385,823 Members | 2,800 Online

Join Now | login

BYTES IT COMMUNITY

Questions | Articles | Browse Topics | Latest | Top Members | FAQ

+ Ask Question

home > topics > c / c++ > questions > how to use eof in c

Search

How to use EOF in C

Ramesh

P: n/a

Hi friends,

I want to learn how to use EOF in C.Please help me.

Jul 11 '07 #1

Post Reply

Share this Question

29 Replies

Question stats

- viewed: 32016
- replies: 29
- date asked: Jul 11 '07

Follow this discussion

Similar topics

- Observations regarding redirection
- Back button
- select from drop list to fill table column with text -- HOW ?
- Image Resize does not work
- MYSQL, PHP and Bind variables in SQL
- ~ Help - counter script ~
- If - ElseIf - SQL Select... ASP YES.. PHP NO ???
- PHP & Authorize.net w/o cURL?
- Experience with PHP & Oracle 9.2.0/OCI8

sumedh

P: n/a

On Jul 11, 10:41 am, Ramesh <chary...@gmail.comwrote:

> Hi friends,
> I want to learn how to use EOF in C.Please help me.

C provides u with EOF denoting end of file. The following code may help you.

```
int get_line(char line[], int max)
{
int nch = 0;
int ch;
max = max - 1; //initally max is pointing to \0
//So we need to get the char before \0

while((ch = getchar()) != EOF) //getchar returns char from
{
if(ch == '\n')
break;

if(nch < max)
{
line[nch] = ch;
nch = nch + 1;
}
}

if(ch == EOF && nch == 0)
return EOF;

line[nch] = '\0';
return nch;
}

#include <stdio.h>
extern int get_line(char [], int);
main()
{
char l[200];

while(getline(l, 200) != EOF)
printf("line read\"%s\"\n", l);

return 0;
}
```

Jul 11 '07 #2

santosh

P: n/a

sumedh wrote:

> On Jul 11, 10:41 am, Ramesh <chary...@gmail.comwrote:
>
> > Hi friends,
> > I want to learn how to use EOF in C.Please help me.
>
> C provides u with EOF denoting end of file.

To be precise, EOF is a macro defined in stdio.h that yields an integer constant. It is used as the return value by many of C's I/O functions when they encounter an end-of-file condition or an error. The pair of functions feof and ferror are typically used to find out

The pair of functions feof and ferror are typically used to find out which of the two it is.

> The following code may
> help you.
> ```
> int get_line(char line[], int max)
> {
> int nch = 0;
> int ch;
> max = max - 1; //initally max is pointing to \0
> //So we need to get the char before \0
> ```

This type of comment is only supported as of C99. Also it tends to break when used in posts to Usenet.

> ```
> while((ch = getchar()) != EOF) //getchar returns char from
> {
> if(ch == '\n')
> break;
>
> if(nch < max)
> {
> line[nch] = ch ;
> nch = nch + 1;
> }
> }
>
> if(ch == EOF && nch == 0)
> return EOF;
>
> line[nch] = '\0';
> return nch;
> }
>
> #include <stdio.h>
> extern int get_line(char [], int);
> main()
> ```

Declare main to explicitly return an int. And if you don't intend to access the command line you can use void to indicate that it does not take any parameters.

```
int main(void) {
```

> ```
> {
> char l[200];
>
> while(getline(l, 200) != EOF)
> ```

Where is getline defined?

> ```
> printf("line read\"%s\"\n", l);
>
> return 0;
> }
> ```

Jul 11 '07 #3

Keith Thompson

P: n/a

Ramesh <chary237@gmail.comwrites:

> I want to learn how to use EOF in C.Please help me.

Any decent C textbook will explain this. I recommend Kernighan &
Ritchie's _The C Programming Language_, 2nd Edition, commonly known as
K&R2.

But I think you're asking the wrong question. Rather than asking how
to use EOF, you should probably be trying to learn how to read and
process input. The use of EOF is just a part of that.

--
Keith Thompson (The_Other_Keith) kst-u@mib.org <http://www.ghoti.net/~kst>
San Diego Supercomputer Center <* <http://users.sdsc.edu/~kst>
"We must do something. This is something. Therefore, we must do this."
-- Antony Jay and Jonathan Lynn, "Yes Minister"

Jul 11 '07 #4

Kelsey Bjarnason

P: n/a

[snips]

On Tue, 10 Jul 2007 22:51:21 -0700, sumedh wrote:

> max = max - 1;

If the user does something silly, such as passing in 0, or a negative
value, doesn't this end up doing bad things to your loop?

> while((ch = getchar()) != EOF) //getchar returns char from
> {
> if(ch == '\n')
> break;
>
> if(nch < max)
> {
> line[nch] = ch;
> nch = nch + 1;
> }
> }
>
> if(ch == EOF && nch == 0)
> return EOF;

Hmm. So if we get EOF, we simply discard the data read to that point,
even though it may be perfectly good data? Why not slap a \0 on the
sucker to turn it into a proper string?

Jul 11 '07 #5

Malcolm McLean

P: n/a

"Ramesh" <chary237@gmail.comwrote in message
news:1184132490.368531.189250@o11g2000prd.googlegr oups.com...

> Hi friends,
> I want to learn how to use EOF in C.Please help me.

the function fgetc() will return an integer, not a character as you may have
imagined, in the range 0-255 except some very odd systems that don't use 8
bit bytes. It returns -1 or EOF to indicate the end of input. That's why it
returns an integer instead of a char. Othewise there would be one value that
could not be represented

could not be represented.

In practise you always need this construct

```
int ch;
FILE *fp;

while( (ch = fgetc(fp)) != EOF)
{
/* process characters here */
printf("val %d char %c\n", ch, ch);
}
```

--
Free games and programming goodies.
http://www.personal.leeds.ac.uk/~bgy1mm

Jul 11 '07 #6

santosh

P: n/a

Malcolm McLean wrote:

> "Ramesh" <chary237@gmail.comwrote in message
> news:1184132490.368531.189250@o11g2000prd.googlegr oups.com...
>
> > Hi friends,
> > I want to learn how to use EOF in C.Please help me.
>
> the function fgetc() will return an integer, not a character as you may have
> imagined, in the range 0-255 except some very odd systems that don't
> use 8
> bit bytes. It returns -1 or EOF to indicate the end of input.

EOF needn't equal -1.

Jul 11 '07 #7

Walter Roberson

P: n/a

In article <ptednd_3mJlB1AjbnZ2dnUVZ8vWdnZ2d@bt.com>,
Malcolm McLean <regniztar@btinternet.comwrote:

> >the function fgetc() will return an integer, not a character as you may
> have
> imagined, in the range 0-255 except some very odd systems that don't
> use 8
> bit bytes. It returns -1 or EOF to indicate the end of input.

To be more precise, it returns some specific (implementation-defined) negative value to indicate the end of input, and the particular implementation negative value is available by using the macro EOF -- no matter what -particular- value the implementation uses, if you refer to EOF then you will get the right value for that implementation. But the implementation value is not necessarily -1, so do not hard-code comparisons to -1. You could code checks for < 0 if you wanted though, as the only negative value allowed to be returned by fgetc() is whatever the system is using for EOF.

> >In practise you always need this construct

> >int ch;

Annotating slightly for the OP: notice Malcolm used 'int'
as the type, not 'char'.

> >FILE *fp;

> >while((ch = fgetc(fp)) != EOF)

Another annotation for the OP: notice that the assignment
to ch is within () . If you were to leave out the () and code

while (ch = fgetc(fp) != EOF) /* WRONG */

then this would be wrong because C would parse this as

while (ch = (fgetc(fp) != EOF))

that is, ch would get assigned the result of the -comparison-
rather than the character that was read in.

> >{
> /* process characters here */
> printf("val %d char %c\n", ch, ch);
> }

--
Programming is what happens while you're busy making other plans.

Jul 11 '07 #8

santosh

P: n/a

Kelsey Bjarnason wrote:

> [snips]
>
> On Tue, 10 Jul 2007 22:51:21 -0700, sumedh wrote:
>
> > max = max - 1;
>
> If the user does something silly, such as passing in 0, or a negative
> value, doesn't this end up doing bad things to your loop?
>
> > while((ch = getchar()) != EOF) //getchar returns char from
> > {
> > if(ch == '\n')
> > break;
> >
> > if(nch < max)
> > {
> > line[nch] = ch ;
> > nch = nch + 1;
> > }
> > }
> >
> > if(ch == EOF && nch == 0)
> > return EOF;

> Hmm. So if we get EOF, we simply discard the data read to that point,
> even though it may be perfectly good data?

Notice the other half of the expression? Anyway, regardless the code is broken since if nch is not less than max, then additional input is simply discarded.

> Why not slap a \0 on the
> sucker to turn it into a proper string?

An '\0' by itself is not a string.

Jul 11 '07 #9

Keith Thompson

P: n/a

"Malcolm McLean" <regniztar@btinternet.comwrites:

> "Ramesh" <chary237@gmail.comwrote in message
> news:1184132490.368531.189250@o11g2000prd.googlegr oups.com...
>
> > >Hi friends,
> > I want to learn how to use EOF in C.Please help me.
>
> the function fgetc() will return an integer, not a character as you
> may have imagined, in the range 0-255 except some very odd systems
> that don't use 8 bit bytes.

It returns an int (one of several integer types), not a char (also one of several integer types, as well as being one of several character types).

"int" and "integer" mean very different things, as do "char" and "character".

> It returns -1 or EOF to indicate the end
> of input. That's why it returns an integer instead of a char. Othewise
> there would be one value that could not be represented.

EOF is typically defined as (-1), but the only requirement is that it's of type int with a negative value. I actually don't know of any implementation where EOF has a value other than -1; nevertheless I would consider something like

while((ch = fgetc(fp)) != -1) /* ... */

to be badly broken (even if it happens to work).

[snip]

--
Keith Thompson (The_Other_Keith) kst-u@mib.org <http://www.ghoti.net/~kst>
San Diego Supercomputer Center <* <http://users.sdsc.edu/~kst>
"We must do something. This is something. Therefore, we must do this."
-- Antony Jay and Jonathan Lynn, "Yes Minister"

Jul 11 '07 #10

Richard Heathfield

P: n/a

santosh said:

<snip>

> An '\0' by itself is not a string.

The Standard says: "A string is a contiguous sequence of characters terminated by and including the first null character."

The Standard mentions "empty strings" on several occasions. An empty string is still a string.

--
Richard Heathfield <http://www.cpax.org.uk>
Email: -www. +rjh@
Google users: <http://www.cpax.org.uk/prg/writings/googly.php>
"Usenet is a strange place" - dmr 29 July 1999

Jul 11 '07 #11

Keith Thompson

P: n/a

santosh <santosh.k83@gmail.comwrites:

> Kelsey Bjarnason wrote:

[...]

> > > Why not slap a \0 on the
> > sucker to turn it into a proper string?
>
> An '\0' by itself is not a string.

A sequence consisting of a single '\0' character is a string, specifically an empty string. (I'm ignoring whatever context led up to this.)

(Just out of curiosity, how do you pronounce '\0' so that it's "An '\0'" rather than "A '\0'"?)

--
Keith Thompson (The_Other_Keith) kst-u@mib.org <http://www.ghoti.net/~kst>
San Diego Supercomputer Center <*> <http://users.sdsc.edu/~kst>
"We must do something. This is something. Therefore, we must do this."
-- Antony Jay and Jonathan Lynn, "Yes Minister"

Jul 11 '07 #12

Richard

P: n/a

santosh <santosh.k83@gmail.comwrites:

> Kelsey Bjarnason wrote:
>
> > >[snips]
> >
> > On Tue, 10 Jul 2007 22:51:21 -0700, sumedh wrote:
> >
> > > max = max - 1;
> >
> > If the user does something silly, such as passing in 0, or a negative value, doesn't this end up doing bad things to your loop?

> > > ```
> > > while((ch = getchar()) != EOF) //getchar returns char from
> > > {
> > > if(ch == '\n')
> > > break;
> > >
> > > if(nch < max)
> > > {
> > > line[nch] = ch ;
> > > nch = nch + 1;
> > > }
> > > }
> > >
> > > if(ch == EOF && nch == 0)
> > > return EOF;
> > > ```
> >
> > Hmm. So if we get EOF, we simply discard the data read to that point,
> > even though it may be perfectly good data?
>
> Notice the other half of the expression? Anyway, regardless the code is broken since if nch is not less than max, then additional input is simply discarded.
>
> > > Why not slap a \0 on the
> > sucker to turn it into a proper string?
>
> An '\0' by itself is not a string.

Of course it is.

If you have a character pointer pointing to that then the pointer is still pointing to a character string.

Jul 11 '07 #13

santosh

P: n/a

Keith Thompson wrote:

> santosh <santosh.k83@gmail.comwrites:
>
> > Kelsey Bjarnason wrote:
>
> [...]
>
> > > Why not slap a \0 on the
> > > sucker to turn it into a proper string?
> >
> > An '\0' by itself is not a string.
>
> A sequence consisting of a single '\0' character is a string, specifically an empty string. (I'm ignoring whatever context led up to this.)

Okay, you're correct, (also Richard Heathfield and Richard). I must consult the Standard more before posting.

> (Just out of curiosity, how do you pronounce '\0' so that it's

> "An '\0'" rather than "A '\0'"?)

I don't. It's a mistake obviously.

Jul 12 '07 #14

Spiros Bousbouras

P: n/a

On 11 Jul, 06:51, sumedh <excuseme2...@gmail.comwrote:

> On Jul 11, 10:41 am, Ramesh <chary...@gmail.comwrote:
>
> > Hi friends,
> > I want to learn how to use EOF in C.Please help me.
>
> C provides u with EOF denoting end of file. The following code may
> help you.
>
> ```
> int get_line(char line[], int max)
> {
> int nch = 0;
> int ch;
> max = max - 1; //initally max is pointing to \0
> //So we need to get the char before \0
>
> while((ch = getchar()) != EOF) //getchar returns char from
> {
> if(ch == '\n')
> break;
>
> if(nch < max)
> {
> line[nch] = ch ;
> nch = nch + 1;
> }
> }
>
> if(ch == EOF && nch == 0)
> return EOF;
>
> line[nch] = '\0';
> return nch;
>
> }
>
> #include <stdio.h>
> extern int get_line(char [], int);
> main()
> {
> char l[200];
>
> while(getline(l, 200) != EOF)
> printf("line read\"%s\"\n", l);
>
> return 0;
>
> }
> ```

This will not compile for 2 reasons:
1) You define get_line but you call getline
(mentioned already).

2) You use EOF before including <stdio.h>
If I remember correctly macros are defined from
the point they appear until the end of file.

With the obvious corrections the code would compile,
work and shows some sort of usage for EOF. Others
have commented that it truncates the line read but
that doesn't mean that it's broken. On several occasions
I've needed the first n characters of each line of a file
for fixed n.

Jul 12 '07 #15

Malcolm McLean

P: n/a

"Keith Thompson" <kst-u@mib.orgwrote in message
news:ln4pkaeg65.fsf@nuthaus.mib.org...

> santosh <santosh.k83@gmail.comwrites:
>
> > >Kelsey Bjarnason wrote:
>
> [...]
>
> > > >> Why not slap a \0 on the
> > > sucker to turn it into a proper string?
> >
> > An '\0' by itself is not a string.
>
> A sequence consisting of a single '\0' character is a string,
> specifically an empty string. (I'm ignoring whatever context led up
> to this.)
>
> (Just out of curiosity, how do you pronounce '\0' so that it's
> "An '\0'" rather than "A '\0'"?)

"An ought". Or maybe "An ul". I've even heard "An ee-oh-ess".

--
Free games and programming goodies.
http://www.personal.leeds.ac.uk/~bgy1mm

Jul 12 '07 #16

Malcolm McLean

P: n/a

"Richard Heathfield" <rjh@see.sig.invalidwrote in message

> The Standard mentions "empty strings" on several occasions. An empty
> string is still a string.

A string of no sausages is a very different thing to no string of sausages.

--
Free games and programming goodies.
http://www.personal.leeds.ac.uk/~bgy1mm

Jul 12 '07 #17

Richard

P: n/a

"Malcolm McLean" <regniztar@btinternet.comwrites:

> "Richard Heathfield" <rjh@see.sig.invalidwrote in message
>
> > >The Standard mentions "empty strings" on several occasions. An
> > empty
> > string is still a string.
>
> A string of no sausages is a very different thing to no string of sausages.

There may be no sausages on it, but the string is still there.

Jul 12 '07 #18

Flash Gordon

P: n/a

Richard wrote, On 12/07/07 10:57:

> "Malcolm McLean" <regniztar@btinternet.comwrites:
>
> > >"Richard Heathfield" <rjh@see.sig.invalidwrote in message
> >
> > > >>The Standard mentions "empty strings" on several
> > > occasions. An empty
> > > string is still a string.
> >
> > A string of no sausages is a very different thing to no string of sausages.
>
> There may be no sausages on it, but the string is still there.

The standard even provides methods of expressing these things Malcolm acknowledges as different.

char *sausages = NULL; /* No string */

char *sausages == ""; /* Empty string */
--
Flash Gordon

Jul 12 '07 #19

Kelsey Bjarnason

P: n/a

On Wed, 11 Jul 2007 21:39:58 +0000, santosh wrote:

> Kelsey Bjarnason wrote:

> Notice the other half of the expression?

Grr. Damn this being human thing sucks. :)

> Anyway, regardless the code
> is broken since if nch is not less than max, then additional input is simply discarded.
>
> > > Why not slap a \0 on the

> > sucker to turn it into a proper string?
>
> An '\0' by itself is not a string.

Pass it to strlen, it'll work. Pass the results of whatever you've got
instead, it'll scream and die - if you're lucky.

Jul 12 '07 #20

Richard Heathfield

P: n/a

Flash Gordon said:

<snip>

> The standard even provides methods of expressing these things Malcolm acknowledges as different.
>
> char *sausages = NULL; /* No string */
> char *sausages == ""; /* Empty string */

/* ITYM syntax error */

--

Richard Heathfield <http://www.cpax.org.uk>
Email: -www. +rjh@
Google users: <http://www.cpax.org.uk/prg/writings/googly.php>
"Usenet is a strange place" - dmr 29 July 1999

Jul 12 '07 #21

Keith Thompson

P: n/a

Kelsey Bjarnason <kbjarnason@gmail.comwrites:

> On Wed, 11 Jul 2007 21:39:58 +0000, santosh wrote:

[...]

> > >An '\0' by itself is not a string.
>
> Pass it to strlen, it'll work.

strlen('\0') invokes undefined behavior.

I know that's not what you meant. If you pass the address of a
character with value '\0' to strlen(), it will return 0.

--
Keith Thompson (The_Other_Keith) kst-u@mib.org <http://www.ghoti.net/~kst>
San Diego Supercomputer Center <*> <http://users.sdsc.edu/~kst>
"We must do something. This is something. Therefore, we must do this."
-- Antony Jay and Jonathan Lynn, "Yes Minister"

Jul 12 '07 #22

Flash Gordon

P: n/a

Richard Heathfield wrote, On 12/07/07 21:10:

> Flash Gordon said:

> Flash Gordon said:
>
> <snip>
>
> > >The standard even provides methods of expressing these things
> > Malcolm
> > acknowledges as different.
> >
> > char *sausages = NULL; /* No string */
> > char *sausages == ""; /* Empty string */
>
> /* ITYM syntax error */

s/==/=/
--
Flash Gordon

Jul 12 '07 #23

Malcolm McLean

P: n/a

"Flash Gordon" <spam@flash-gordon.me.ukwrote in message
news:rjhim4xbhv.ln2@news.flash-gordon.me.uk...

> Richard Heathfield wrote, On 12/07/07 21:10:
>
> > >Flash Gordon said:
> >
> > <snip>
> >
> > > >>The standard even provides methods of expressing these
> > > things Malcolm
> > > acknowledges as different.
> > >
> > > char *sausages = NULL; /* No string */
> > > char *sausages == ""; /* Empty string */
> >
> > /* ITYM syntax error */
>
> s/==/=/
> --
> Flash Gordon

char O=0; /* sausage */

char *sausages = 0-0-0-0-0-0-0; /* no string */
char *sausage = &O; /* empty string */

--
Free games and programming goodies.
http://www.personal.leeds.ac.uk/~bgy1mm

Jul 12 '07 #24

Ben Pfaff

P: n/a

"Malcolm McLean" <regniztar@btinternet.comwrites:

> char O=0; /* sausage */
>
> char *sausages = 0-0-0-0-0-0-0; /* no string */

> char *sausage = &O; /* empty string */

char *confusing_sausage = &O-0-0-0-0-0-0-0; /* empty string */
--

```
char a[]="\n .CJacehknorstu";int putchar(int);int main(void){unsigned long b[]
={0x67dffdff,0x9aa9aa6a,0xa77ffda9,0x7da6aa6a,0xa6
7f6aaa,0xaa9aa9f6,0x11f6},*p
=b,i=24;for(;p+=!*p;*p/=4)switch(0[p]&3)case 0:{return 0;for(p--;i--;i--)case+
2:{i++;if(i)break;else default:continue;if(0)case 1:putchar(a[i&15]);break;}}}
```

Jul 12 '07 #25

Richard Bos

P: n/a

"Malcolm McLean" <regniztar@btinternet.comwrote:

> char O=0; /* sausage */
>
> char *sausages = 0-0-0-0-0-0-0; /* no string */
> char *sausage = &O; /* empty string */

# Constraints
# 1 The operand of the unary & operator shall be either a function
# designator, the result of a [] or unary * operator, or an lvalue
# that designates an object that is not a bit-field and is not
# declared with the register storage-class specifier.

A literal 0 is not any of those, so &0 violates this constraint. Did you perhaps mean "0" (or even &"0")?

Richard

Jul 13 '07 #26

P: n/a

Keith Thompson

rlb@hoekstra-uitgeverij.nl (Richard Bos) writes:

> "Malcolm McLean" <regniztar@btinternet.comwrote:
>
> > >char O=0; /* sausage */
> >
> > char *sausages = 0-0-0-0-0-0-0; /* no string */
> > char *sausage = &O; /* empty string */
>
> # Constraints
> # 1 The operand of the unary & operator shall be either a function
> # designator, the result of a [] or unary * operator, or an lvalue
> # that designates an object that is not a bit-field and is not
> # declared with the register storage-class specifier.
>
> A literal 0 is not any of those, so &0 violates this constraint. Did you
> perhaps mean "0" (or even &"0")?

Get a better font, or copy-and-paste the code and try to compile it. The initializer for 'sausage' is '&O', not '&0' (the letter, not the digit); 'O' was declared three lines earlier as an object of type char, initialized to zero.

--
Keith Thompson (The_Other_Keith) kst-u@mib.org <http://www.ghoti.net/~kst>
San Diego Supercomputer Center <*> <http://users.sdsc.edu/~kst>

"We must do something. This is something. Therefore, we must do this."
-- Antony Jay and Jonathan Lynn, "Yes Minister"

Jul 13 '07 #27

Kelsey Bjarnason

P: n/a

On Thu, 12 Jul 2007 13:59:20 -0700, Keith Thompson wrote:

> Kelsey Bjarnason <kbjarnason@gmail.comwrites:
>
>> >On Wed, 11 Jul 2007 21:39:58 +0000, santosh wrote:
>
> [...]
>
>> > >>An '\0' by itself is not a string.
>>
>> Pass it to strlen, it'll work.
>
> strlen('\0') invokes undefined behavior.

True.

> I know that's not what you meant. If you pass the address of a character with value '\0' to strlen(), it will return 0.

In the context used. '\0' "works". In the general case, it doesn't, for hopefully obvious reasons.

Jul 13 '07 #28

Malcolm McLean

P: n/a

"Keith Thompson" <kst-u@mib.orgwrote in message
news:lnlkdk94hv.fsf@nuthaus.mib.org...

> rlb@hoekstra-uitgeverij.nl (Richard Bos) writes:
>
>> >"Malcolm McLean" <regniztar@btinternet.comwrote:
>>
>> > >>char O=0; /* sausage */
>> >
>> > char *sausages = 0-0-0-0-0-0-0; /* no string */
>> > char *sausage = &O; /* empty string */
>>
>> # Constraints
>> # 1 The operand of the unary & operator shall be either a function
>> # designator, the result of a [] or unary * operator, or an lvalue
>> # that designates an object that is not a bit-field and is not
>> # declared with the register storage-class specifier.
>>
>> A literal 0 is not any of those, so &0 violates this constraint. Did you perhaps mean "0" (or even &"0")?
>
> Get a better font, or copy-and-paste the code and try to compile it. The initializer for 'sausage' is '&O', not '&0' (the letter, not the digit); 'O' was declared three lines earlier as an object of type char, initialized to zero.

It was a charred sausage.

--
Free games and programming goodies.
http://www.personal.leeds.ac.uk/~bgy1mm

Jul 14 '07 #29

P: n/a

Richard Bos

"Malcolm McLean" <regniztar@btinternet.comwrote:

> "Keith Thompson" <kst-u@mib.orgwrote in message
> news:lnlkdk94hv.fsf@nuthaus.mib.org...
>
> > rlb@hoekstra-uitgeverij.nl (Richard Bos) writes:
> >
> > > "Malcolm McLean" <regniztar@btinternet.comwrote:
> > > char O=0; /* sausage */
> > >
> > > char *sausages = 0-0-0-0-0-0-0; /* no string */
> > > char *sausage = &O; /* empty string */
> > >
> > > # Constraints
> > > # 1 The operand of the unary & operator shall be either a
> > > function
> > > # designator, the result of a [] or unary * operator, or an lvalue
> > > # that designates an object that is not a bit-field and is not
> > > # declared with the register storage-class specifier.
> > >
> > > A literal 0 is not any of those, so &0 violates this constraint.
> > > Did you
> > > perhaps mean "0" (or even &"0")?
> >
> > Get a better font, or copy-and-paste the code and try to compile it.
> > The initializer for 'sausage' is '&O', not '&0' (the letter, not the
> > digit); 'O' was declared three lines earlier as an object of type
> > char, initialized to zero.

Hangs head in shame

> It was a charred sausage.

Groans

Richard

Jul 16 '07 #30

This discussion thread is closed

Replies have been disabled for this discussion. Start new discussion

Browse more C / C++ Questions on Bytes

About Bytes | Advertise on Bytes | Contact Us
Sitemap | C / C++ Answers Sitemap | C / C++ Insights Sitemap

Follow us to get the Latest Bytes Updates