

Moose Quick-Ref Card

A modern object system for Perl 5

Exported Functions

```
use Moose;
    Turns on strict and warnings.
    Exports confess and blessed.

extends @superclasses
    Moose's alternative to use base.
    Note that it will re-set @ISA.

with @roles
with $role => { %options }
    Consume roles (interfaces) as an
    alternative to extending classes.

has $name => %options
    Install an attribute into this class.
    See below for %options details.

has "+$name" => %options
    Clone and extend an attribute.

before @names => sub {...}
around @names => sub {...}
after @names => sub {...}
    Extend a superclass's method. around is
    passed ($next_method, $self, @args).

override $name => sub { super() }
    Explicit override of a method.

augment $name => sub { inner() }
    The inverse of override/super.
```

Attribute Constructor Options

```
is => 'rw'|'ro'
    Creates a read/write or read-only
    accessor. If you omit this option, no
    accessor will be created.

isa => $type_name | '$ta|$tb|...'
    Set up run-time type checking.
    See below for $type_name details.

does => $role
    Value's class must consume $role.

metaclass => $name
    Extend attribute via a metaclass.

traits => [@role_names]
    Apply roles to attribute's meta-object.

coerce => 1|0
    Allow coercion to $type_name on storage.
    See below for details.

required => 1|0
    Attribute must always have a value.

weak_ref => 1|0
    Value is stored as weakened ref
    (note: conflicts with coercion).

lazy => 1|0
    Don't create a value from the (required)
    default until accessed.

auto_deref => 1|0
    Accessor will dereference array or hash
    references (isa must be set).
```

```
trigger => sub {...}
    Code to run after attribute is set. Is
    passed ($self, $new_val).

default
=> $val | sub{ []|{}|sub{...} }
    Default value to initialize attribute.
    The outer sub{} is passed $self.

predicate => $name
    Method $name will perform a basic
    defined test on the attribute.

reader|writer|clearer => $sub_name
    Provide the subroutine names used to
    read from, write to, and uninitialized the
    stored value.

builder => $sub_name
    Separate method to return default value.
    Better than default for subclassing.

lazy_build => 1
    Sets lazy, required, predicate (has_$name),
    clearer (clear_$name) and
    builder (_build_$name).

init_arg => $name
    Name for attribute when passed into the
    constructor, or disallowed if undef.

handles =>
@ary|$sh|qr//|$role|sub{...}
    Sets up methods which delegate to
    methods of the value's class.
    Requires that isa be set.
```

Data Type Constraints

The built-in type-constraints are:

```
Any
Maybe[TypeName]
Item
Bool
Undef (use with care)
Defined
Value
Num
Int
Str
RoleName
ClassName (means "is loaded" and isa)
Ref
ScalarRef
ArrayRef or ArrayRef[TypeName]
HashRef or HashRef[TypeName]
CodeRef
RegexpRef
GlobRef
FileHandle
Object
Role
```

To define your own, *global* types:
use Moose::Util::TypeConstraints;

```
type $name
=> where { <code> }
=> message { $message };
    A new type-constraint with no parent.
```

```
subtype $name
=> as $parent
=> where { <code> }
=> message { $message };
    Subtype of an existing type.
```

It is recommended that you always quote \$name. Moose checks \$parent constraints first. The block of <code> must evaluate to true. A \$message is optional, and used in confess if the constraint check fails.

Data Type Constraints, continued...

```
enum $name => @values;
    Constraining to a list of Str values.

subtype 'TypeName'
=> as class_type 'SomeClass';
    Idiomatic check of value's class.

has $name => (isa => 'SomeClass');
    Magical version of above.
```

Data Type Coercions

```
use Moose::Util::TypeConstraints;
coerce $type
=> from $some_type
=> via { <code> }
=> from $some_other_type
=> via { <other_code> };
    Instruct Moose in how to coerce data
    from $some_type to $type. You can chain
    alternative coercions as shown.
```

Coercion <code> is passed a value in \$_ and returns the value to be stored.

Choice Related Modules

- Class::MOP
- Moose::Exporter
- MooseX::AttributeHelpers
- MooseX::ClassAttribute
- MooseX::Getopt
- MooseX::Object::Pluggable
- MooseX::Role::Parameterized
- MooseX::Storage
- MooseX::Types

Other Tidbits

```
use Moose::Role;
    A role (or interface or trait) can only be
    consumed, not instantiated directly.

requires @methods;
    Methods which must be implemented by
    the consuming class.

my $meta = __PACKAGE__->meta;
    Get the cached metaclass for a package.
```

```
$meta->make_immutable;
no Moose;
no Moose::Role;
    Finalize the class to make it faster,
    and unimport the Moose 'keywords'.
```

The BUILD method of each class will be executed after the type constraint checks by the constructor, and is passed (\$self, \$params).

Before that, BUILDARGS is passed (\$class, @params) to convert into the \$params hashref.

The DEMOLISH method of each class is called at object destruction.

Meta Class and Trait namespaces:
Moose::Meta::Attribute::Custom::Metaclass
Moose::Meta::Type::Custom::Trait::Strait

This quick-ref card is © Oliver Gorwits
2009-05-31 version 4.1
<http://tinyurl.com/moosequickref>
Thanks to many people from #moose