Subject: Re: IDLWAVE 4.7/Tutorial
Posted by John-David T. Smith on Fri, 08 Dec 2000 17:25:22 GMT
View Forum Message <> Reply to Message

Alright, now this is some exciting news. Take a look online at this tutorial too, for more friendly reading:

http://www.strw.LeidenUniv.nl/~dominik/Tools/idlwave/idlwave .html#SEC3

P.S. For linux users: If you'd like to make use of that silly Windows key just outside your "Alt" keys (which in Windows is used to mail a copy of your last 100 browsing locations to bill@gates.com) you can easily do this. For me, adding the following to ~/.Xmodmap worked:

keycode 115 = Hyper_L
keycode 116 = Hyper_R
clear Mod5
add Mod5 = Hyper_L Hyper_R

(and yes david, you can read all about this notation: man xmodmap)

If these keycodes aren't correct, run "xev", position input in little white window, and press the Gates-was-here key. You'll see something like:

blah blah, keycode 115 (keysym 0x0, NoSymbol), blah blah

and similar for the other key. From this you can get the appropriate "keycode" above, but 115,116 will probably work. You can see how to use this method for any other keys too (you can make a "Super" modifier too if you have a spare).

Now you can have lots of fun with:

H-down,H-up,H-t,H-l,H-w,H-e,H-x,H-@,H-?,H-p,H-q,H-z,H-y,H-r,
H-h,H-m,H-o,H-u,H-k,H-n,H-s,H-a,H-d,H-i

Ahh the joy.

JD

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by Jack Saba on Fri, 08 Dec 2000 18:26:25 GMT
View Forum Message <> Reply to Message

Carsten Dominik wrote:
>

> Hi, I have release IDLWAVE 4.7. Sorry that this happens so quickly
> after 4.6, but the recent discussion here has prompted a new way of
> assigning keys to debugging commands which I would like to get out
> now.
>
> Also, JD and I have been cooking up a tutorial which was requested by
> several contributions here. While it may not quite be simple enough
> for David ;-), I hope it will be OK for most people who have used
> Emacs before. Feedback on the Tutorial is welcome. The Tutorial
> should work with 4.6 except in one or two details. The keybinding
> methods described only work with 4.7.
>
> - Carsten
>

OK, Carsten, here's some feedback from a would-be user.

I've tried IDLWave in the past, and always ended up disabling it because I was
unable to customize it to my satisfaction. It looks like a very useful tool.
This turorial certainly helps, but I'm still unable to figure out how to
control some of the settings.

Basically, I want complete control of text layout in the buffer.

1. I enter

```
FOR i=0,n
  stuff
ENDFOR
```

When I hit the <CR>, IDLWave reformats this by indenting the ENDFOR 3 spaces:

```
FOR i=0,n
  stuff
  ENDFOR
```

How do I prevent this reformatting?

2. How do I make tabs in IDLwave mode work the way they do in text mode?
Currently, the first tab moves the cursor under the first character on the last
line, but subsequent tabs fall into a black hole.

FWIW, here are the settings I've come up with so far:

```
(setq indent-line-function 'indent-relative)  ; I thought this would give me the
        ; same behavior I get in text-mode.
```

```
(setq idlwave-hanging-indent nil)
(setq idlwave-end-offset 0)
(setq idlwave-main-block-indent 0)
(setq idlwave-code-comment "klajsdfjpawe")  ;   This is an attempt to disable
                                            ;   this function.
(setq idlwave-no-change-comment ";...")
(setq idlwave-continuation-indent 0)
(setq idlwave-block-indent 0)
```

Jack Saba
<jack.saba@gsfc.nasa.gov>

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by John-David T. Smith on Fri, 08 Dec 2000 19:16:09 GMT
View Forum Message <> Reply to Message

Jack Saba wrote:
> OK, Carsten, here's some feedback from a would-be user.
>
> I've tried IDLWave in the past, and always ended up disabling it because I was
> unable to customize it to my satisfaction. It looks like a very useful tool.
> This turorial certainly helps, but I'm still unable to figure out how to
> control some of the settings.
>
> Basically, I want complete control of text layout in the buffer.
>
> 1. I enter
>
>   FOR i=0,n
>   stuff
>   ENDFOR
>
>   When I hit the <CR>, IDLWave reformats this by indenting the ENDFOR 3 spaces:
>
>   FOR i=0,n
>   stuff
>   ENDFOR
>
>   How do I prevent this reformatting?
>
> 2. How do I make tabs in IDLwave mode work the way they do in text mode?
> Currently, the first tab moves the cursor under the first character on the last
> line, but subsequent tabs fall into a black hole.

Think of <TAB> as more of a code cleaner-upper than a tab insertion
mechanism. Almost all programming modes do it this way. Once a line of

code is cleaned, cleaning it again will make no change. You *really* shouldn't be lining code up by hand, it's far too tedious.

If you would just like code to line up as in your first example, try:

```
idlwave-block-indent 3          ; Indentation settings
idlwave-end-offset -3
```

This will line the "ENDFOR" up with the "FOR".

Here's a little pictogram to help with the indents:

```
pro foo
<M>ThisPro, a
<M>for i=1,10 do begin
<M><B>print, 'yay' + $
<M><B><C>AnotherVar
<M><B><E>endfor
end
```

<M> = Main block indent
<B> = Block indent
<C> = Continuation extra indent
<E> = End offset

in particular: <M>=3, <B>=3, <C>=1, <E>=-3 would produce:

```
pro foo
   ThisPro, a
   for i=1,10 do begin
      print, 'yay' + $
       AnotherVar
   endfor
end
```

see how <E> cancels <B> in the endfor line? Pretty simple. Your end offset is leaving things hanging.

<M>=0, <B>=5, <C>=3, <E>=-3 would make it:

```
pro foo
ThisPro, a
for i=1,10 do begin
     print, 'yay' + $
        AnotherVar
  endfor
end
```

Oooh, ugly. But any degree of ugliness is tolerated.

Currently, <RET> indents the line after inserting the newline. If you would prefer "<RET>" not to indent the line for you, you can simply set:

(local-set-key "\r" 'newline)

in your idlwave-mode-hook. But I can't see why anyone would really want this... you'll just end up hitting <TAB> after <RET> everytime. But give it a try if you like.

If you don't want <TAB> to behave specially at all (by the way, you can always use C-<TAB> to get a "real" tab in any case), you can simply:

(local-set-key "\t" 'idlwave-hard-tab)

but this is getting really silly. The point of this indentation is to pick a style you like, and let IDLWAVE use it everywhere. It makes your code so much more maintainable. It would be nice if we could all agree on a code indentation scheme and commenting style, but *at least* a personally enforced standard is necessary.

Also, if you ever find that you don't like what your customizations have done, disable them. Carsten has worked hard to make the default settings very useable.

JD

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by Jeff Guerber on Fri, 08 Dec 2000 20:56:49 GMT
View Forum Message <> Reply to Message

Jack, I can take a quick stab at your questions.

On Fri, 8 Dec 2000, Jack Saba wrote:

> 1. I enter
>
>   FOR i=0,n
>   stuff
>   ENDFOR
>
>   When I hit the <CR>, IDLWave reformats this by indenting the ENDFOR
> 3 spaces:
>
>   FOR i=0,n
>   stuff

>   ENDFOR
>
>   How do I prevent this reformatting?

   (You did mean "FOR i=0,n DO BEGIN", didn't you?) You say later on that
you're doing:

> (setq idlwave-end-offset 0)
> (setq idlwave-block-indent 0)

Emacs offsets are always(?) relative to the previous line; since
idlwave-end-offset is 0, the indentation for ENDFOR goes to the same
column as "stuff". If you make these -3 and 3 respectively (the defaults,
at least in idlwave 4.5, are -4 and 4), it should indent, automatically,
to the columns it looks like you want it to (including "stuff").

> 2. How do I make tabs in IDLwave mode work the way they do in text
> mode? Currently, the first tab moves the cursor under the first
> character on the last line, but subsequent tabs fall into a black
> hole.

   I'm not sure how to change this, but it looks like idlwave makes
indent-line-function buffer-local and then sets it to its own function,
'idlwave-indent-and-action. So, your setq isn't making any difference.

> FWIW, here are the settings I've come up with so far:
>
>
> (setq indent-line-function 'indent-relative) ; I thought this would give me the
>   ; same behavior I get in text-mode.
> (setq idlwave-hanging-indent nil)
> (setq idlwave-end-offset 0)
> (setq idlwave-main-block-indent 0)
> (setq idlwave-code-comment "klajsdfjpawe") ;   This is an attempt to disable
>   ;   this function.
> (setq idlwave-no-change-comment ";...")
> (setq idlwave-continuation-indent 0)
> (setq idlwave-block-indent 0)

    I hope this helps,

                                    Jeff Guerber

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by Jeff Guerber on Fri, 08 Dec 2000 21:43:51 GMT
View Forum Message <> Reply to Message

On Fri, 8 Dec 2000, Jeff Guerber wrote:

> Jack, I can take a quick stab at your questions.
> [...]

Jack caught me in the hall (we work in the same group here at Goddard), and told me that what he really wants to do is disable *all* automatic indentation done by idlwave, and make TAB work the same way it does in text-mode. (Did I get that right, Jack?) He'll probably weigh in with more details later. Sorry I misunderstood what he was trying to do.

Jeff Guerber

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by Jack Saba on Mon, 11 Dec 2000 14:29:25 GMT
View Forum Message <> Reply to Message

Thanks, JD.

The section with the <M>, <B>, ..., clarifies the four variables used to control positioning very nicely. And I can get almost everything I want with them.

However, it doesn't look like it is possible to use the continuation indentation scheme I prefer. I usually want the first nonblank character in the second line to line up with the beginning of the second word in the preceding line. Using your example, the code layout would be

```
> pro foo
>   ThisPro, a
>   for i=1,10 do begin
>     print, 'yay' + $
>           AnotherVar
>   endfor
> end
```

That's what would happen in text mode (for me, using NTEmacs in W98) if I hit 2 tabs at the start of the "AnotherVar" line. If I can set up continuation-extra-indent to do this, it would probably be sufficient, but I would still prefer to be able to tell idlwave mode to use the same tab control that text mode does. Is there a way to do either of these?

Jack Saba
<jack.saba@gsfc.nasa.gov

>

JD Smith wrote:
>
> Jack Saba wrote:
...
>>
>> Basically, I want complete control of text layout in the buffer.
>>
>> 1. I enter
>>
>> FOR i=0,n
>> stuff
>> ENDFOR
>>
>> When I hit the <CR>, IDLWave reformats this by indenting the ENDFOR 3 spaces:
>>
>> FOR i=0,n
>> stuff
>> ENDFOR
>>
>> How do I prevent this reformatting?
>>
>> 2. How do I make tabs in IDLwave mode work the way they do in text mode?
>> Currently, the first tab moves the cursor under the first character on the last
>> line, but subsequent tabs fall into a black hole.
>
> Think of <TAB> as more of a code cleaner-upper than a tab insertion
> mechanism. Almost all programming modes do it this way. Once a line of
> code is cleaned, cleaning it again will make no change. You *really*
> shouldn't be lining code up by hand, it's far too tedious.
>
> If you would just like code to line up as in your first example, try:
>
> idlwave-block-indent 3 ; Indentation settings
> idlwave-end-offset -3
>
> This will line the "ENDFOR" up with the "FOR".
>
> Here's a little pictogram to help with the indents:
>
> pro foo
> <M>ThisPro, a
> <M>for i=1,10 do begin
> <M><B>print, 'yay' + $
> <M><B><C>AnotherVar
> <M><B><E>endfor

> end
>
> <M> = Main block indent
> <B> = Block indent
> <C> = Continuation extra indent
> <E> = End offset
>
> in particular: <M>=3, <B>=3, <C>=1, <E>=-3 would produce:
>
> pro foo
> ThisPro, a
> for i=1,10 do begin
> print, 'yay' + $
> AnotherVar
> endfor
> end
>
> see how <E> cancels <B> in the endfor line? Pretty simple. Your end
> offset is leaving things hanging.
>
> <M>=0, <B>=5, <C>=3, <E>=-3 would make it:
>
> pro foo
> ThisPro, a
> for i=1,10 do begin
> print, 'yay' + $
> AnotherVar
> endfor
> end
>
> Oooh, ugly. But any degree of ugliness is tolerated.
>
> Currently, <RET> indents the line after inserting the newline. If you
> would prefer "<RET>" not to indent the line for you, you can simply set:
>
> (local-set-key "\r" 'newline)
>
> in your idlwave-mode-hook. But I can't see why anyone would really want
> this... you'll just end up hitting <TAB> after <RET> everytime. But
> give it a try if you like.
>
> If you don't want <TAB> to behave specially at all (by the way, you can
> always use C-<TAB> to get a "real" tab in any case), you can simply:
>
> (local-set-key "\t" 'idlwave-hard-tab)
>
> but this is getting really silly. The point of this indentation is to
> pick a style you like, and let IDLWAVE use it everywhere. It makes your

> code so much more maintainable. It would be nice if we could all agree
> on a code indentation scheme and commenting style, but *at least* a
> personally enforced standard is necessary.
>
> Also, if you ever find that you don't like what your customizations have
> done, disable them. Carsten has worked hard to make the default
> settings very useable.
>
> JD

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by John-David T. Smith on Mon, 11 Dec 2000 15:56:55 GMT
View Forum Message <> Reply to Message

Jack Saba wrote:
>
> Thanks, JD.
>
> The section with the <M>, <B>, ..., clarifies the four variables used to control
> positioning very nicely. And I can get almost everything I want with them.
>
> However, it doesn't look like it is possible to use the continuation indentation
> scheme I prefer. I usually want the first nonblank character in the second line
> to line up with the beginning of the second word in the preceding line. Using
> your example, the code layout would be
>
>> pro foo
>> ThisPro, a
>> for i=1,10 do begin
>> print, 'yay' + $
>> AnotherVar
>> endfor
>> end
>
> That's what would happen in text mode (for me, using NTEmacs in W98) if I hit 2
> tabs at the start of the "AnotherVar" line. If I can set up
> continuation-extra-indent to do this, it would probably be sufficient, but I
> would still prefer to be able to tell idlwave mode to use the same tab control
> that text mode does. Is there a way to do either of these?

I guess maybe I don't know what you mean by same tab control as Text Mode. There, <TAB> runs the command indent-relative (which you can find out with C-h k <TAB>). To get this exact command to run in IDLWAVE-mode, you'd need to undo the Tab functionality it sets up, using (in your idlwave-mode-hook):

```
(local-set-key "\t" 'indent-relative)
```

The problem is, if "smart" indenting is on ever, or if you run indent-region (M-C-\), this line will be reindented from your chosen position.

So it's not really a good solution, not to mention that it requires 4 or more key presses to achieve when starting on the first line (<SPACE> $ <RET> <TAB> <TAB> ...), when it should really require only one (M-<RET>) -- you have tried M-<RET>, yes?

However, there is a probable solution.

Note that IDLWAVE does use some intelligence when it comes to certain continuations. I suppose I should have noted that <C> only operates when some more compelling continuation indentation is not identified. Witness:

```
print, convol(a, $
          b, $
          c)
```

For your scheme, you'd get:

```
print, convol(a, $
      b, $
      c)
```

also, what about something like

```
a=myfunc(thisvar,thatvar,theothervar,THISKEYWORD=1, $
                                     THATKEYWORD=2)
```

Yuck. Here I'd bet you'd opt for lining it up by hand with 10 spaces. The flaw in your algorithm is demanding the second *word* of the first line line up. For this case, IDLWAVE already does what you want (I think):

```
a=myfunc(thisvar,thatvar,theothervar,THISKEYWORD=1, $
        THATKEYWORD=2)
```

It seems what you *really* want is for smart indenting to line up things with the first non-blank character after the comma in a procedure call, in the same way the "(" is treated in a function call, e.g.:

```
print, a, b, c, d, 1, 2, 3, $
      x,y,z
```

This looks like a reasonable option, in light of IDLWAVE's behavior with
continued functions. The "normal" continuation offset is still needed
in other cases, e.g.:

```
a=b+c+d+e+ $
  f+g+h
```

Perhaps even this could be a special case, always yielding:

```
a=b+c+d+e+ $
 f+g+h
```

Since functions and procedures are already being identified for other
reasons, I should think this is quite doable. This is exactly the kind
of thing that Carsten is eager to hear about, and also the sort of issue
that usually prompts a new alpha version addressing it the very same day
(unless it's really harebrained)!

It is difficult to relinquish the notion of lining up all your bits of
code yourself, but once you give in and let the program do what it was
designed for, your energies are freed for far more productive things.
As I said before, thinking about indentation is not quality use of time.

JD

--
J.D. Smith | WORK: (607) 255-6263
Cornell Dept. of Astronomy | (607) 255-5842
304 Space Sciences Bldg. | FAX: (607) 255-5875
Ithaca, NY 14853 |

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by Henkie on Tue, 12 Dec 2000 08:20:12 GMT
View Forum Message <> Reply to Message

Hi,

maybe you know how I can make my cursor keys work with the command history
as well ? Now they work like cursor keys, but I don't want them to ;-) .
How can I change this ?
Also, in the shell, my own libraries are not in the path, although I've set
idlwave-library-path. Any idea ?

Thanks,

Henk

"JD Smith" <jdsmith@astro.cornell.edu> wrote in message
news:3A311982.95FD3858@astro.cornell.edu...
>
>
> Alright, now this is some exciting news. Take a look online at this
> tutorial too, for more friendly reading:
>
> http://www.strw.LeidenUniv.nl/~dominik/Tools/idlwave/idlwave .html#SEC3
>
> P.S. For linux users: If you'd like to make use of that silly Windows
> key just outside your "Alt" keys (which in Windows is used to mail a
> copy of your last 100 browsing locations to bill@gates.com) you can
> easily do this. For me, adding the following to ~/.Xmodmap worked:
>
> keycode 115 = Hyper_L
> keycode 116 = Hyper_R
> clear Mod5
> add Mod5 = Hyper_L Hyper_R
>
> (and yes david, you can read all about this notation: man xmodmap)
>
> If these keycodes aren't correct, run "xev", position input in little
> white window, and press the Gates-was-here key. You'll see something
> like:
>
> blah blah, keycode 115 (keysym 0x0, NoSymbol), blah blah
>
> and similar for the other key. From this you can get the appropriate
> "keycode" above, but 115,116 will probably work. You can see how to use
> this method for any other keys too (you can make a "Super" modifier too
> if you have a spare).
>
> Now you can have lots of fun with:
>
>
H-down,H-up,H-t,H-l,H-w,H-e,H-x,H-@,H-?,H-p,H-q,H-z,H-y,H-r, H-h,H-m,H-o,H-u,
H-k,H-n,H-s,H-a,H-d,H-i
>
> Ahh the joy.
>
> JD

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by dominik on Tue, 12 Dec 2000 16:17:12 GMT
View Forum Message <> Reply to Message

>>>> > "JS" == JD Smith <jdsmith@astro.cornell.edu> writes:

JS> It seems what you *really* want is for smart indenting to line up things
JS> with the first non-blank character after the comma in a procedure call,
JS> in the same way the "(" is treated in a function call, e.g.:

```
JS> print, a, b, c, d, 1, 2, 3, $
JS>        x,y,z
```

JS> This looks like a reasonable option, in light of IDLWAVE's behavior with
JS> continued functions.

This is possible and very useful. It was easy to implement.

JS> The "normal" continuation offset is still needed
JS> in other cases, e.g.:

```
JS> a=b+c+d+e+ $
JS>    f+g+h
```

JS> Perhaps even this could be a special case, always yielding:

```
JS> a=b+c+d+e+ $
JS>   f+g+h
```

This on the other hand is really hard. In particular in your code,
JD, where the left side of the assignment can involve any combination
of pointers an objects :-). It would be trivial with a scalar
variable assignment, but not the general case.

JS> Since functions and procedures are already being identified for
JS> other reasons, I should think this is quite doable. This is
JS> exactly the kind of thing that Carsten is eager to hear about, and
JS> also the sort of issue that usually prompts a new alpha version
JS> addressing it the very same day (unless it's really harebrained)!

4.7c on my site.

- Carsten

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by Jeff Guerber on Tue, 12 Dec 2000 21:09:15 GMT
View Forum Message <> Reply to Message

Unlike my esteemed colleague Jack, I *do* like IDLWAVE's automatic
indentation... mostly. But while we're on the subject:

On Mon, 11 Dec 2000, JD Smith wrote:

> Note that IDLWAVE does use some intelligence when it comes to certain
> continuations. I suppose I should have noted that <C> only operates
> when some more compelling continuation indentation is not identified.
> Witness:
>
> print, convol(a, $
> b, $
> c)

Is there some way to disable _just_ this behavior, while still using the rest of IDLWAVE's automatic indentation features? When I'm doing something like assigning the result of a call to a function method of an object ref that's contained in a structure field, to a field of another structure (and I find myself doing things like this quite a bit), the open parenthese is pretty far over to the right, so this indentation style doesn't leave much room for the arguments. As clever as this is, frankly I'd rather just have the regular continuation indentation as specified by idlwave-continuation-indent. I've checked the manual and even the code, but couldn't find anything. (I'm using IDLWAVE 4.5, but I didn't see anything in the 4.7 manuals on the web site, either.) Thanks!

And many thanks, Carsten, for an extremely useful program!

Jeff Guerber

Subject: Re: IDLWAVE 4.7/Tutorial
Posted by Martin Schultz on Wed, 20 Dec 2000 15:59:22 GMT
View Forum Message <> Reply to Message

Jeff Guerber wrote:
>
> Unlike my esteemed colleague Jack, I *do* like IDLWAVE's automatic
> indentation... mostly. But while we're on the subject:
>
> On Mon, 11 Dec 2000, JD Smith wrote:
>
>> Note that IDLWAVE does use some intelligence when it comes to certain
>> continuations. I suppose I should have noted that <C> only operates
>> when some more compelling continuation indentation is not identified.
>> Witness:
>>
>> print, convol(a, $
>> b, $
>> c)
>
> Is there some way to disable _just_ this behavior, while still using

> the rest of IDLWAVE's automatic indentation features? When I'm doing
> something like assigning the result of a call to a function method of an
> object ref that's contained in a structure field, to a field of another
> structure (and I find myself doing things like this quite a bit), the open
> parenthese is pretty far over to the right, so this indentation style
> doesn't leave much room for the arguments. As clever as this is, frankly
> I'd rather just have the regular continuation indentation as specified by
> idlwave-continuation-indent. I've checked the manual and even the code,
> but couldn't find anything. (I'm using IDLWAVE 4.5, but I didn't see
> anything in the 4.7 manuals on the web site, either.) Thanks!
>
> And many thanks, Carsten, for an extremely useful program!
>
> Jeff Guerber

... how about a parameter to set the maximum indentation? I think the
automatic identation works just great for up to about 25 characters,
afterwards I second Jeff in that you end up with long lines unwanted.

I must admit I haven't downloaded 4.7 yet, and as I will not be
developing any IDL code this year, I will probably wait for 4.8 ;-)
But I really like the drive that you, Carsten, put behind this, and I
will enjoy to see my feature request in action.

Cheers,
Martin

--
[[[[[[[[[[
[[Dr. Martin Schultz Max-Planck-Institut fuer Meteorologie [[
[[Bundesstr. 55, 20146 Hamburg [[
[[phone: +49 40 41173-308 [[
[[fax: +49 40 41173-298 [[
[[martin.schultz@dkrz.de [[
[[[[[[[[[