NAME

MooseX::Role::Parameterized::Extending − extending MooseX::Role::Parameterized roles

DESCRIPTION

There are heaps of useful modules in the `MooseX` namespace that you can use to make your roles more powerful. However, they do not always work out of the box with MooseX::Role::Parameterized, but it's fairly straight-forward to achieve the functionality you desire.

MooseX::Role::Parameterized was designed to be as extensible as the rest of Moose, and as such it is possible to apply custom traits to both the parameterizable role or the ordinary roles they generate. In this example, we will look at applying the fake trait `MooseX::MagicRole` to a parameterizable role.

First we need to define a new metaclass for our parameterizable role.

```
package MyApp::Meta::Role::Parameterizable;
use Moose;
extends 'MooseX::Role::Parameterized::Meta::Role::Parameterizable';
with 'MooseX::MagicRole';
```

This is a class (observe that it uses Moose, not Moose::Role) which extends the class which governs parameterizable roles. MooseX::Role::Parameterized::Meta::Role::Parameterizable is the metaclass that packages using MooseX::Role::Parameterized receive by default.

Note that the class we are extending, MooseX::Role::Parameterized::Meta::Role::Parameteriz***able***, is entirely distinct from the similarly-named class which governs the ordinary roles that parameterized roles generate. An instance of MooseX::Role::Parameterized::Meta::Role::Parameteriz***ed*** represents a role with its parameters already bound.

Now we can take advantage of our new subclass by specifying that we want to use `MyApp::Meta::Role::Parameterizable` as our metaclass when importing MooseX::Role::Parameterized:

```
package MyApp::Role;
use MooseX::Role::Parameterized −metaclass => 'MyApp::Meta::Role::Parameteriz

role {
    ...
}
```

And there you go! `MyApp::Role` now has the `MooseX::MagicRole` trait applied.