How Ruby Works?

- Mayra Cabrera

“Ruby is magical”

- *Every rubyist programmer in the world*

```
> puts “Hello World”

=> “Hello World”



> 1 + 1

=> 2
```

“Ruby is not magical”

Let’s explore Ruby internals…

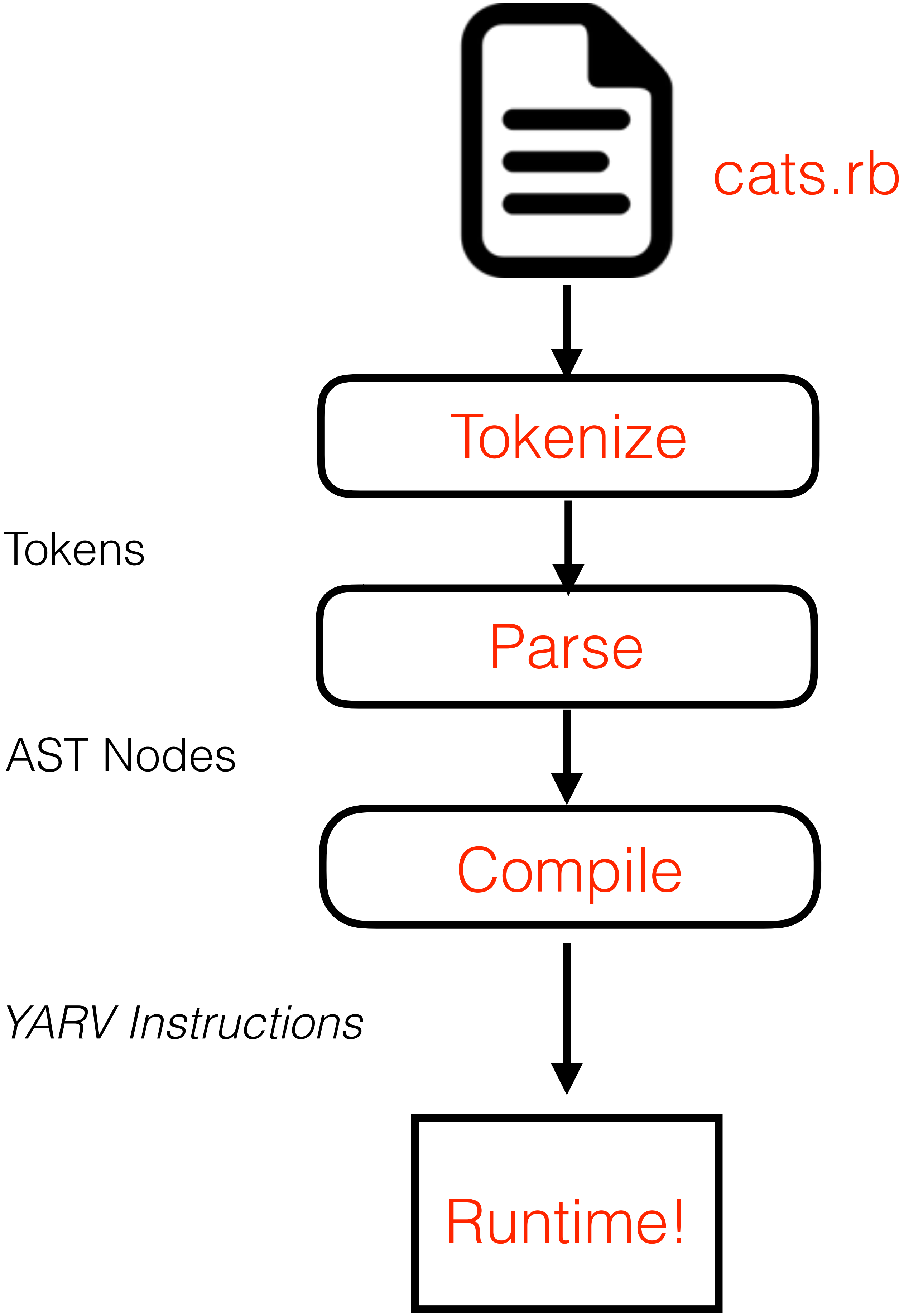
Stuff that we might know

- Ruby is written in C
- MRI (*Mat’z Ruby Interpreter*)

How many times do you think Ruby reads and transforms your code before running it?

🤔

3 times!



cats.rb

Tokenize

Tokens

Parse

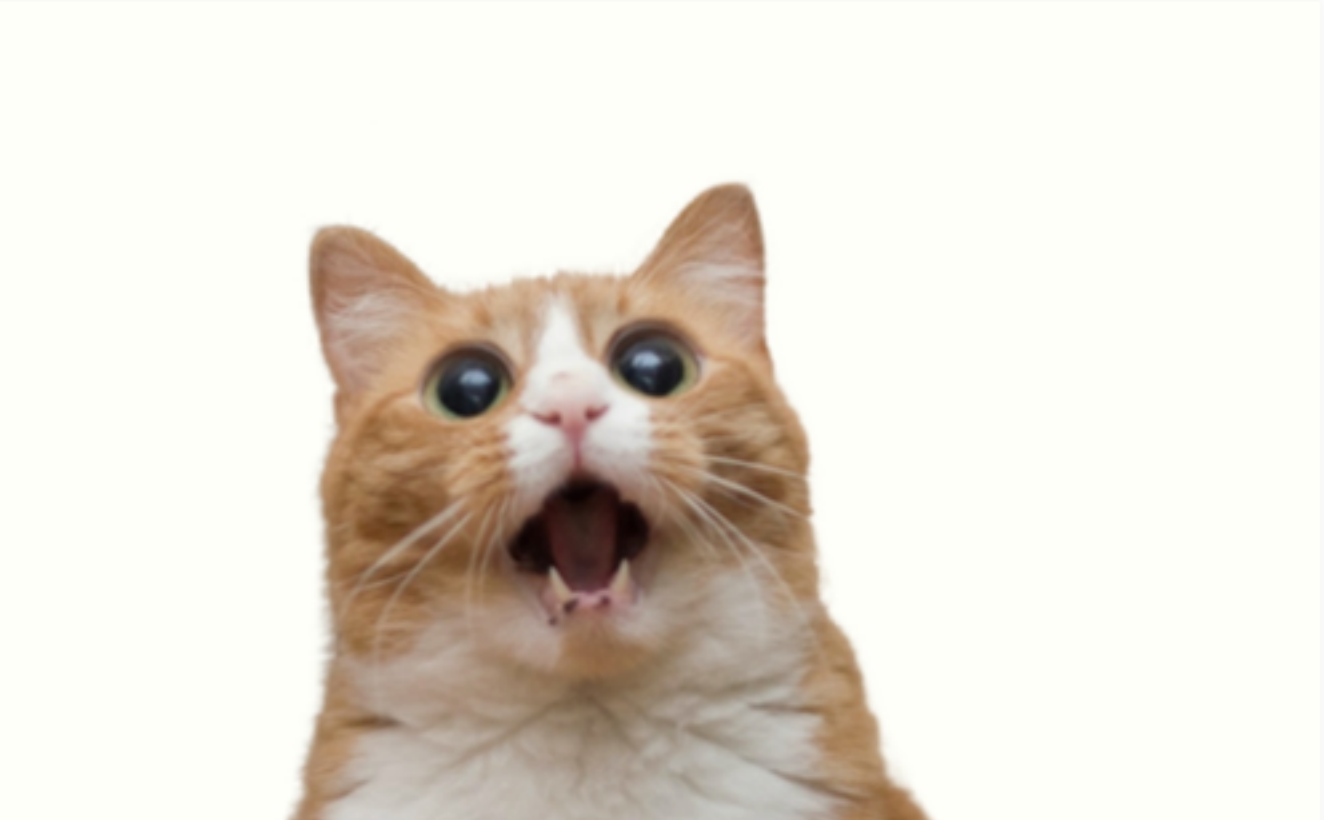
AST Nodes

Compile

YARV Instructions

Runtime!

Tokenize



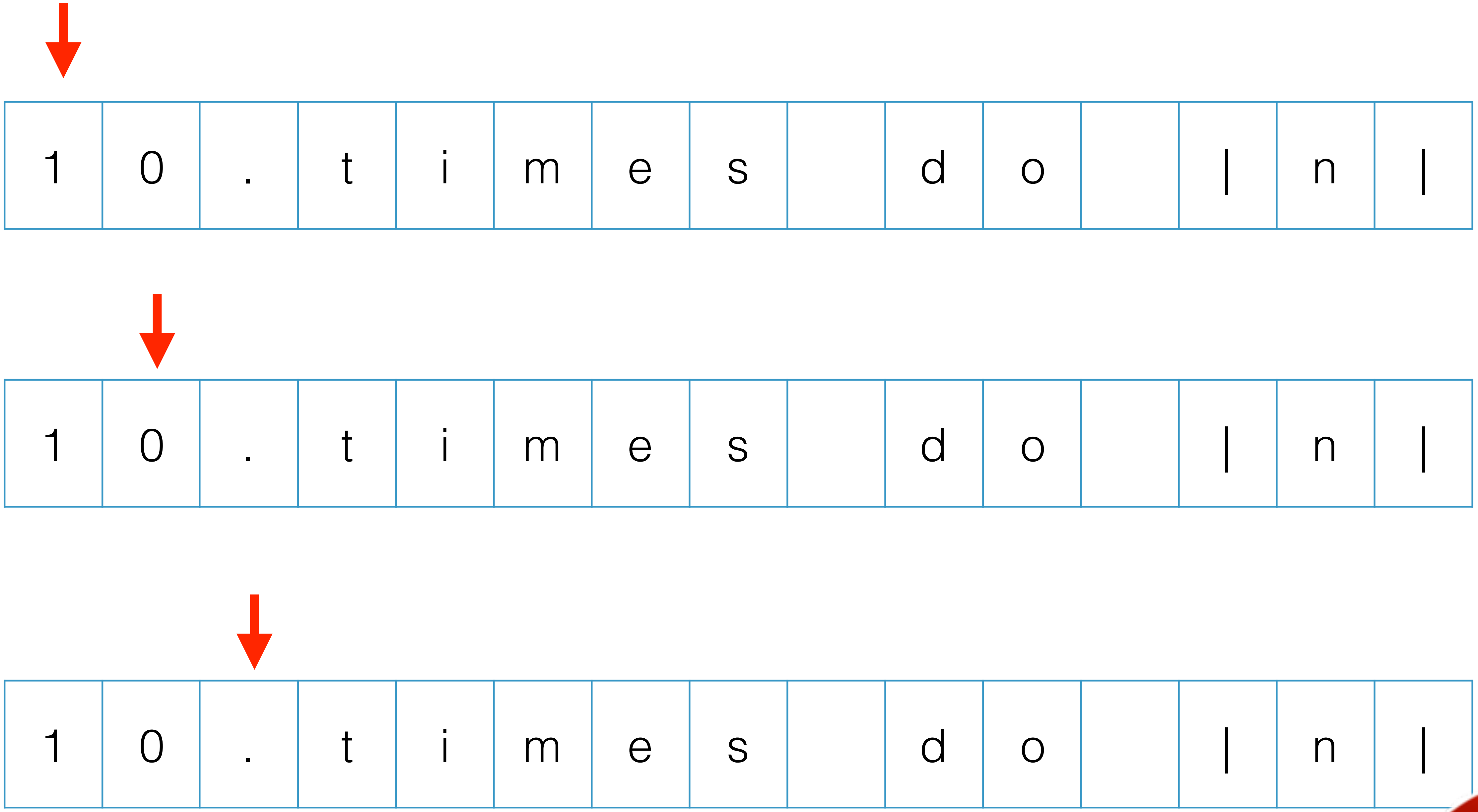
Tokenize

counter.rb

```
10.times do |n|
  puts n
end
```

```
> ruby counter.rb
1
2
3
...
10
```

Tokenize

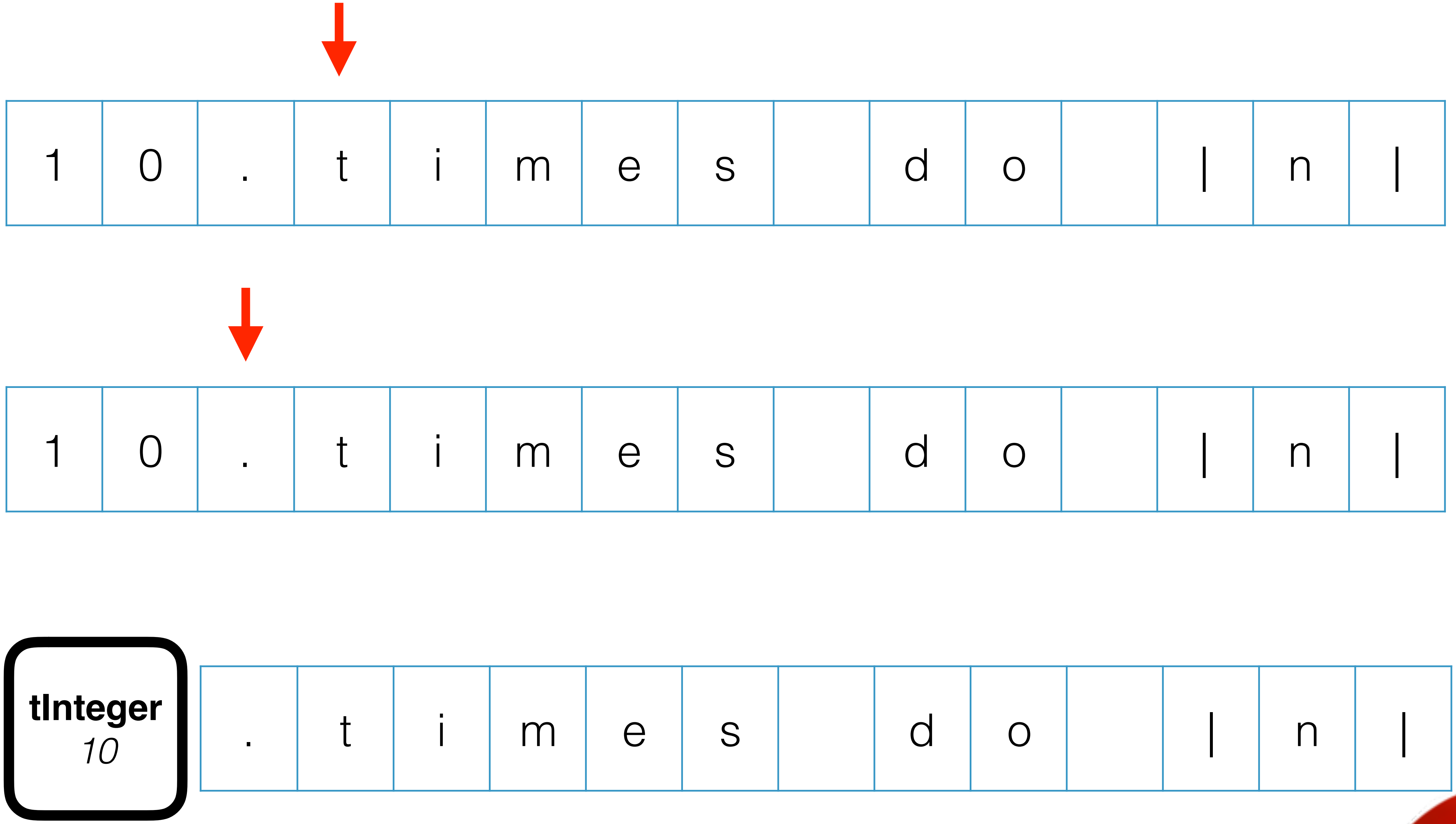


1	0	.	t	i	m	e	s		d	o		|	n	|

1	0	.	t	i	m	e	s		d	o		|	n	|

1	0	.	t	i	m	e	s		d	o		|	n	|

Tokenize

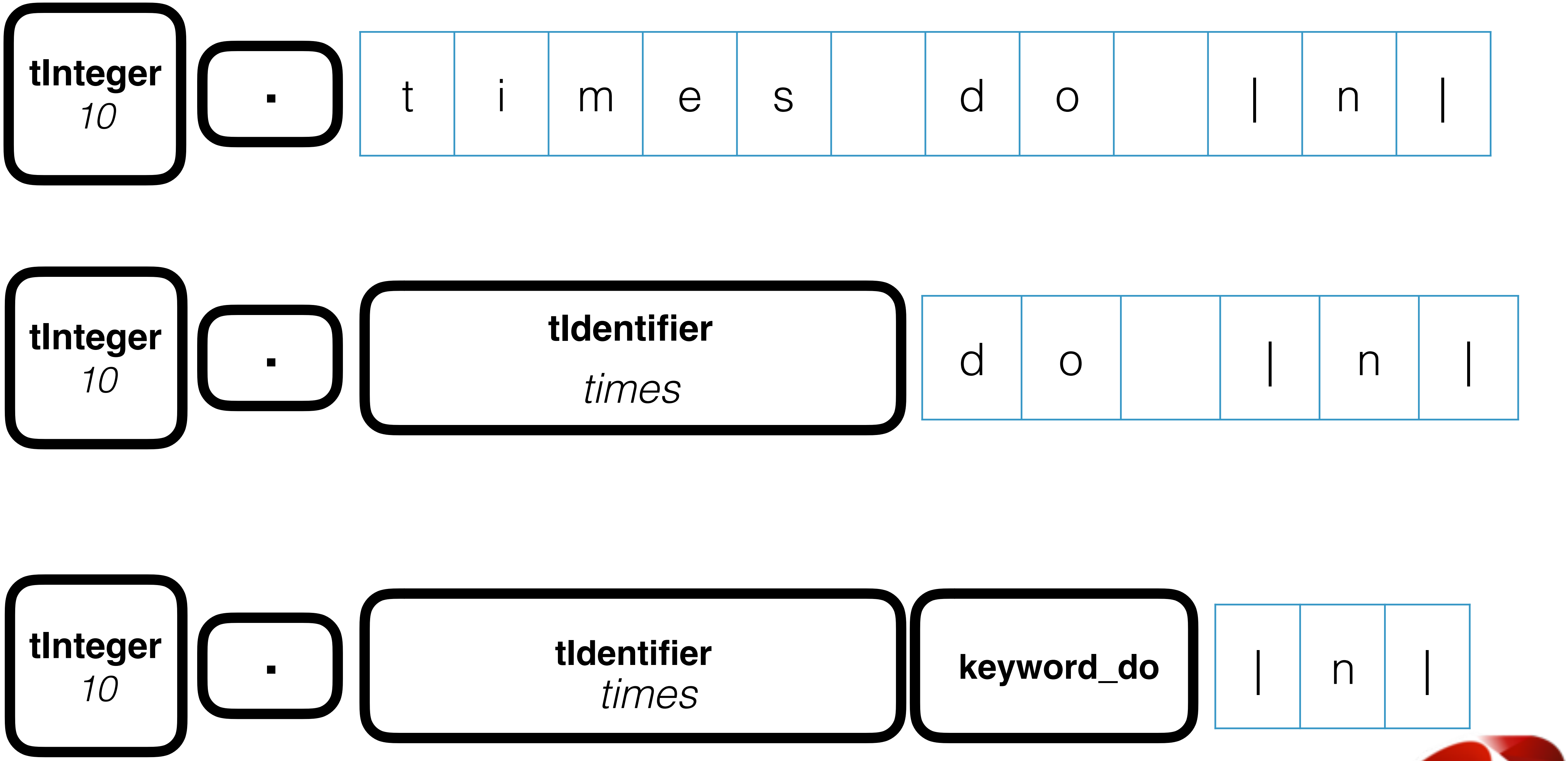


1	0	.	t	i	m	e	s		d	o		|	n	|

1	0	.	t	i	m	e	s		d	o		|	n	|

tInteger *10*	.	t	i	m	e	s		d	o		|	n	|

Tokenize

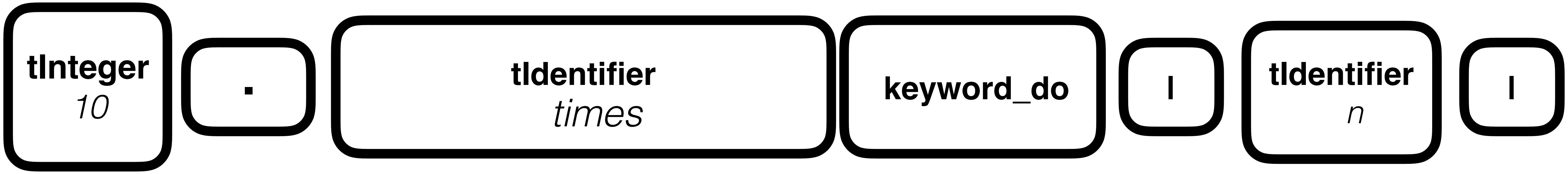


tInteger *10* | . | t | i | m | e | s | | d | o | | \| | n | \|

tInteger *10* | . | **tIdentifier** *times* | d | o | | \| | n | \|

tInteger *10* | . | **tIdentifier** *times* | **keyword_do** | \| | n | \|

Tokenize



tInteger
10

.

tIdentifier
times

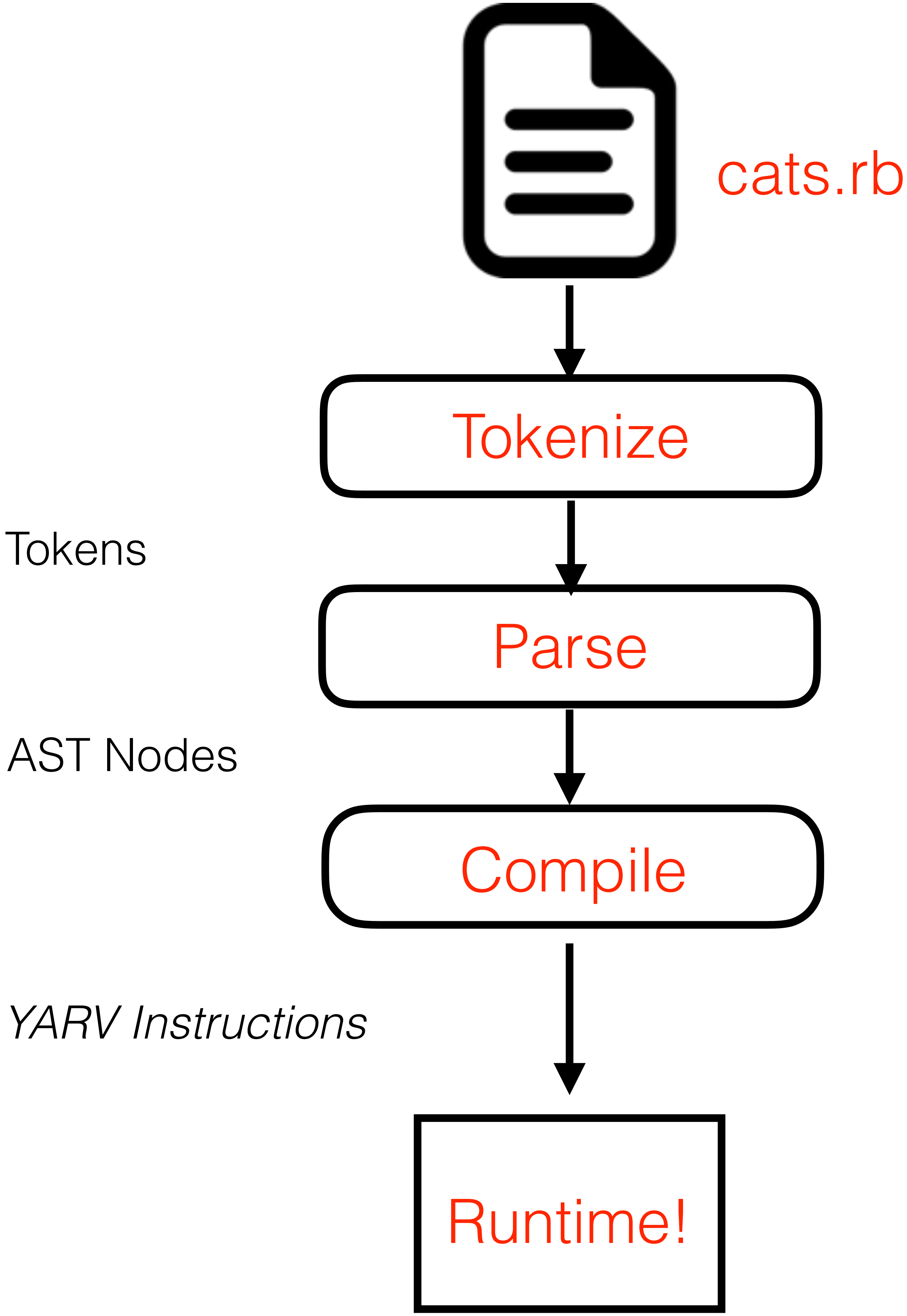
keyword_do

|

tIdentifier
n

|

What’s next?



cats.rb

Tokenize

Tokens

Parse

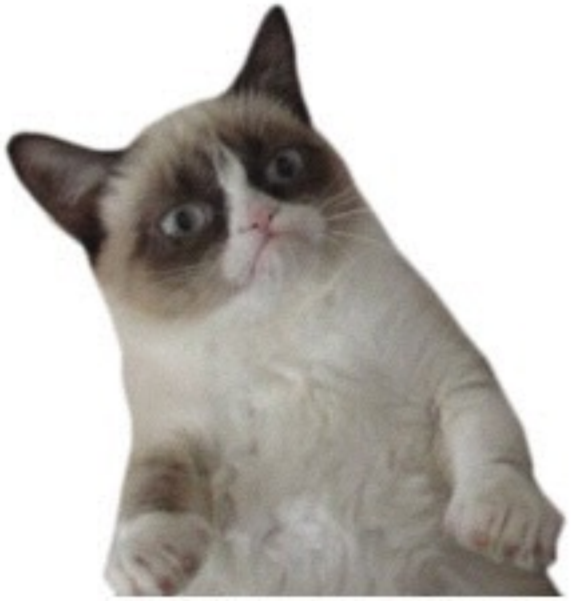
AST Nodes

Compile

YARV Instructions

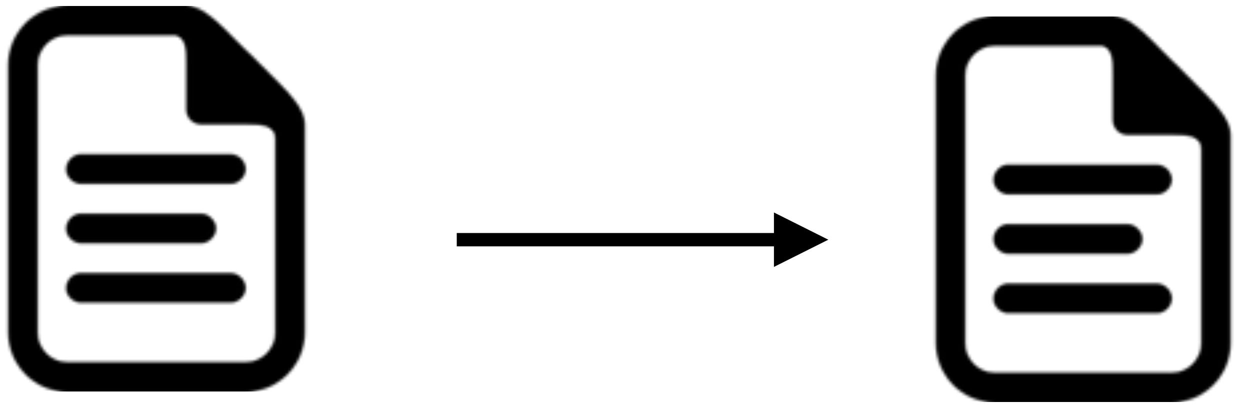
Runtime!

Parsing



Parsing

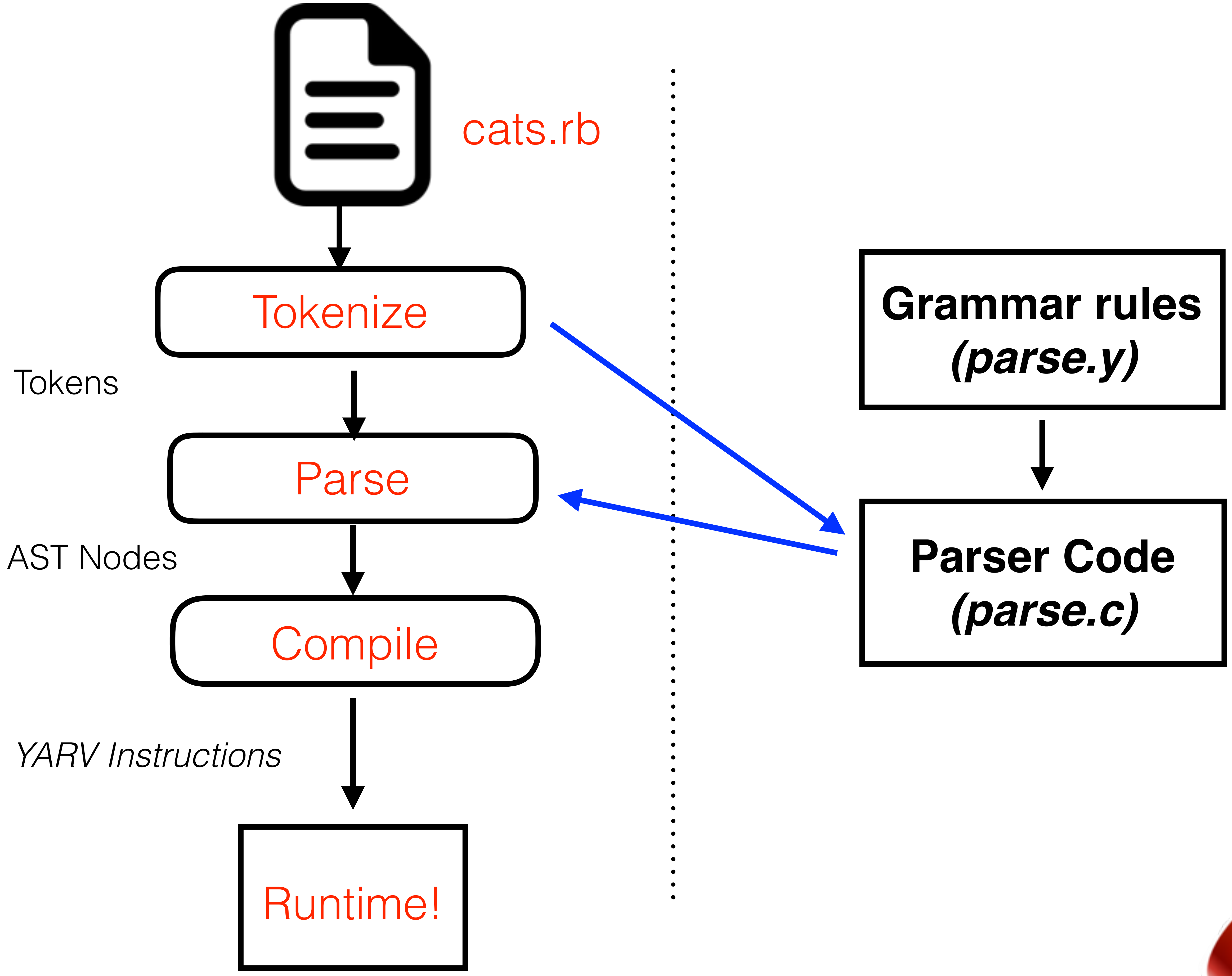
Ruby uses a parser to process tokens, but the parser itself is generated with a **parser generator**



parse.y

parse.c

Grammatical rules



cats.rb

Tokenize

Tokens

Parse

AST Nodes

Compile

YARV Instructions

Runtime!

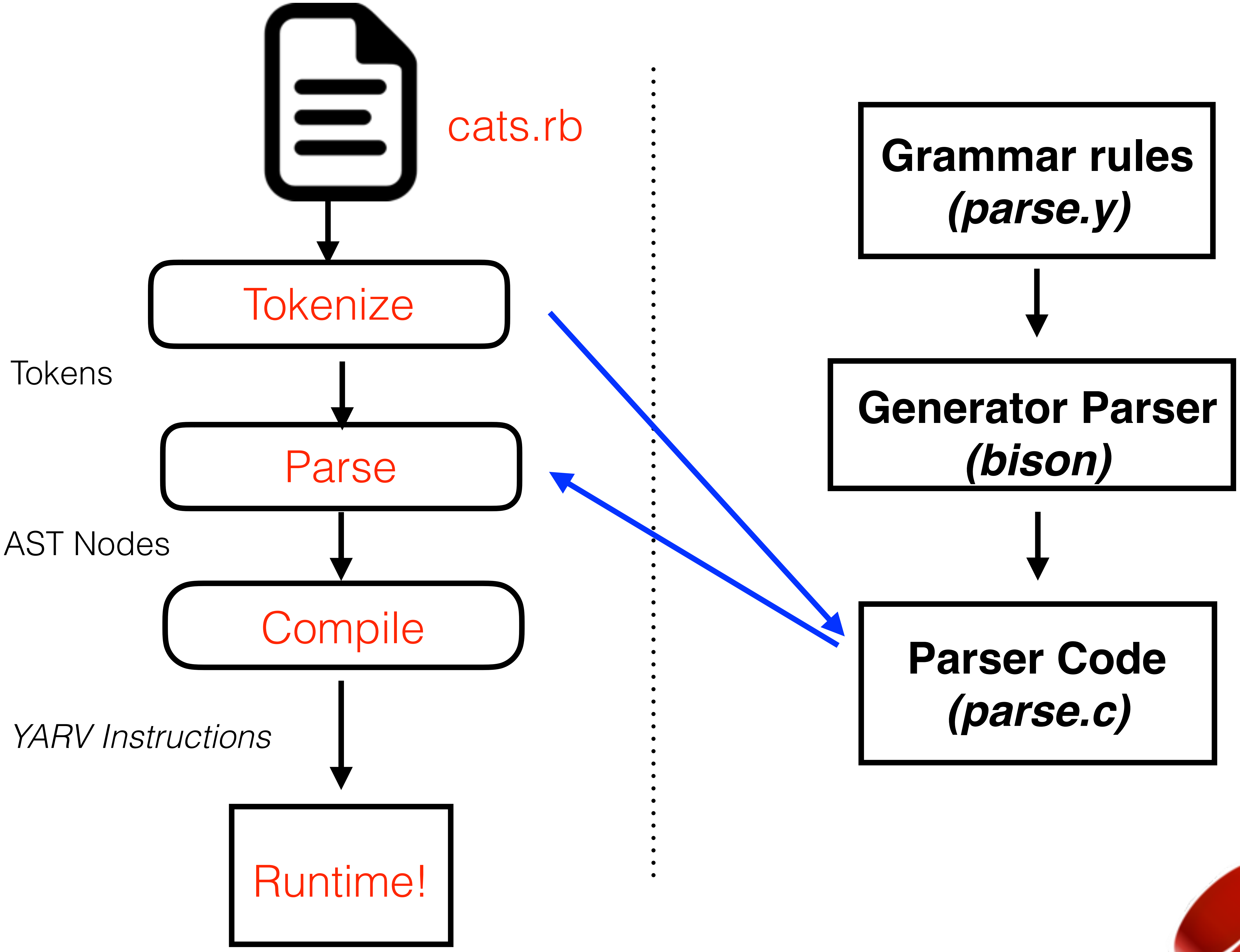
Grammar rules
(parse.y)

Parser Code
(parse.c)

Parsing

Yacc: *Yet Another Compiler Compiler*

Ruby uses a newer version of Yacc called Bison



cats.rb

Tokenize

Tokens

Parse

AST Nodes

Compile

YARV Instructions

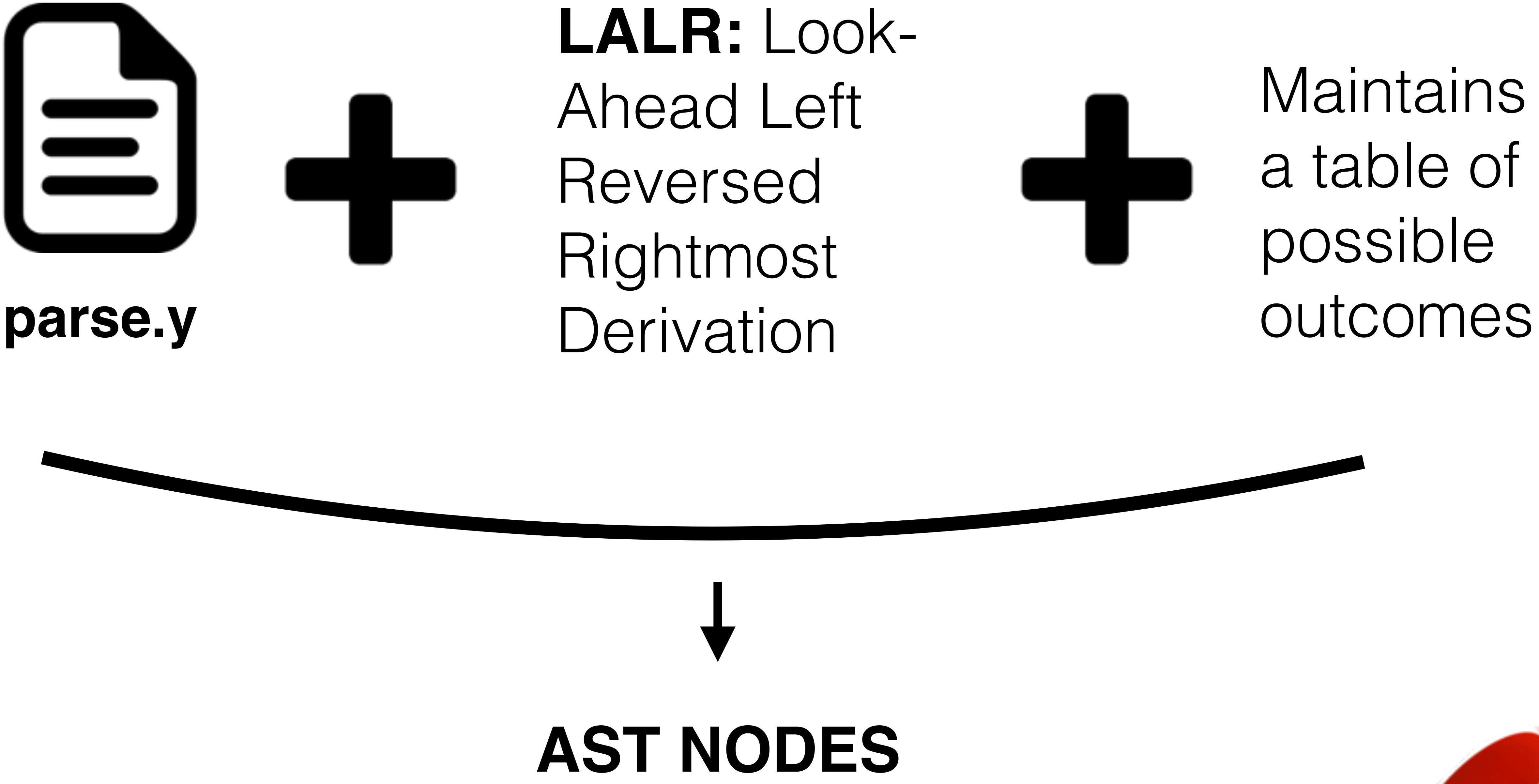
Runtime!

Grammar rules
(parse.y)

Generator Parser
(bison)

Parser Code
(parse.c)

Parsing



parse.y

+

LALR: Look-Ahead Left Reversed Rightmost Derivation

+

Maintains a table of possible outcomes

↓

AST NODES

Parsing

counter.rb

```
10.times do |n|
  puts n
end
```

Grammar rules

program: top_compstmt
top_compstmt: top_stmts opt_terms
top_stmts: ... l top_stmt l ...
top_stmt: stmt l ...
stmt: ... l expr
expr: ... l arg
arg: ... l primary
primary: ... l method_call brace_block l ...

Parsing

counter.rb

```ruby
10.times do |n|
  puts n
end
```

Grammar rules

```
program: top_compstmt
top_compstmt: top_stmts opt_terms
top_stmts: ... | top_stmt | ...
top_stmt: stmt | ...
stmt: ... | expr
expr: ... | arg
arg: ... | primary
primary: ... | method_call brace_block | ...
```

Parsing

counter.rb

```
10.times
```

Grammar rules

method_call: ... | primary_value '.' operation2 | ...

Table

primary_value	10
.	.
operation2	times

Parsing

counter.rb

```
do |n|
  puts n
end
```

Grammar rules

```
brace_block: ... l keyword_do opt_block_param
compstmt keyword_end l ...
```

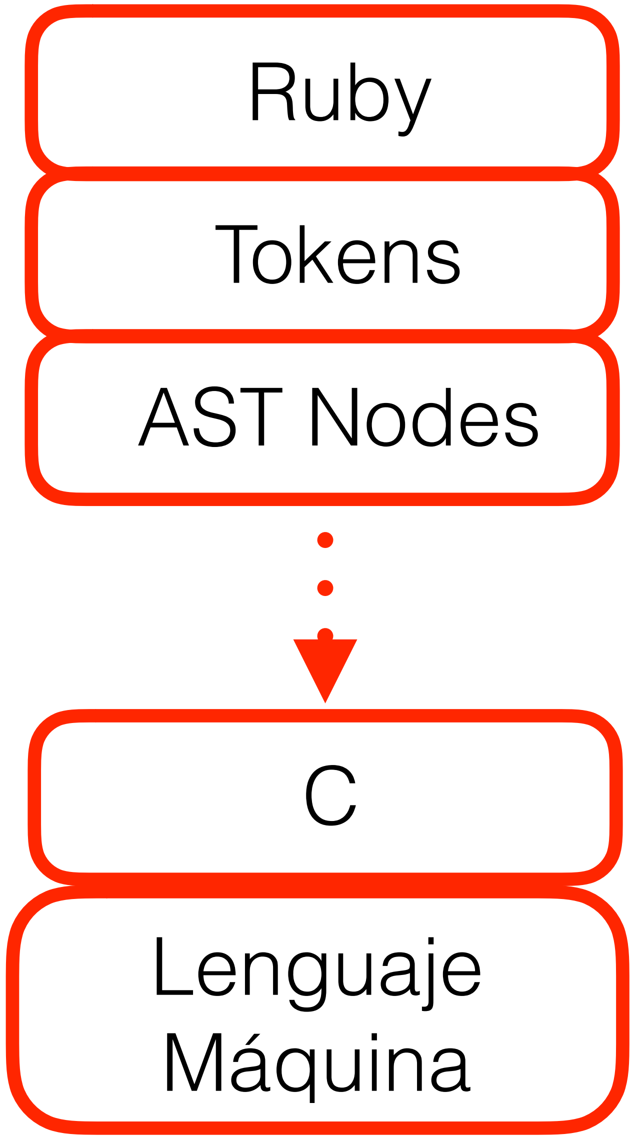
Table

keyword_do	do
opt_block_param	|n|
compstmt	puts n
keyword_end	end

Compilation

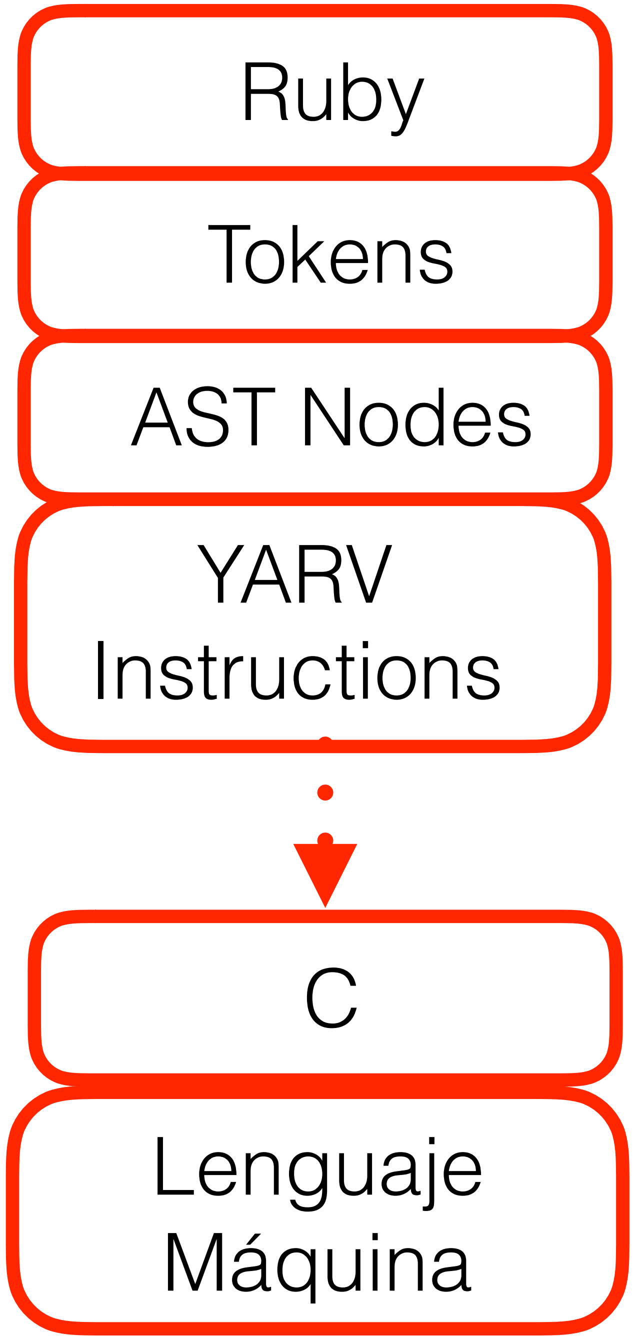
the final step…

Compilation



No Compiler for Ruby 1.8

Compilation



Ruby

Tokens

AST Nodes

YARV Instructions

C

Lenguaje Máquina

Ruby 1.9 introduce a Compiler: YARV (Yet Another Ruby Virtual Machine)

O

YARV: Stack Oriented Virtual Machine

```
puts 2 + 2
```

Taken from Ruby Under a Microscope

Compilation

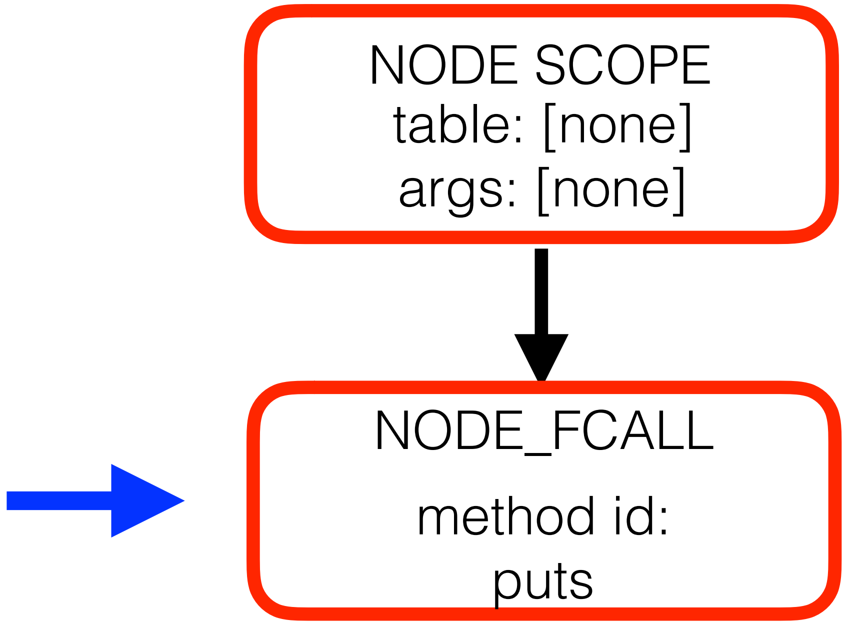
puts 2 + 2

NODE SCOPE
table: [none]
args: [none]

YARV Instructions

Compilation

puts 2 + 2



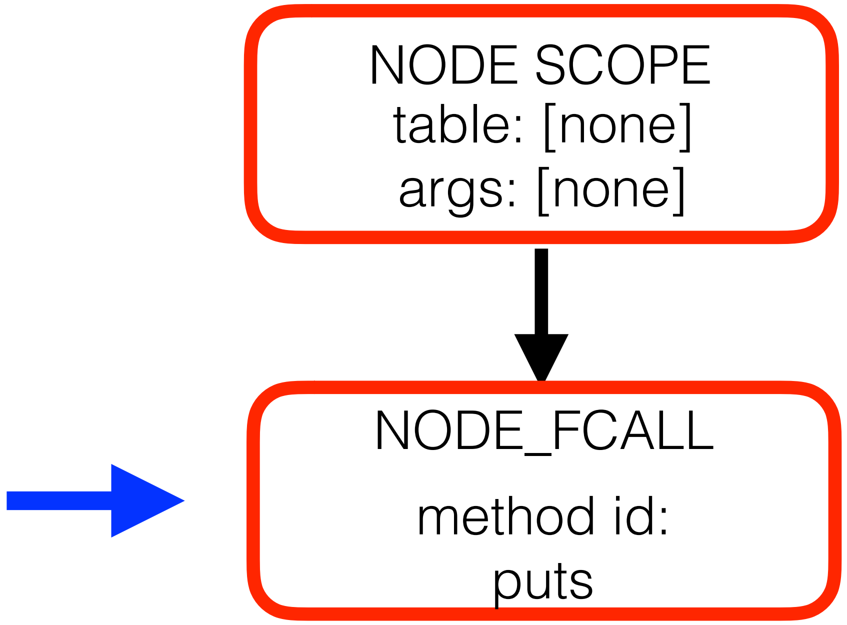
YARV Instructions

```
putself
putsobject 2
putsobject 2
send  <callinfo!mid:+, argc:1, ...
```

Taken from Ruby Under a Microscope

Compilation

puts 2 + 2



NODE SCOPE
table: [none]
args: [none]

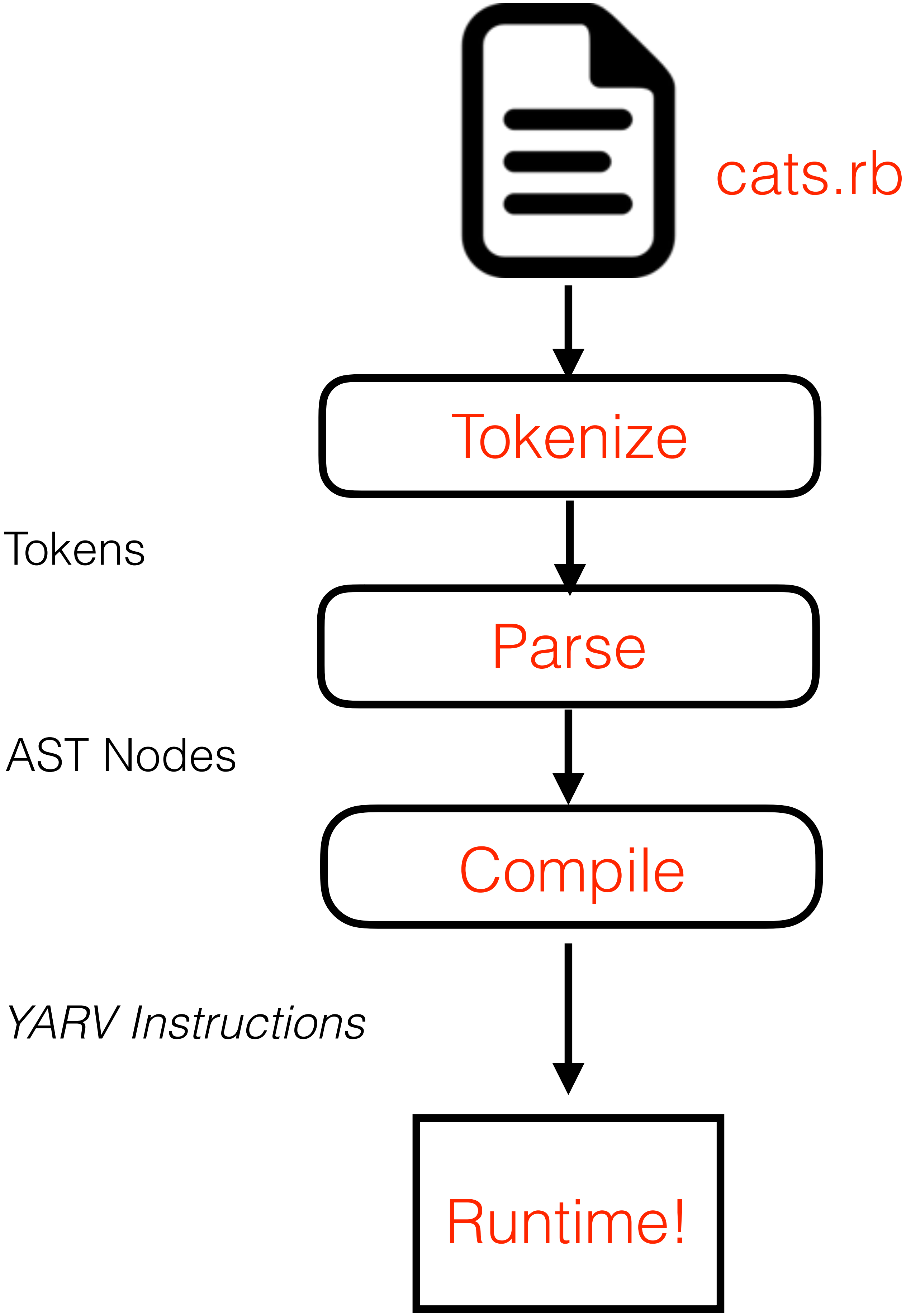
NODE_FCALL
method id:
puts

YARV Instructions

```
putself
putsobject 4

send <callinfo!mid:puts, argc:1, ...
```

Taken from Ruby Under a Microscope



cats.rb

Tokenize

Tokens

Parse

AST Nodes

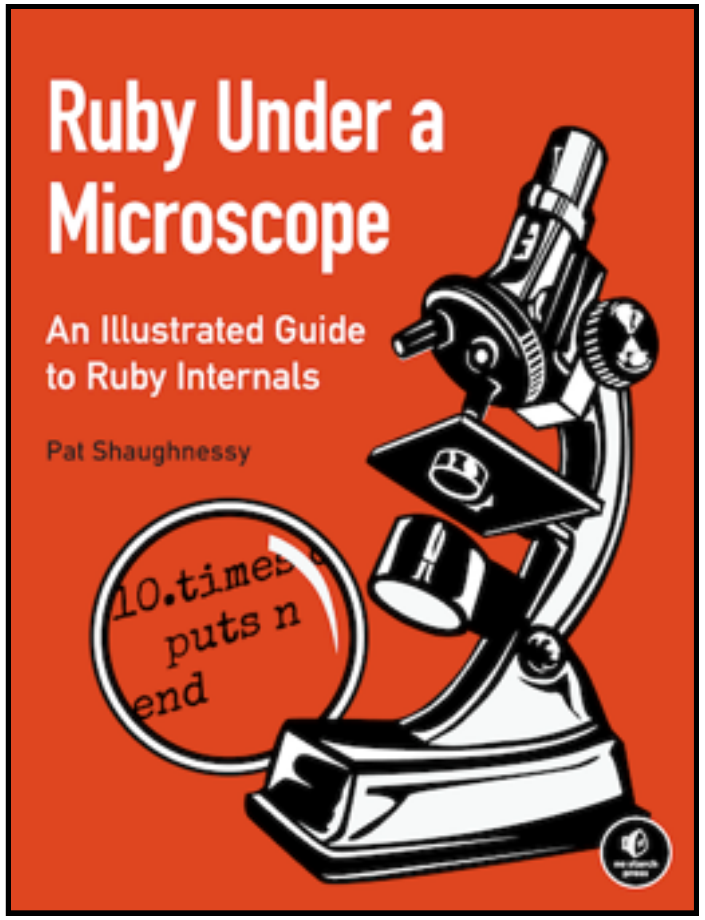
Compile

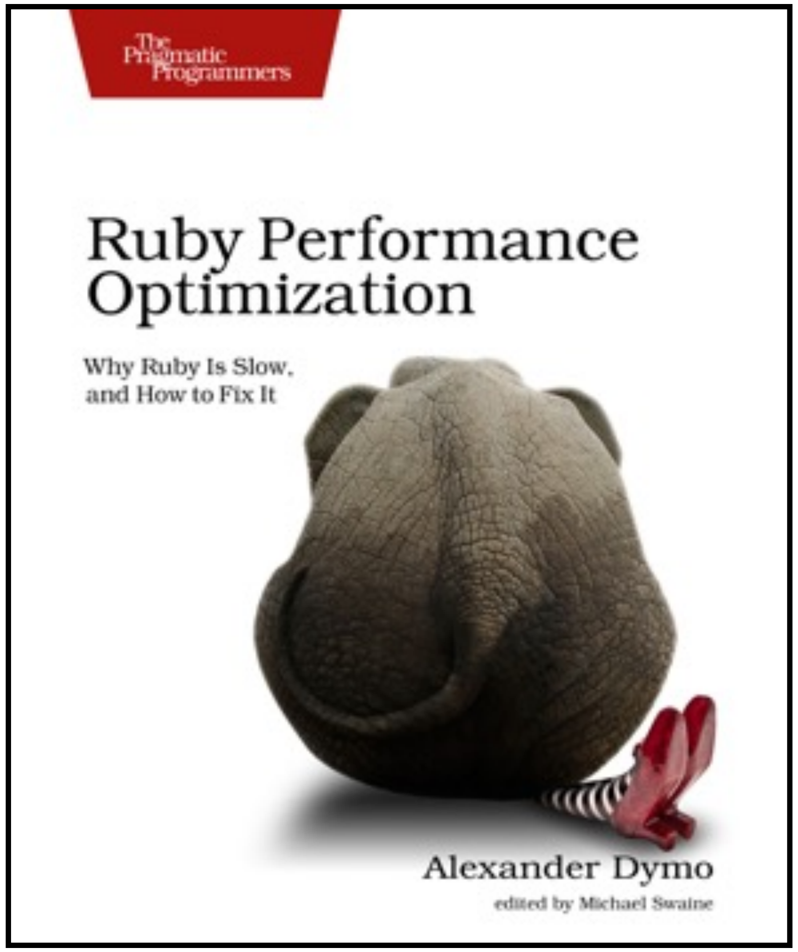
YARV Instructions

Runtime!

¿Preguntas?

Referencias




The Pragmatic Programmers
Ruby Performance Optimization
Why Ruby Is Slow, and How to Fix It
Alexander Dymo
edited by Michael Swaine

https://github.com/mayra-cabrera/ruby_meetup_11dic

Gracias!

may_cabrera

mayra-cabrera