-space-tab-space-tab-tab-linefeed (STSSTSTTL) represents the binary number 1001011 (positive, because the number starts with a space)
  - 75 in decimal.
- tab-tab-tab-space-space-tab-space-linefeed (TTTSSTSL) represents the binary number 110010 (negative, because the number starts with a tab)
  - −50 in decimal.

Labels

Labels are simply [LF] terminated lists of spaces and tabs. There is only one global namespace so all labels must be unique.[1]

Sample code

The following is a commented Whitespace program that simply prints "Hello, world!", where each Space, Tab, or Linefeed character is preceded by the identifying comment "S", "T", or "L", respectively:

```
S S S T	S S T	S S S L:Push_+1001000=72='H'_onto_the_stack
T	L
S S :Output_'H';_S S S T	T	S S T	S T	
L:Push_+1100101=101='e'_onto_the_stack
T	L
S S :Output_'e';_S S S T	T	S T	T	S S L:+1101100=108='l'
T	L
S S S S S T	T	S T	T	S S L:+1101100=108='l'
T	L
S S S S S T	T	S T	T	T	T	L:+1101111=111='o'
T	L
S S S S S T	S T	T	S S L:+101100=44=','
T	L
S S S S S T	S S S S S L:+100000=32=Space
T	L
S S S S S T	T	T	S T	T	T	L:+1110111=119='w'
T	L
S S S S S T	T	S T	T	T	T	L:+1101111=111='o'
T	L
S S S S S T	T	T	S S T	S L:+1110010=114='r'
T	L
S S S S S T	T	S T	T	S S L:+1101100=108='l'
T	L
S S S S S T	T	S S T	S S L=+1100100=100='d'
T	L
S S S S S T	S S S S T	L:+100001=33='!'
T	L
S S :Output_'!';_L
L
L:End_the_program
```

Note that when Whitespace source code is displayed in some browsers, the horizontal spacing produced by a tab character is not fixed, but depends on its location in the text relative to the next horizontal tab stop. Depending on the software, tab characters may also get replaced by the corresponding variable number of space characters.

See also

- Polyglot, a program valid in more than one language
- Steganography, the technique of concealing a message within another message
- Off-side rule languages, where blocks are expressed by whitespace indentation
  - Python, the best-known example of a language with syntactically significant whitespace
- Esoteric programming languages
  - Brainfuck, which consists of only eight simple commands and an instruction pointer
  - INTERCAL, the "Compiler Language With No Pronounceable Acronym"
  - LOLCODE, which is patterned after a series of Internet memes
  - Malbolge, which is specifically designed to be nearly impossible to program in

References

1. "Whitespace" (https://web.archive.org/web/20150618184706/http://compsoc.dur.ac.uk/whitespace/tutorial.php). *Compsoc*. Archived from the original (http://compsoc.dur.ac.uk/whitespace/tutorial.php) on 18 June 2015. Retrieved 8 December 2015.
2. Parker, Matt (2019). *Humble Pi: a comedy of maths errors*. Allen Lane. p. 21. ISBN 978-0-241-36023-1.
3. Timothy (1 April 2003). "New Whitespace-Only Programming Language" (http://developers.slashdot.org/story/03/04/01/0332202/New-Whitespace-Only-Programming-Language). *Slashdot*. Retrieved 23 January 2014.
4. Stroustrup, Bjarne. "Generalizing Overloading for C++2000" (http://www.stroustrup.com/whitespace98.pdf) (PDF). Florham Park, NJ, USA: AT&T Labs. Retrieved 23 January 2014.

External links

- Official website (https://web.archive.org/web/20150618184706/http://compsoc.dur.ac.uk/whitespace/tutorial.php)
- Release announcement on Slashdot (http://developers.slashdot.org/article.pl?sid=03/04/01/0332202)
- The Whitespace Corpus (https://github.com/wspace/corpus) A collection of interpreters, compilers, and programs for Whitespace
- Collection of Whitespace interpreters in various script languages (https://github.com/hostilefork/whitespacers/)
- Acme::Bleach (http://metacpan.org/module/Acme::Bleach) A Perl module that rewrites the body of your module to a whitespace-only encoding ("for *really* clean programs").

Retrieved from "https://en.wikipedia.org/w/index.php?title=Whitespace_(programming_language)&oldid=1161503609"