

debug hard

```
require 'test/unit'

require_relative '../introduction'

class IntroductionTest < Test::Unit::TestCase

  def test_introduction

    assert_equal(
      'Welcome to RubyConf Thailand',
      Introduction.new.method

    )

  end

end
```

```
test/introduction_test.rb:2:in `require_relative':  
cannot load such file -  
/Users/vishalchandnani/Technical-Talks/idea_1  
/introduction/introduction (LoadError)  
from test/introduction_test.rb:2:in `<main>'
```

```
class Introduction

def method
  'Welcome to RubyConf Thailand'
end

end
```

Finished in 0.000394 seconds.

1 tests, 1 assertions, 0 failures, 0 errors

0 pendings, 0 omissions, 0 notifications

100% passed

```
def method  
  'Welcome to RubyConf Thailand'  
end
```



no bugs were harmed in the making of this talk

once upon a time

0800 Arctan started
 1000 " stopped - arctan ✓
 1300 (032) MP-MC
 (033) PRO 2
 Relays 6-2 in 033 failed special speed test
 in relay

{ 1.2700 9.037847025
 9.037846995 const
~~1.98267000~~
~~2.130476415~~ 4.615925059(-2)
 2.130676415

Relay
 2145
 Relay 3370

1100 Started Cosine Tape (Sine check)
 1525 Started Multi Adder Test.

1545



Relay #70 Panel F
(moth) in relay.

First actual case of bug being found.
 1630 Arctangent started.
 1700 closed down.







<https://www.ruby-lang.org/en/documentation/installation/>

<https://www.ruby-lang.org/en/downloads/>

Ruby Version: 2.5.1

ruby . c

#1
grep

```
$ grep -r rb_str_reverse *
```

string.c

NAT COHEN PRESENTS FOR EMI FILM DISTRIBUTORS LTD. A JOHN BRADOURNE-RICHARD GOODWIN PRODUCTION

AGATHA CHRISTIE's
MURDER ON THE ORIENT EXPRESS



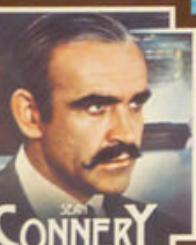
ALBERT
FINNEY



MARTIN
BALSAM



INGRID
BERGMAN



SEAN
CONNERY



JOHN
IELGUD



VANESSA
REDGRAVE



RACHEL
ROBERTS



LAUREN
BAKALL



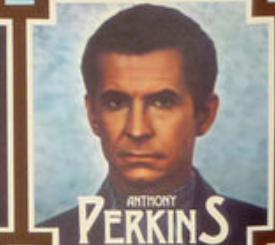
JACQUELINE
BISET



JEAN-PIERRE
CASSEL



WENDY
HILLER



ANTHONY
PERKINS



RICHARD
WIDMARK



MICHAEL
YORK

File: test/ruby/test_string.rb

```
require 'test/unit'
```

```
class TestString < Test::Unit::TestCase
```

```
  def test_reverse
```

```
    assert_equal(S("beta"), S("ateb").reverse)
```

```
    assert_equal(S("madamImadam"), S("madamImadam").reverse)
```

```
    a=S("beta")
```

```
    assert_equal(S("ateb"), a.reverse)
```

```
    assert_equal(S("beta"), a)
```

```
  end
```

```
end
```

File: string.c

```
/*
 *  call-seq:
 *
 *      str.reverse. -> new_str
 *
 *  Returns a new string with the characters from <i>str</i>
 *
 *  in reverse order.
 *
 *  "stressed".reverse  #=> "desserts"
 */

static VALUE
rb_str_reverse(VALUE str)
{
    . . .
}
```

all you need to do

3 days later

debugger.rb

Raphaël



ieahpaR

#2
chars

“Raphaël”.chars

["R", "a", "p", "h", "a", "e", ü, "l"]

ë

unicode





**Twelve terrorists. One cop.
The odds are against John McClane...
That's just the way he likes it.**

BRUCE WILLIS
DIE HARD

#3
codepoints

```
“Raphaël”.codepoints do |c|
  puts "#{c} : 0x#{c.to_s(16)}"
end
```

"R" : 82 (0x52)
"a" : 97 (0x61)
"p" : 112 (0x70)
"h" : 104 (0x68)
"a" : 97 (0x61)
"e" : 101 (0x65)
"i" : 776 (0x308)
"l" : 108 (0x6c)

#4
each_byte

```
"Raphaël".each_byte do |c|
  puts "#{c} : 0x#{c.to_s(16)}"
end
```

“R” : 82 (0x52)
“a” : 97 (0x61)
“p” : 112 (0x70)
“h” : 104 (0x68)
“a” : 97 (0x61)
“e” : 101 (0x65)
“i” : 204 (0xcc) and 136 (0x88)
“l” : 108 (0x6c)

pointers



```
#include <stdio.h>
#include <string.h>

int main ( void ) {
    char str[25] = "hello world";
    char *ptr;

    for(ptr = str; *ptr != '\0'; ptr++) {
        printf("%c", *ptr);
    }
}
```

#5
printf

```
printf("\n length: %d", length);
```

```
static VALUE  
rb_str_reverse(VALUE str)  
{  
    . . .  
    while (s < e) {  
        int clen = rb_enc_fast_mbcrlen(s, e, enc);  
        printf("\nclen: %d", clen);  
        p -= clen;  
        memcpy(p, s, clen);  
        s += clen;  
    }  
    . . .  
}
```



#6
gdb

```
$ gdb ruby  
(gdb) break string.c:5575  
(gdb) run -Ilib debugger.rb  
(gdb) s
```

```
$ gdb ruby
(gdb) break string.c:5575
(gdb) break regenc.c:62
(gdb) run -Ilib debugger.rb
(gdb) s
```



File: enc/utf_8.c

```
static int  
mbc_enc_len(  
const UChar* p,  
const UChar* e,  
OnigEncoding enc ARG_UNUSED) {  
    . . .  
    . . .  
}
```

s: R 82 52

 clen: 1

s: a 97 61

 clen: 1

s: p 112 70

 clen: 1

s: h 104 68

 clen: 1

s: a 97 61

 clen: 1

s: e 101 65

 clen: 1

s: X -52 ffffffcc

 clen: 2

s: l 108 6c

 clen: 1

```
if(c1 == 'e') {  
    vptr = s;  
    *vptr = 0xc3;  
    vptr++;  
    *vptr = 0xab;  
    vptr++;  
    *vptr = 'l';  
    vptr++;  
    *vptr = '\0';  
}
```

s: R 82 52

 clen: 1

s: a 97 61

 clen: 1

s: p 112 70

 clen: 1

s: h 104 68

 clen: 1

s: a 97 61

 clen: 1

s: e 101 65

 hack

 clen: 2

s: l 108 6c

 clen: 1

lëahpaR



unicode_normalize

Canonical Composition

```
irb> vc = "e\u0308".unicode_normalize  
=> "ë"  
  
irb> vc.chars  
=> ["ë"]  
  
irb> vc.size  
=> 1
```

Canonical Decomposition

```
irb> vc = "e\u0308".unicode_normalize
=> "ë"
irb> vc.unicode_normalize(:nfd).chars
=> ["e", "ï"]
irb> vc.unicode_normalize(:nfd).chars.size
=> 2
```

chars

before

"R"

"a"

"p"

"h"

"a"

"e"

"i"

"l"

chars

after

"R"

"a"

"p"

"h"

"a"

"ë"

"l"

code_points

before

"R" : 82

"a" : 97

"p" : 112

"h" : 104

"a" : 97

"e" : 101

"ü" : 776

"l" : 108

code_points

after

"R" : 82

"a" : 97

"p" : 112

"h" : 104

"a" : 97

"ë" : 235

"l" : 108

`each_byte`

`before`

`"R" : 82 (0x52)`

`"a" : 97 (0x61)`

`"p" : 112 (0x70)`

`"h" : 104 (0x68)`

`"a" : 97 (0x61)`

`"e" : 101 (0x65)`

`"ü" : 204 (0xcc) and 136 (0x88)`

`"l" : 108 (0x6c)`

`each_byte`

`after`

`"R" : 82 (0x52)`

`"a" : 97 (0x61)`

`"p" : 112 (0x70)`

`"h" : 104 (0x68)`

`"a" : 97. (0x61)`

`"ë" : 195 (0xc3) and 171(0xab)`

`"l" : 108 (0x6c)`

File: lib/unicode_normalize/normalize.rb

```
def self.normalize(string, form = :nfc)
  ...
  ...
  case form
    when :nfc then
      string.gsub REGEXP_C, NF_HASH_C
      ...
    ...
  end
```

```
m1 = string[REGEXP_C]  
regex match: "é"  
chars: ["e", "\u00e9"]  
codepoints: [101, 776]  
each byte iteration  
101:0x65 | 204:0xcc | 136:0x88
```

```
m2 = NF_HASH_C[m1]  
hash value: "é"  
chars: ["é"]  
codepoints: [235]  
each byte iteration  
195:0xc3 | 171:0xab
```

Raphaël

lëahpaR

```
def debug_hard  
    grep  
    chars  
    code_points  
    each_byte  
    printf  
    gdb  
end
```

Email: vschandnani@gmail.com

Twitter: [@vschandnani](https://twitter.com/vschandnani)

GitHub: <https://github.com/vchandnani>

LinkedIn: <https://www.linkedin.com/in/vchandnani/>