cpanfile

Tatsuhiko Miyagawa
London Perl Workshop 2011

Hey, cpanminus
is *so* Hot!

Hey, cpanminus
is *so* Hot!

Making a Perl module

that depends on CPAN modules

= Making a CPAN distribution

Declaring CPAN Dependencies

Makefile.PL
Build.PL

```
use ExtUtils::MakeMaker;
WriteMakefile(
  NAME => ‘MyModule’,
  VERSION => ‘1.22’,
  PREREQ_PM => {
    ‘JSON’ => 0,
    ‘Hash::MultiValue’ => ‘0.11’,
  },
);
```

```
use Module::Build;
my $build = Module::Build->new(
  module_name => ‘MyModule’,
  requires => {
    ‘JSON’ => ‘2.00’,
    ‘Hash::MultiValie’ => ‘0.20’,
  },
);
$build->create_build_script;
```

```
use inc::Module::Install;

name ‘MyApp’;
all_from ‘lib/MyApp.pm’;

requires ‘JSON’, ‘2.00’;
requires ‘Hash::MultiValue’ => 0.10;

WriteAll;
```

Or use Dist::Zilla.

Making a Perl *application*

that depends on CPAN modules

≠ Making a CPAN distribution

Declaring CPAN Dependencies

?

```
Makefile.PL
Build.PL
```

Problems with

Makefile.PL/Build.PL

Lots of boilerplate

```
use ExtUtils::MakeMaker;
WriteMakefile(
  NAME => ‘MyModule’,
  VERSION => ‘1.22’,
  PREREQ_PM => {
    ‘JSON’ => 0,
    ‘Hash::MultiValue’ => ‘0.11’,
  },
);
```

```
use Module::Build;
my $build = Module::Build->new(
  module_name => ‘MyModule’,
  requires => {
    ‘JSON’ => ‘2.00’,
    ‘Hash::MultiValie’ => ‘0.20’,
  },
);
$build->create_build_script;
```

```
use inc::Module::Install;

name ‘MyApp’;
all_from ‘lib/MyApp.pm’;

requires ‘JSON’, ‘2.00’;
requires ‘Hash::MultiValue’ => 0.10;

WriteAll;
```

dzil has no boilerplate

... but it’s not a *distribution*!

Can have deps

configure_requires, META.yml
inc/Module/Install.pm

Makefile.PL
Build.PL

configure
build
test
install

✔ configure

build

test

install

My application is not meant to be *installed*!

e.g. dotcloud push

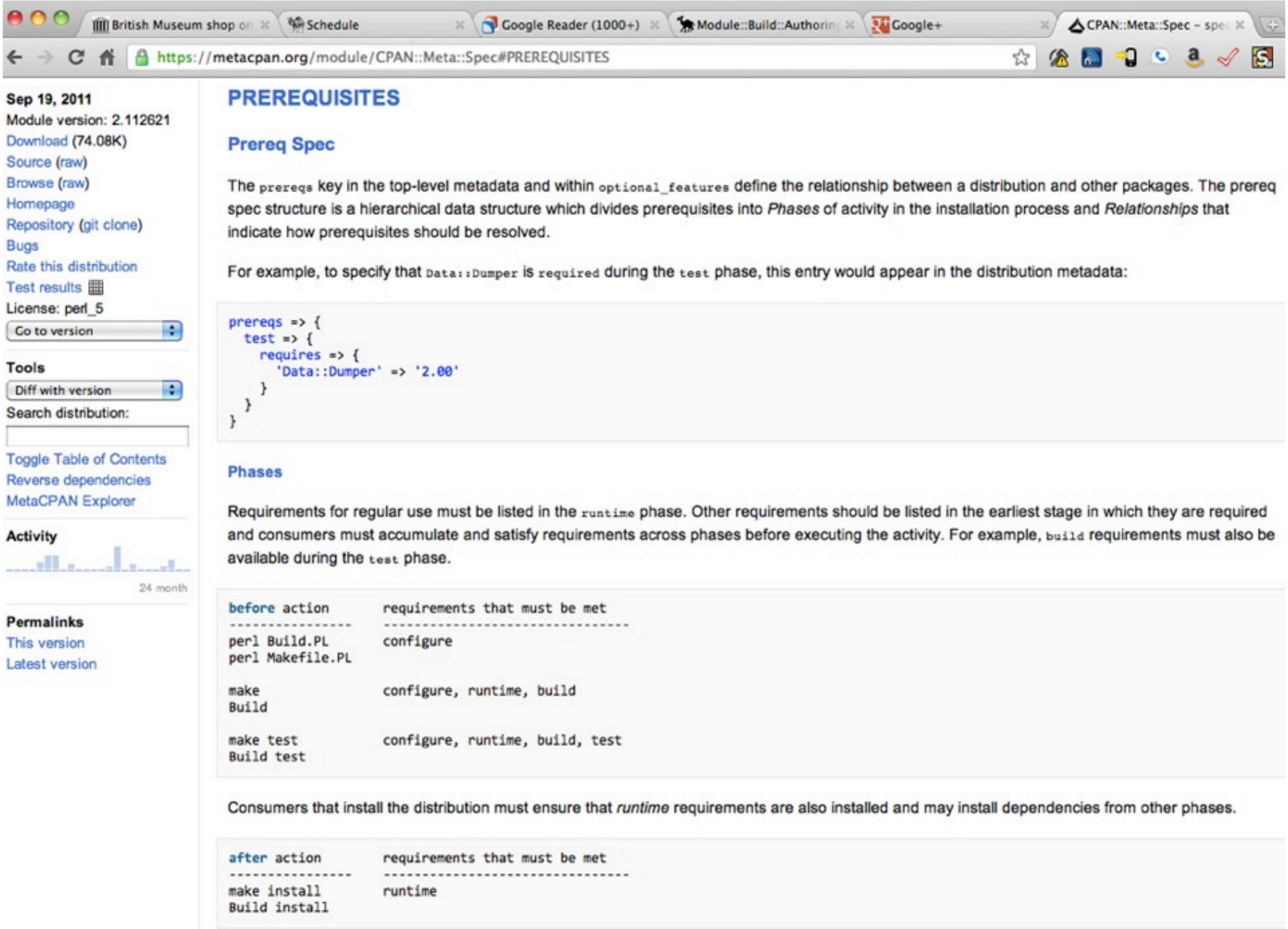
Great for *dists*.
Not-so for *apps*.

By the way...

I <3
Module::Install's DSL

CPAN Meta Spec v2

Awesome stuff.



PREREQUISITES

Prereq Spec

The `prereqs` key in the top-level metadata and within `optional_features` define the relationship between a distribution and other packages. The prereq spec structure is a hierarchical data structure which divides prerequisites into *Phases* of activity in the installation process and *Relationships* that indicate how prerequisites should be resolved.

For example, to specify that `Data::Dumper` is `required` during the `test` phase, this entry would appear in the distribution metadata:

```
prereqs => {
  test => {
    requires => {
      'Data::Dumper' => '2.00'
    }
  }
}
```

Phases

Requirements for regular use must be listed in the `runtime` phase. Other requirements should be listed in the earliest stage in which they are required and consumers must accumulate and satisfy requirements across phases before executing the activity. For example, `build` requirements must also be available during the `test` phase.

```
before action       requirements that must be met
----------------    --------------------------------
perl Build.PL       configure
perl Makefile.PL

make                configure, runtime, build
Build

make test           configure, runtime, build, test
Build test
```

Consumers that install the distribution must ensure that *runtime* requirements are also installed and may install dependencies from other phases.

```
after action        requirements that must be met
----------------    --------------------------------
make install        runtime
Build install
```

Marrying them together?

A Proposal

cpanfile

```
requires ‘Plack’, 0.9980;
requires ‘Catalyst’, ‘>= 5.8, < 5.9’;

conflicts ‘MooseX::Foo’, ‘< 0.10’;

on ‘test’ => sub {
    requires ‘Test::More’, 0.80;
};
on ‘develop’ => sub {
    requires ‘Devel::NYTProf’;
};
```

inspired by:

gemfile(5)

Backward compatible to:

Module::Install(::DSL)

Convertible to:

CPAN::Meta::Prereqs

```
requires ‘Plack’, 0.9980;
requires ‘Catalyst’, ‘>= 5.8, < 5.9’;

conflicts ‘MooseX::Foo’, ‘< 0.10’;

on ‘test’ => sub {
    requires ‘Test::More’, 0.80;
};
on ‘develop’ => sub {
    requires ‘Devel::NYTProf’;
};
```

```
CPAN::Meta::Prereqs->new({
  runtime => {
    requires => {
      ‘Plack’ => 0.9980,
      ‘Catalyst’ => ‘>= 5.8, < 5.9’,
    },
    conflicts => {
      ‘MooseX::Foo’ => ‘< 0.10’,
    },
  },
  test => {
    requires => {
      ‘Test::More’ => 0.80,
    },
  },
  develop => {
    requires => {
      ‘Devel::NYTProf’ => 0,
    },
  }
});
```

Will be supported by:

cpanm 1.6, carton 1.0

Carton

github.com/miyagawa/carton

Apps first,
Dists later.

Turing Complete?

PPI, B::*, Safe

Validated by authoring tools

github.com/miyagawa/cpanfile

Thanks!

http://speakerdeck.com/u/miyagawa