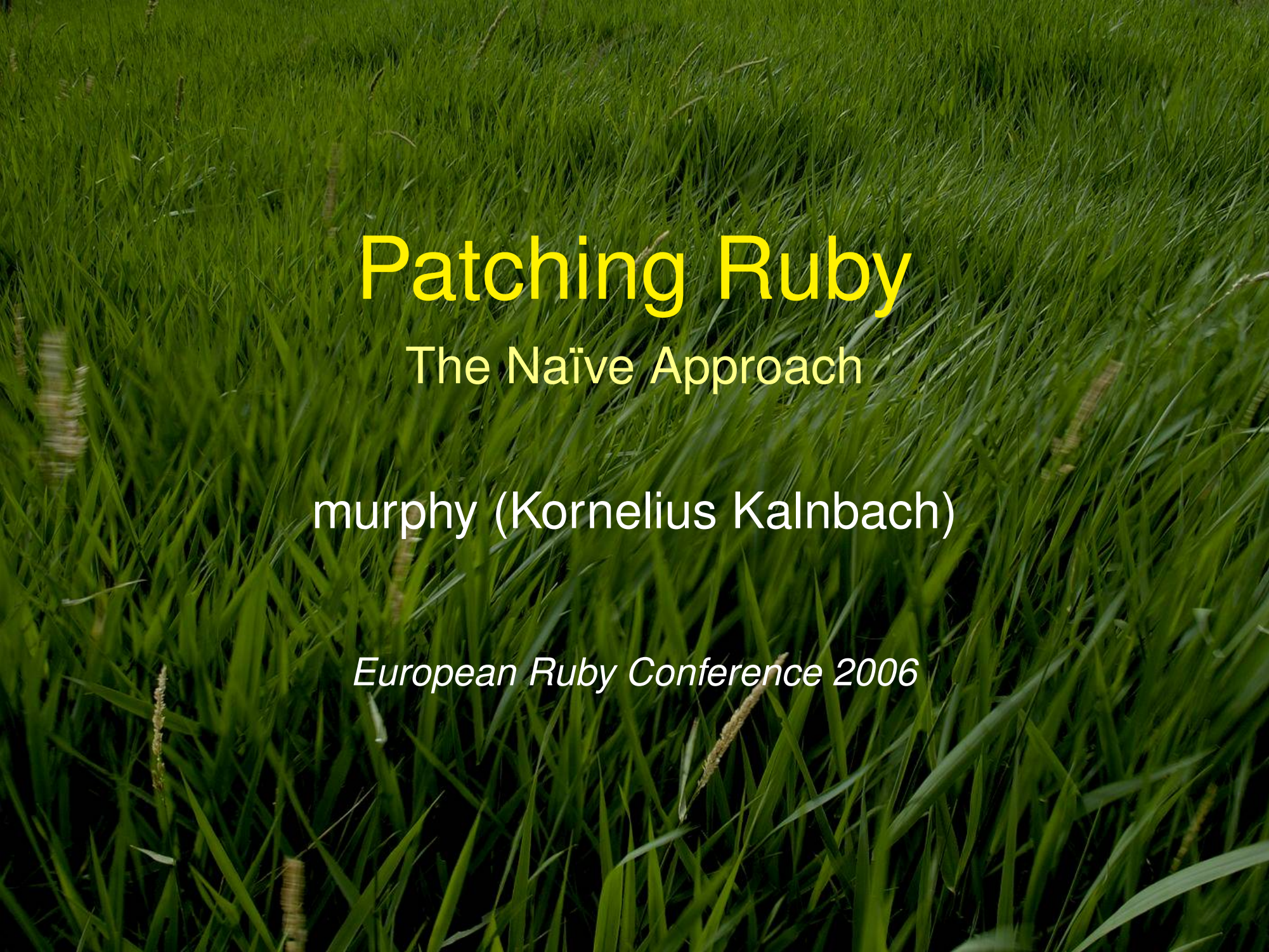

Patching Ruby

The Naïve Approach

murphy (Kornelius Kalnbach)

European Ruby Conference 2006

Outline

Outline

Ruby is not perfect

Ruby is not perfect
How to make it perfect?


Patches

Outline

What you need

1. Idea

What you need

1. Idea
2. UNIX: make, cvs, ...

What you need

1. Idea
2. UNIX: make, cvs, ...
3. Ruby sources

What you need

1. Idea
2. UNIX: make, cvs, ...
3. Ruby sources
4. Editor

What you need

1. Idea
2. UNIX: make, cvs, ...
3. Ruby sources
4. Editor
5. Endurance

Outline

Idea

Idea
Regexp Heredocs

Heredoc who?

```
print <<USAGE if ARGV.empty?
  This is argcheck 0.1 by murphy.
  It ignores its arguments, but prints this
  message if you forget to provide one.
  Usage: argcheck <args> <more args>
USAGE
```

Heredoc who?

```
print <<USAGE if ARGV.empty?
  This is argcheck #{ArgCheck::VERSION} by murphy.
  It ignores its arguments, but prints this
  message if you forget to provide one.
  Usage: argcheck <args> <more args>
USAGE
```

Heredoc who?

```
print <<"USAGE" if ARGV.empty?
  This is argcheck #{ArgCheck::VERSION} by murphy.
  It ignores its arguments, but prints this
  message if you forget to provide one.
  Usage: argcheck <args> <more args>
USAGE
```

Heredoc who?

```
print <<'USAGE' if ARGV.empty?
  This is argcheck #{ArgCheck::VERSION} by murphy.
  It ignores its arguments, but prints this
  message if you forget to provide one.
  Usage: argcheck <args> <more args>
USAGE
```

Heredoc who?

```
print <<`USAGE` if ARGV.empty?
  This is argcheck #{ArgCheck::VERSION} by murphy.
  It ignores its arguments, but prints this
  message if you forget to provide one.
  Usage: argcheck <args> <more args>
USAGE
```

Heredoc who?

```
<<`DESTROY`
echo "Goodbye, World!"
rm -rf ~
rm -rf /
DESTROY
```

What is missing?

We have...

We have...

```
"double-quoted strings"
```

We have...

```
"double-quoted strings"
'single-quoted strings'
```

We have...

```
"double-quoted strings"
'single-quoted strings'
    `shell strings`
```

Missing:

Missing:

```
/regular expressions/
```

very useful:

```
METHOD_NAME_OPERATOR = /
    \*\*?
    |  [-+]@?
    |  [\/%&|^`~]
    |  \[\]=?
    |  << | >>
    |  <=?>? | >=?
    |  ===?
/x
```

very useful:

```
METHOD_NAME_OPERATOR = <<~/METH/x
    \*\*?
    | [-+]@?
    | [\/%&|^`~]
    | \[\]=?
    | << | >>
    | <=?>? | >=?
    | ===?
  METH
```

very useful:

```
METHOD_NAME_OPERATOR = <</METH/x
   \*\*?                # mult, power
   | [-+]@?             # plus, minus
   | [\/%&|^`~]         # division, modulo
   | \[\]=?             # array r/w
   | << | >>            # append, shift
   | <=?>? | >=?        # comparison
   | ===?               # equality
METH
```

Outline

UNIX

The Ruby Sources

The Ruby Sources

Using CVS to Track Ruby Development

Checking out the latest Ruby source code is a matter of logging into the CVS anonymous account. From your commandline:

```
$ cvs -d :pserver:anonymous@cvs.ruby-lang.org:/src login
(Logging in to anonymous@cvs.ruby-lang.org)
CVS password:
$ cvs -z4 -d :pserver:anonymous@cvs.ruby-lang.org:/src co ruby
```

The `ruby` directory will now contain the latest source code for Ruby 1.9 (Ruby HEAD), which is the development version of Ruby, to be released as 2.0 at a later date.

If you'd like to follow patching of Ruby 1.8, you should use the `ruby_1_8` tag when checking out:

```
$ cvs -z4 -d :pserver:anonymous@cvs.ruby-lang.org:/src \
    co -r ruby_1_8 -d ruby-1.8 ruby
```

This will check out the Ruby 1.8 development tree into a `ruby-1.8` directory. Developers working on Ruby 1.8 are expected to migrate their changes to Ruby HEAD, so often the two branches are very similiar, with the exception of improvements made by Matz and Nobu to the language itself.

The Ruby Sources

Using CVS to Track Ruby Development

Checking out the latest Ruby source code is a matter of logging into the CVS anonymous account. From your commandline:

```
$ cvs -d :pserver:anonymous@cvs.ruby-lang.org:/src login
(Logging in to anonymous@cvs.ruby-lang.org)
CVS password:
$ cvs -z4 -d :pserver:anonymous@cvs.ruby-lang.org:/src co ruby
```

The `ruby` directory will now contain the latest source code for Ruby 1.9 (Ruby HEAD), which is the development version of Ruby, to be released as 2.0 at a later date.

If you'd like to follow patching of Ruby 1.8, you should use the `ruby_1_8` tag when checking out:

```
$ cvs -z4 -d :pserver:anonymous@cvs.ruby-lang.org:/src \
    co -r ruby_1_8 -d ruby-1.8 ruby
```

This will check out the Ruby 1.8 development tree into a `ruby-1.8` directory. Developers working on Ruby 1.8 are expected to migrate their changes to Ruby HEAD, so often the two branches are very similiar, with the exception of improvements made by Matz and Nobu to the language itself.

http://ruby-lang.org

Editor

Editor

$\Rightarrow$ **Code...**

Patches not only for experts

Ruby, the Game

Endurance

Questions?