

NAME

Type::Tiny::Class – type constraints based on the "isa" method

STATUS

This module is covered by the Type-Tiny stability policy.

DESCRIPTION

Type constraints of the general form `{ $_->isa("Some::Class") }`.

This package inherits from Type::Tiny; see that for most documentation. Major differences are listed below:

Constructor

`new`

When the constructor is called on an *instance* of Type::Tiny::Class, it passes the call through to the constructor of the class for the constraint. So for example:

```
my $type = Type::Tiny::Class->new(class => "Foo::Bar");
my $obj = $type->new(hello => "World");
say ref($obj); # prints "Foo::Bar"
```

This little bit of DWIM was borrowed from MooseX::Types::TypeDecorator, but Type::Tiny doesn't take the idea quite as far.

Attributes

`class`

The class for the constraint.

`constraint`

Unlike Type::Tiny, you *cannot* pass a constraint coderef to the constructor. Instead rely on the default.

`inlined`

Unlike Type::Tiny, you *cannot* pass an inlining coderef to the constructor. Instead rely on the default.

`parent`

Parent is automatically calculated, and cannot be passed to the constructor.

Methods

`plus_constructors($source, $method_name)`

Much like `plus_coercions` but adds coercions that go via a constructor. (In fact, this is implemented as a wrapper for `plus_coercions`.)

Example:

```
package MyApp::Minion;

use Moose; extends "MyApp::Person";

use Types::Standard qw( HashRef Str );
use Type::Utils qw( class_type );

my $Person = class_type({ class => "MyApp::Person" });

has boss => (
    is      => "ro",
    isa     => $Person->plus_constructors(
        HashRef,    "new",
        Str,         "_new_from_name",
    ),
    coerce => 1,
);
```

```

package main;

 MyApp::Minion->new(
    ...,
    boss => "Bob", ## via MyApp::Person->_new_from_name
 );

 MyApp::Minion->new(
    ...,
    boss => { name => "Bob" }, ## via MyApp::Person->new
 );

```

Because coercing HashRef via constructor is a common desire, if you call `plus_constructors` with no arguments at all, this is the default.

```

$classstye->plus_constructors(HashRef, "new")
$classstye->plus_constructors() ## identical to above

```

This is handy for Moose/Mouse/Moo-based classes.

```
stringifies_to($constraint)
See Type::Tiny::ConstrainedObject.
```

```
numifies_to($constraint)
See Type::Tiny::ConstrainedObject.
```

```
with_attribute_values($attr1 => $constraint1, ...)
See Type::Tiny::ConstrainedObject.
```

BUGS

Please report any bugs to <http://rt.cpan.org/Dist/Display.html?Queue=Type-Tiny>.

SEE ALSO

Type::Tiny::Manual.

Type::Tiny.

Moose::Meta::TypeConstraint::Class.

AUTHOR

Toby Inkster <tobyink@cpan.org>.

COPYRIGHT AND LICENCE

This software is copyright (c) 2013–2014, 2017–2019 by Toby Inkster.

This is free software; you can redistribute it and/or modify it under the same terms as the Perl 5 programming language system itself.

DISCLAIMER OF WARRANTIES

THIS PACKAGE IS PROVIDED “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.