NAME

      MooseX::MultiMethods – Multi Method Dispatch based on Moose type constraints

VERSION

      version 0.10

SYNOPSIS

```
package Paper;    use Moose;
package Scissors; use Moose;
package Rock;     use Moose;
package Lizard;   use Moose;
package Spock;    use Moose;

package Game;
use Moose;
use MooseX::MultiMethods;

multi method play (Paper    $x, Rock     $y) { 1 }
multi method play (Paper    $x, Spock    $y) { 1 }
multi method play (Scissors $x, Paper    $y) { 1 }
multi method play (Scissors $x, Lizard   $y) { 1 }
multi method play (Rock     $x, Scissors $y) { 1 }
multi method play (Rock     $x, Lizard   $y) { 1 }
multi method play (Lizard   $x, Paper    $y) { 1 }
multi method play (Lizard   $x, Spock    $y) { 1 }
multi method play (Spock    $x, Rock     $y) { 1 }
multi method play (Spock    $x, Scissors $y) { 1 }
multi method play (Any      $x, Any      $y) { 0 }

my $game = Game->new;
$game->play(Paper->new, Rock->new);       # 1, Paper covers Rock
$game->play(Spock->new, Paper->new);      # 0, Paper disproves Spock
$game->play(Spock->new, Scissors->new);   # 1, Spock smashes Scissors
```

DESCRIPTION

      This module provides multi method dispatch based on Moose type constraints. It does so by providing a `multi` keyword that extends the `method` keyword provided by MooseX::Method::Signatures.

      When invoking a method declared as `multi` a matching variant is being searched in all the declared multi variants based on the passed parameters and the declared type constraints. If a variant has been found, it will be invoked. If no variant could be found, an exception will be thrown.

AUTHOR

      Florian Ragwitz <rafl AT debian DOT org>

COPYRIGHT AND LICENSE