# Closing the Gaps
## Inter-Module Synergy

Ed J (mohawk)

London Perl Workshop 2014

# Introduction

- Modules: generalities
- Example: Inline, ExtUtils::Depends
- Slides available on web

# Modules

- Each encapsulates functionality
- Assembly
- Interaction

# Example: XS support

- Writing XS is hard
- Writing the build system is hard
- Writing XS that uses other XS modules is hard
- Why must it be so hard?
- WHY?

# And... breathe

# Gimp-Perl: Before

- Has own XS for GIMP C libraries
- Uses other XS: Gtk2, PDL
- Each had own way of providing build system info (paths etc)
- GP build system was “challenging”

# Gimp-Perl: Before

```
my %gimpcfg = ExtUtils::PkgConfig->find("gimp-2.0");
require PDL::Core::Dev;
my $pkg = new ExtUtils::Depends qw(Gimp Gtk2);
$pkg->set_inc(join ' ',
   $gimpcfg{"cflags"}, &PDL::Core::Dev::PDL_INCLUDE);
$pkg->set_libs(join('', `$gimptool –libs`));
WriteMakefile($pkg->get_makefile_vars);
```

# Gimp-Perl: Vision

```
my $pkg = ExtUtils::Depends->new(
  qw(Gimp Alien::Gimp Gtk2 PDL));
WriteMakefile($pkg->get_makefile_vars);
```

# Vision: Obstacles

- Me understanding EU::D workings
- Making PDL support EU::D
- Making EU::D support easy
- Getting PDL buy-in
- Getting EU::D buy-in

# ExtUtils::Depends

- Previously support was “black box”
  $pkg->save_config('Gimp/IFiles.pm');
- Now API can conform to
  package PDL::Install::Files;
  require PDL::Core::Dev;
  $inc = &PDL::Core::Dev::PDL_INCLUDE;

# Opportunity

- PDL uses Inline
- Inline use requires PDL boilerplate
- Inline doc talks about “with”:
  use Inline with => 'Event';
- Inline has interface to encapsulate XS build info:
  sub Event::Inline { … }
- Why not make EU::D work with that


ATTEMPTED MURDER

# Execution

- Make EU::D::save_config output:
  sub Other::Install::Files::Inline { … }
- Make EU::D::load_config also call the Inline method

# Result

- Other modules can just support In-line and get free EU::D support:

```
package PDL::Install::Files;
sub Inline { … }
```

- Gimp's Makefile.PL can be:

```
my $pkg = ExtUtils::Depends->new(qw(Gimp Gtk2 PDL));
WriteMakefile($pkg->get_makefile_vars);
```

- Can use for Inline code like this:

```
use Inline with => 'PDL', C => “...”;
```

# And Alien::Gimp?

- Subclassing from Alien::Base enables easy Alien modules to encapsulate XS build info
- Why not make that work with EU::D and Inline
- Now it does!

# Synergy

- Making these modules work together took work
- Save lots of other module-writers lots of duplicated effort
- Benefit:
  module-writers
  module-users
  program users

# Conclusions

- Working with other module-writers can be like herding cats
- Source of ideas
- Helps find bugs
- Makes your module more valuable

# Further reading

- Inline
- ExtUtils::Depends
- Alien::Base

- https://github.com/mohawk2/lpw2014-slides

# Questions?