

Ruby išplėtimai C kalba

Pranas Kiziela

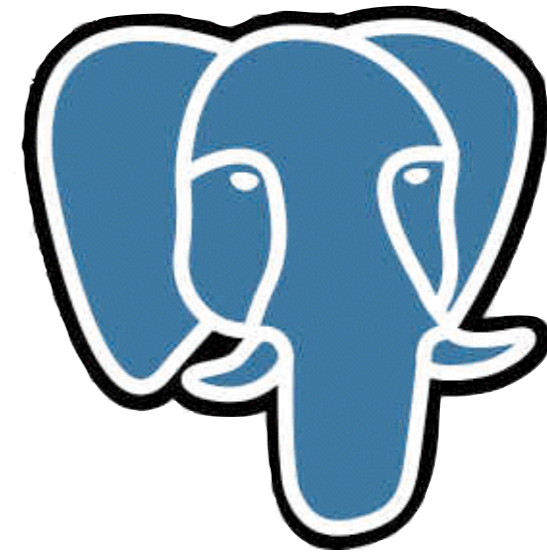
- Motyvācija
- Išplētimo būdai
- Ruby ir C integrācija
- Gijos
- Atminties valdymas C kalboje
- Rēsumē

Motyvacija

- Kodo perpanaudojimas
- Geležies ir OS API
- Greitis



PostgreSQL



SQLite



README.rdoc

home_run

home_run is an implementation of ruby's Date/DateTime classes in C, with much better performance (20-200x) than the version in the standard library, while being almost completely compatible.

Performance increase (microbenchmarks)

The speedup you'll get depends mostly on your version of ruby, but also on your operating system, platform, and compiler. Here are some comparative results for common methods:

#	i386	i386	i386	i386	amd64
#	Windows	Linux	Linux	Linux	OpenBSD
#	1.8.6	1.8.7	1.9.1	1.9.2	1.9.2
#	-----	-----	-----	-----	-----
Date.civil	82x	66x	27x	21x	14x
Date.parse	56x	56x	33x	30x	25x
Date.today	17x	6x	2x	2x	2x
Date.strptime	43x	62x	63x	37x	23x
DateTime.civil	252x	146x	52x	41x	17x
DateTime.parse	52x	54x	32x	27x	20x
DateTime.now	78x	35x	11x	8x	4x
DateTime.strptime	63x	71x	58x	35x	23x
Date#strftime	156x	104x	110x	70x	62x
Date#+	34x	32x	5x	5x	4x
Date#<<	177x	220x	86x	72x	40x
Date#to_s	15x	6x	5x	4x	2x
DateTime#strftime	146x	107x	114x	71x	60x
DateTime#+	34x	37x	8x	6x	3x
DateTime#<<	88x	106x	40x	33x	16x
DateTime#to_s	144x	47x	54x	29x	24x

- Tradiciniai MRI C išplėtimai
- Ruby/DL ir Ruby-FFI

Tradicioniniai MRI C išplėtimai

- Rašomi C kalba (API primena Ruby)
- Kompilijuojami naudojo kompiuteryje
- Dažniausiai naudojami
- 1.8 ir 1.9 nepilnai suderinami
- Tik dalinai suderinami Rubinius, JRuby

```
#include "ruby.h"

VALUE MyTest = Qnil;

void Init_mytest();

VALUE method_test1(VALUE self);

void Init_mytest() {
    MyTest = rb_define_module("MyTest");
    rb_define_method(MyTest, "test1", method_test1, 0);
}

VALUE method_test1(VALUE self) {
    int x = 10;
    return INT2NUM(x);
}
```

Ruby/DL

- "Interface to the Dynamic Loader, for creating extensions."
- Ruby standartinė biblioteka
- Nedokumentuota
- Nebenaudojama
- "Archaiškos" sintaksės
- Nuo 1.9.2 DL naudoja libffi

```
require 'dl/import'
```

```
module LIBC
```

```
  extend DL::Importable
```

```
  dlopen "libc.so"
```

```
  extern "int strlen(const char *)"
```

```
end
```

```
LIBC.strlen("abc") # => 3
```

ruby-ffi

“It's all Unicorns and Rainbows in theory. It'll be pure Ruby! It's cross platform! You can use it with JRuby and Rubinius!”

- Konvertuoja duomenis
- libffi interfeisas
- MRI realizuotas kaip tradicinis C išplėtimas
- Suteikia gražų DSL
- MRI, JRuby, Rubinius, MacRuby

libsomething_foo()
kvietimas

FFI

libsomething.so

ruby-ffi code

```
module Tidy
  extend FFI::Library

  ffi_lib "libtidy.dylib"

  attach_function :tidyFileExists, [:string], :int
  attach_function :tidyCreate, [], :pointer
  attach_function :tidyParseString, [:pointer, :string], :int
  attach_function :tidySaveStdout, [:pointer], :int
end

tdoc = Tidy.tidyCreate
Tidy.tidyParseString tdoc, "<title>Foo</title>"
Tidy.tidySaveStdout tdoc
```

Tradicioniniai išplėtimai

- Naudojami
 - Kai duomenų konvertavimas per FFI sukuria per didelį “overhead”
- Problemos
 - Jautrūs vidinio Ruby API pasikeitimams
 - Tiesioginis (nenaudojant API) priėjimas prie duomenų (1.9 API buvo išplėstas stengiantis išspręsti šią problemą)

FFI

- Naudojimas
 - Kai norima palaikyti įvairius Ruby interpretatoriaus
 - “Greitam priėjimui prie bibliotekų”
- Problemos
 - Skirtingos bibliotekų versijos (paprastai išsprendžiama kompiliuojant)

Kaip tai veikia?

extconf.rb

```
require 'mkmf'
```

```
create_makefile('test')
```

test.c

```
// Ruby API aprašai ir MACROS
#include "ruby.h"

// VALUE Ruby objekto duomenų tipas
VALUE Test = Qnil;

// funkcijų prototipai
void Init_test();
VALUE method_test1(VALUE self);

// pakrovimo metodas
void Init_test() {
    Test = rb_define_module("Test");
    rb_define_method(Test, "test1", method_test1, 0);
}

VALUE method_test1(VALUE self) {
    int x = 10;
    return INT2NUM(x);
}
```

ruby.h

- INT2NUM(int) -> Fixnum or Bignum
- INT2FIX(int) -> Fixnum (faster)
- INT2NUM(long or int) -> Fixnum or Bignum
- INT2FIX(long or int) -> Fixnum (faster)
- CHR2FIX(char) -> Fixnum
- rb_str_new2(char *) -> String
- rb_float_new(double) -> Float

ruby.h

- `int NUM2INT(Numeric)` (Includes type check)
- `int FIX2INT(Fixnum)` (Faster)
- `unsigned int NUM2UINT(Numeric)` (Includes type check)
- `unsigned int FIX2UINT(Fixnum)` (Includes type check)
- `long NUM2LONG(Numeric)` (Includes type check)
- `long FIX2LONG(Fixnum)` (Faster)
- `unsigned long NUM2ULONG(Numeric)` (Includes type check)
- `char NUM2CHR(Numeric orString)` (Includes type check)
- `char * STR2CSTR(String)`

ruby.h

```
void rb_define_method(VALUE classmod, char *name, VALUE(*func)(), int argc")
void rb_define_module_function(VALUE classmod, char *name, VALUE(*func)(), int argc")
void rb_define_global_function(char *name, VALUE(*func)(), int argc")
void rb_define_singleton_method(VALUE classmod, char *name, VALUE(*func)(), int argc")
void rb_undef_method(VALUE classmod, const char *name")
void rb_define_alias(VALUE classmod, const char *newname, const char *oldname")
VALUE rb_ary_new("")
VALUE rb_ary_new2(long length")
VALUE rb_ary_new3(long length, ...")
VALUE rb_ary_new4(long length, VALUE *values")
void rb_ary_store(VALUE self, long index, VALUE value")
VALUE rb_ary_push(VALUE self, VALUE value")
...
rb_eval_string("anObject.each{|x| x.clearFlag }");
```

```
~/test> ruby extconf.rb
```

```
creating Makefile
```

```
~/test> make
```

```
gcc -I. -I/Users/pranas/.rvm/rubies/ruby-1.8.7-p334/lib/ruby/1.8/i686-darwin10.7.0 -I/Users/pranas/.rvm/rubies/ruby-1.8.7-p334/lib/ruby/1.8/i686-darwin10.7.0 -I. -D_XOPEN_SOURCE -D_DARWIN_C_SOURCE -fno-common -g -O2 -fno-common -pipe -fno-common -c test.c
```

```
cc -dynamic -bundle -undefined suppress -flat_namespace -o test.bundle test.o -L. -L/Users/pranas/.rvm/rubies/ruby-1.8.7-p334/lib -L. -lruby -ldl -lobjc
```

```
~/test> ls
```

```
Makefile  extconf.rb  test.bundle test.c  test.o
```

```
~/test> otool -L test.bundle
```

```
test.bundle:
```

```
    /Users/pranas/.rvm/rubies/ruby-1.8.7-p334/lib/libruby.dylib (compatibility version 1.8.0, current version 1.8.7)
```

```
    /usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 125.2.10)
```

```
    /usr/lib/libobjc.A.dylib (compatibility version 1.0.0, current version 227.0.0)
```

```
irb(main):001:0> require 'test'
```

```
=> true
```

```
irb(main):002:0> include Test
```

```
=> Object
```

```
irb(main):003:0> puts test1
```

```
10
```

Gijos

“Žalios” gijos

```
void rb_thread_start_timer() {
    ...
    signal(SIGVTALRM, catch_timer);
    ...
    tval.it_interval.tv_usec = 10000;
    setitimer(ITIMER_VIRTUAL, &tval, NULL);
    ...
} //eval.c

static void catch_timer(sig) {
    if (!rb_thread_critical) {
        rb_thread_pending = 1;
    }
} //eval.c

#define CHECK_INTS ... if (rb_thread_pending) rb_thread_schedule(); ...
```

- Besiblokuojantys kvietimai užblokuos visas gijas
- Vidinių interpretatoriaus metodų kvietimas gali sąlygoti valdymo perdavimą kitai gijai

Kaip išvengti užsiblokavimo?

- `rb_thread_wait_fd`
- `rb_thread_fd_writable`
- `rb_thread_polling`
- `rb_thread_wait_for`
- `rb_thread_schedule`

Ruby 1.9

- OS gijos
- Giant VM Lock

Ruby 1.9

- `rb_thread_blocking_region`

Atminties valdymas

- `ALLOC(void)`
- `ALLOC_N(void, n)`
- `REALLOC_N(k, void, m)`
- `xfree(k)`

```
#define GC_MALLOC_LIMIT 8000000
static unsigned long malloc_limit = GC_MALLOC_LIMIT;

void * ruby_xmalloc(size) {
    ...
    if (ruby_gc_stress || (malloc_increase+size) >
malloc_limit) {
        garbage_collect();
    }
    ...
} // gc.c
```

```
int* ids;
```

```
VALUE result;
```

```
int i;
```

```
ids=ALLOC_N(int,5);
```

```
for(i=0; i<5; i++) ids[i] = i;
```

```
result = rb_ary_new2();
```

```
for(i=0;i<5;i++) rb_ary_push(result,INT2NUM(ids[i]));
```

```
xfree(ids);
```

```
rb_protect(rb_ary_push_wrap,(VALUE)&args,&exception);
```

```
if(exception) {
```

```
    xfree(ids);
```

```
    rb_jump_tag(exception);
```

```
}
```

Résumé

- Būdas pagreitinti Ruby aplikācijas
- Galimybė perpanaudoti esamas bibliotekas
- Reikalauja žinių

Klausimai?

Ačīū už dėmesį