

# Adventures in Perl6

Nelson Ferraz < [nferraz@gmail.com](mailto:nferraz@gmail.com) >

Nordic Perl Workshop  
Copenhagen, 2007

# What is an Adventure?

- Computer-based
- Interactive
- Text-based (mostly)
- Story-telling

# Why adventures?

- It's a fun way to learn a new language:
  - Data structures
  - Text processing
  - Flow control

# Example

```
>inventory
```

```
You are carrying:
```

```
    a chocolate biscuit
```

```
    an electric torch (providing light and closed)
```

```
    a crumpled piece of paper
```

```
>read paper
```

```
    Things to do:
```

```
    1. Find map
```

```
    2. Phone airport to check parking
```

```
    3. Health forms...
```

```
>enter trapdoor
```

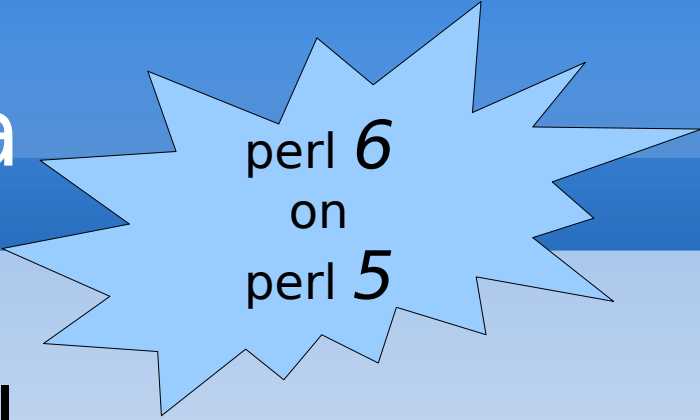
```
Yes, probably just as well to give up looking... Oh well.
```

```
*** You have missed the point entirely ***
```

# Perl6 now

- Pugs
- v6-alpha
- mini-perl6
- kinda-perl6

# v6-alpha



perl 6  
on  
perl 5

- 1. Install

```
# cpan v6-alpha
```

- 2. Use

```
#!/usr/bin/perl  
  
use v6-alpha;
```

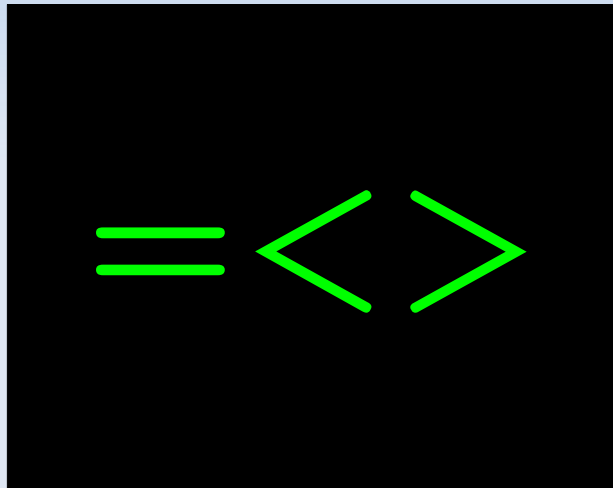
# The main loop

- In Perl5...

```
while (1) {  
    my $input = <>;  
}
```

- But... there's no more diamond operator in Perl6!!! :-)

# The fish operator



# The new main loop

```
while (1) {  
    my $input = =<>;  
}
```

# Parsing commands

- In Perl 5...

```
# clean up articles, prepositions, etc.  
$input =~ s/\b(a|an|the|in|on|at)\b/ /g;  
$input =~ s/\s+/ /;  
  
# extract verb and object from input  
my ( $verb, $object ) = split( $input );
```

- In Perl6 we have grammars!!! :-)

# A simple Grammar

```
grammar Adventure {
  token command {
    | <verb> <ws> <article> <ws> <object>
    | <verb> <ws> <object>
    | <verb>
  }

  token verb { look|take|drop|use }
  token object { table|book|diamond|knife }
  token article { a|an|the|at|in|on|to }
}
```

# Using the grammar

```
my $command = Adventure.command( $input );  
if ( $command{'verb'} ) {  
    # do something with $command{'object'};  
}
```

# We can extend the grammar

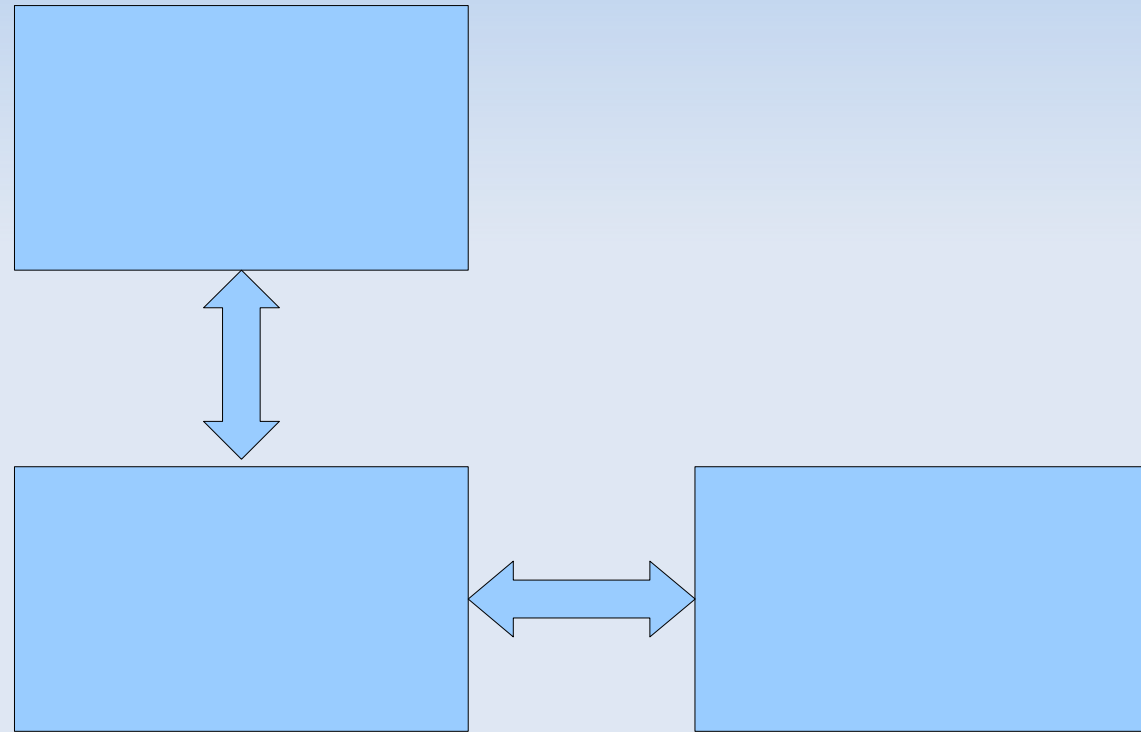
```
grammar Adventure {
  token command {
    | <verb> <ws> <article> <ws> <object>
    | <verb> <ws> <object>
    | <verb>
    | <direction>
  }

  token verb      { look|take|drop|use }
  token object    { table|book|diamond|knife }
  token article   { a|an|the|at|in|on|to }
  token direction { north|south|east|west }
}
```

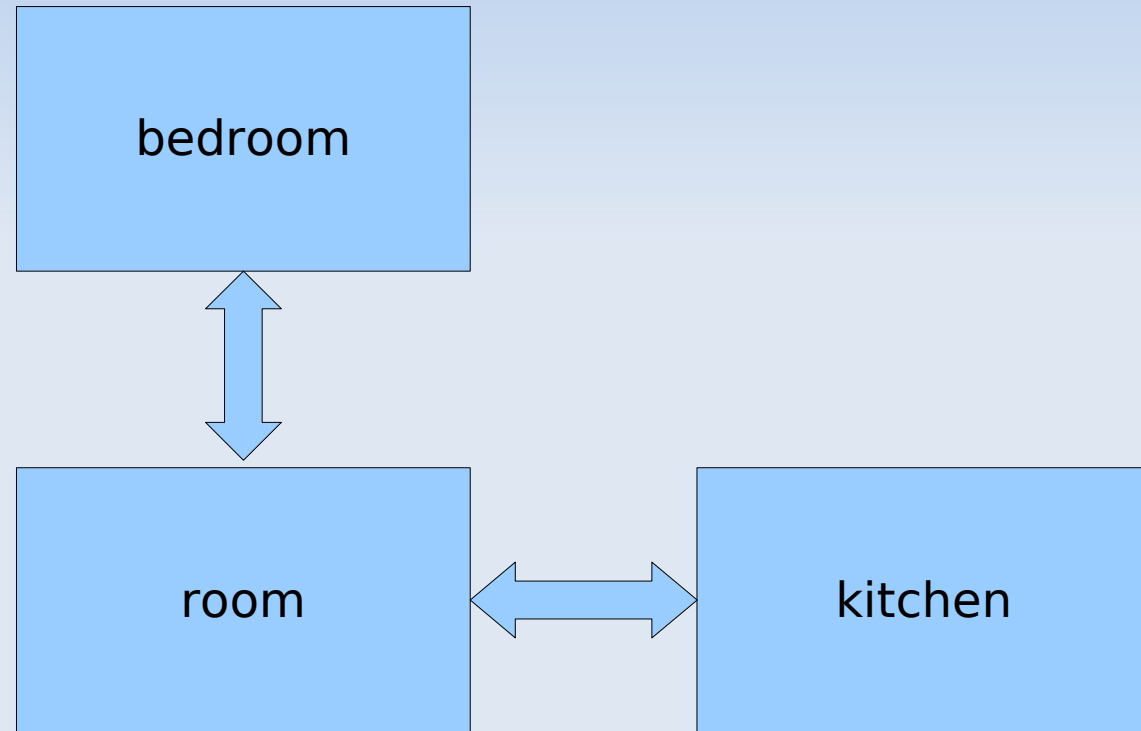
# Populating our adventure

- Places
- Objects
- Challenges

# The Map



# The Map



# The map

```
my %places = (  
    bedroom => {  
        south => "room",  
    },  
    room => {  
        north => "bedroom",  
        east  => "kitchen",  
    },  
    kitchen => {  
        west => "room",  
    },  
);
```

# Variables

- Variable names look pretty much like Perl5, but:
  - `%object{foo}` does the right thing
  - and `@array[42]` does the right thing too

# The objects

```
my %object = (  
    bed => {  
        place => "bedroom",  
    },  
    table => {  
        place => "room",  
    },  
    book => {  
        place => "table",  
    },  
    knife => {  
        place => "kitchen",  
    },  
);
```

# The player

```
my %player = (  
    place    => "bedroom",  
);
```

# Action: walk\_to <direction>

```
sub walk_to( $direction ) {  
    my $new_place =  
        %map{ %player{'place'} }{$direction};  
  
    if $new_place { # can go there  
        say "You entered the $new_place";  
        %player{'place'} = $new_place;  
    } else {  
        say "You can't go $direction";  
    }  
}
```

# Action: take <object>

```
sub take( $object ) {  
    if is_here( $object ) {  
        say "You took the $object";  
        %object{$object}{'place'} = 'player';  
    } else {  
        say "I don't see that here";  
    }  
}
```

# Action: drop <object>

```
sub drop( $object ) {
    if i_have( $object ) {
        say "You dropped the $object";
        %object{$object}{'place'} =
            %player{'place'};
    } else {
        say "You don't have that";
    }
}
```

# Source code

# Questions?

**Nelson Ferraz < [nferraz@gmail.com](mailto:nferraz@gmail.com) >**