Polyglots

Julian Fietkau

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

ANSI C

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

ANSI C PHP

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

ANSI C PHP bash

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

ANSI C PHP bash

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

```
#define a /*
#<?php
echo "\010Hello, world!\n"// 2> /dev/null > /dev/null \ ;
// 2> /dev/null; x=a;
$x=5 // 2> /dev/null \ ;
if (($x))
// 2> /dev/null; then
return 0;
// 2> /dev/null; fi
#define e ?>
#define b */
#include <stdio.h>
#define main() int main()
#define printf printf(
#define true )
#define function
function main()
{
printf "Hello, world!\n"true/* 2> /dev/null | grep -v true*/;
return 0;
}
#define c /*
main
#*/
```

COBOL

COBOL, Pascal

COBOL, Pascal, Fortran

COBOL, Pascal, Fortran, C

COBOL, Pascal, Fortran, C, PostScript

COBOL, Pascal, Fortran, C,
PostScript, sh

COBOL, Pascal, Fortran, C,
PostScript, sh, Perl

COBOL, Pascal, Fortran, C,
PostScript, sh, Perl, x86 ASM

```
                                                                      (*O/*_/
Cu  #%* )pop mark/CuG 4 def/# 2 def%%%%@@P[TX---P\P_SXPY!Ex(mx2ex("SX!Ex4P)Ex=
CuG #%*                                                              *+Ex=
CuG #%*--------------------------------------------------------------*+Ex=
CuG #%*   POLYGLOT - a program in eight languages      15 February 1991  *+Ex=
CuG #%*   10th Anniversary Edition                       1 December 2001  *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*   Written by Kevin Bungard, Peter Lisle, and Chris Tham         *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*   Polyglot suports the following languages:                     *+Ex=
CuG #%*     1. COBOL (ANSI)                                             *+Ex=
CuG #%*     2. Pascal (ISO)                                             *+Ex=
CuG #%*     3. Fortran (ANSI, f77)                                      *+Ex=
CuG #%*     4. C (ANSI-ish)                                             *+Ex=
CuG #%*     5. PostScript                                               *+Ex=
CuG #%*     6. Linux/Unix shell script (bash, sh, csh)                  *+Ex=
CuG #%*     7. x86 machine language (MS-DOS, Win32, Linux)              *+Ex=
CuG #%*     8. Perl (version 5)                                         *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*   Usage:                                                        *+Ex=
CuG #%*     1. Rename this file to polyglot.[cob|pas|f77|c|ps|sh|com|pl] *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*     2. Compile and run with your favorite compiler and operating *+Ex=
CuG #%*        system.                                                  *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*   Notes:                                                        *+Ex=
CuG #%*     1. We have attempted to use only standard language features. *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*     2. Without the -traditional flag gcc will issue a warning.  *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*     3. When transferring from Unix to DOS make sure that a LF   *+Ex=
```

```
CuG #%*        is correctly translated into a CR/LF.                    *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*     4. Keep the blank lines at the start of the program. They   *+Ex=
CuG #%*        are important.                                           *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*     5. This text is a comment block in all eight languages.     *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*   Please mail any comments, corrections or additions to         *+Ex=
CuG #%*   polyglot@ideology.com.au                                      *+Ex=
CuG #%*                                                                 *+Ex=
CuG #%*-----------------------------------------------------------------*QuZ=
CuG #%*                                                                 *+Ex=
CuG #%*!Mx)ExQX5ZZ5SSP5n*5X!)Ex+ExPQXH,B+ExP[-9A-9B(g?(gA'UTTER_XYZZXX!X *+
CuG #(*                                                                  *(
C   # */);                                                             /*(
C   # *)  program         polyglot (output);                           (*+
C   #     identification division.
C   #     program-id.     polyglot.
C   #
C   #     data            division.
C   #     procedure       division.
C   #
C   # * ))cleartomark   /Bookman-Demi findfont 36 scalefont setfont     (
C   # *                                                                 (
C   #
C   # *                   hello polyglots$
C   #     main.
C   #         perform
C  /# * ) 2>_$$; echo   "hello polyglots"; rm _$$; exit;
C   # * (
C   #
```

```
C     *0 ) unless print "hello polyglots\n"; __END__
               print
C              stop run.
     -*,                  'hello polyglots'
C
C          print.
C              display   "hello polyglots".                                  (
C     */   int i;                                                           /*
C     */   main () {                                                        /*
C     */       i=printf ("hello polyglots\n"); O= &i; return *O;            /*
C     *)                                                                    (*
C     *)   begin                                                           (*
C     *)       writeln  ('hello polyglots');                                (*
C     *)                                                                    (* )
C     * ) pop 60 360                                                       (
C     * ) pop moveto    (hello polyglots) show                             (
C     * ) pop showpage                                                     ((
C     *)
            end                                                           .(* )
C)pop%      program       polyglot.                                       *){*/}
```

zsh

zsh, bash

zsh, bash, sh

zsh, bash, sh, Ruby

zsh, bash, sh, Ruby, Perl

zsh, bash, sh, Ruby, Perl, tcl

zsh, bash, sh, Ruby, Perl,
tcl, make

zsh, bash, sh, Ruby, Perl,
tcl, make, C

zsh, bash, sh, Ruby, Perl,
tcl, make, C, C++

zsh, bash, sh, Ruby, Perl,
tcl, make, C, C++, Haskell

zsh, bash, sh, Ruby, Perl, tcl, make, C, C++, Haskell, Brainfuck

zsh, bash, sh, Ruby, Perl,
tcl, make, C, C++, Haskell,
Brainfuck, Python

zsh, bash, sh, Ruby, Perl,
tcl, make, C, C++, Haskell,
Brainfuck, Python, HTML

zsh, bash, sh, Ruby, Perl, tcl, make, C, C++, Haskell, Brainfuck, Python, HTML, JavaScript

zsh, bash, sh, Ruby, Perl,
tcl, make, C, C++, Haskell,
Brainfuck, Python, HTML,
JavaScript, Perl 6

```
 # /* [   <!-- */ include <stdio.h> /*    \
#{\
`""""true \\#{"\n#"};                  \
\
if [ -n "$ZSH_VERSION" ]; then                  \
\
    echo exec echo I\'m a zsh script.; \
\
elif [ -n "$BASH_VERSION" ]; then           \
\
    echo exec echo I\'m a bash script.; \
else  \
    echo exec echo    I\'m    a sh    script.;        \
fi`;  #\
BEGIN{print"I'm a ", 0 ? "Ruby"   :"Perl",    " program.\n";  exit; }
#\
%q~

set dummy =0; puts [list "I'm"    "a" "tcl"   "script."]; exit

all: ; @echo "I'm a Makefile."               \
#*/
/*: */ enum {a, b};                  \
\
static int c99(void) {

 #ifndef __cplusplus /* bah */

unused1: if ((enum {b, a})0)            \
(void)0;
 #endif
```

```
unused2:    return a;      \
}     \
static int trigraphs(void) {                \
\
    return sizeof "??!"   ==  2;        \
}     \
char X;                           \
\
int main(void) {                      \
\
    struct X  {             \
\
      char    a[2];       \
};\
    if (sizeof(X) !=  1) {            \
\
printf("I'm a C++ program (trigraphs %sabled).\n",                \
\
     trigraphs()  ? "en"  : "dis");\
\
}else if (1//**/2


)unused3 : { ; \
        printf("I'm a C program (C%s, trigraphs %sabled).\n", \
               c99() ? "89 with // comments" : "99", \
               trigraphs() ? "en" : "dis"); \
    } else { \
        printf("I'm a C program (C89, trigraphs %sabled).\n", \
               trigraphs() ? "en" : "dis"); \
    } \
    return 0; \
```

```
} /*
 # \

> main :: IO () -- -- \
> main = putStr "I'm a Literate Haskell program.\n"

 # \
]>++++++++[<++++++++++>-]<+.>>++++[<++++++++++>-]<-.[-]>++++++++++ \
[<++++++++++>-]<-.>>++++[<++++++++>-]<.>>++++++++++[<++++++++++> \
-]<- - -.<.>+.->>++++++++++[<++++++++++>-]<++++.<.>>++++++++++[ \
<++++++++++>-]<+++++.<<<<+.->>>>- - -.<+++.- - -<++.- ->>>>>++++ \
+++++[<++++++++++>-]<- - -.<<<<<.<+++.>>>.<<<-.- ->>>>+.<.<.<<.> \
++++++++++++++.[-]++++++++++"""`
 # \
print "I'm a Python program."; """[-][--><html><head>
<!--:--><title>I'm a HTML page</title></head><body>
<!--:--><h1>I'm a <marquee><blink>horrible HTML</blink></marquee> page</h1>
<!--:--><script language="JavaScript">
<!--: # \
setTimeout( // \
   function () { // \
      document.body.innerHTML = "<h1>I'm a javascript-generated HTML page</h1>";
  // \
   }, 10000); // \
//-->
</script><!--: \
</body></html><!-- } # \
say "I'm a Perl6 program", try { " ($?PUGS_VERSION)" } // "", "."; # """ # */
 #define FOO ]-->~
```

When you stare into the abyss, the abyss stares back at you.

-Friedrich Nietzsche