Subject: IDLWAVE 5.3 -- idlwave.org
Posted by JD Smith on Mon, 28 Jun 2004 04:31:39 GMT
View Forum Message <> Reply to Message

IDLWAVE 5.3 -- http://idlwave.org

A small IDLWAVE release with improvements to continuation indentation,
especially within statement blocks, optional Kernighan-style
parenthetical indentation (see the new `idlwave-indent-parens-nested'
customize variable), executive command completion, and a convenient
"visit routines in this file" binding which makes large files with many
routines (like object definition files) easier to work with.

Available now at:

  http://idlwave.org

JD

 ==========
==========
IDLWAVE Tip of the Month:

IDLWAVE uses completion heavily; all prompts for filenames, keywords,
routine names, system variables, tags, executive commands, etc. have
associated with them a (sometimes large) list of possible completions.
For many things, such as routine info, IDLWAVE prompts you with a
default entry based on what's nearby in the text, which can be very
convenient.

A good example completion is the new "visit routines in this file"
command, bound to C-u C-C C-v. E.g., when editing
idlitsystem__define.pro:

  self->RegisterOperation, [C-u C-c C-v]

will prompt with:

Module (Default IDLitSystem::RegisterOperation<p>):

You can hit return to visit RegisterOperation directly, or Tab to see
the possible completion choices in a buffer (here, routines in this
file). This looks like:

Possible completions are:
IDLitSystem::AddOnNotifyObserver<p>
IDLitSystem::AddService<p>   IDLitSystem::AddSetting<p>
IDLitSystem::BeginClipboardInteraction<p>

IDLitSystem::Cleanup<p> IDLitSystem::ClearClipboard<p>
IDLitSystem::CreateTool<f> IDLitSystem::CreateVisualization<p>
IDLitSystem::DeleteTool<p> IDLitSystem::DoOnNotify<p>
IDLitSystem::DoSetProperty<f> IDLitSystem::DoUIService<f>
IDLitSystem::EndClipboardInteraction<p>
....<lots more trimmed>....

Middle-clicking selects one. You can also type in portions of a name
and let Tab do the rest:

Module (Default IDLitSystem::RegisterOperation<p>): IDLitSystem::Add[Tab]

Possible completions are:
IDLitSystem::AddOnNotifyObserver<p>
IDLitSystem::AddService<p> IDLitSystem::AddSetting<p>

and in this way can narrow down on the routine you want.

This is all very convenient, but wouldn't it be nice if there were a
quicker way to pare down all those selections without requiring the
mouse or too much typing? Luckily, there is. It's called mcomplete,
and is available here:

http://homepage1.nifty.com/bmonkey/emacs/elisp/mcomplete.el

This great utility enhances all completion prompts to make selection
very fast. Among its most useful features is substring searching.
Instead of requiring the full name, any sub-part of the name can be
used. In our example:

Module (Default IDLitSystem::RegisterOperation<p>): cli

magically turns into:

Module (Default IDLitSystem::RegisterOperation<p>):
cli[pboard]<IDLitSystem::BeginClipboardInteraction<p>,..>

which shows the first matching routine containing the substring "cli"
(which, it reports, can be unambiguously exanded to "clipboard"). C-s
and C-r cycle through the matches, Return selects the current one, and
Tab lists them:

Possible completions are:
IDLitSystem::BeginClipboardInteraction<p>
IDLitSystem::ClearClipboard<p> IDLitSystem::EndClipboardInteraction<p>
IDLitSystem::GetClipboardItemCount<f>
IDLitSystem::_UpdateClipboardStatus<p>

Usually just a few quick keystrokes and return gets you to the right item very quickly. It's very useful for jumping among routines with common prefixes (like a class name) which is cumbersome to type again and again: e.g. large foo__define.pro files. With mcomplete enabled, all completion selections (switching buffers, etc.) behave in this way.

==========