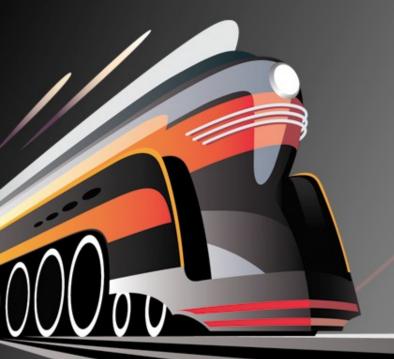
Lourens Naudé, Trade2Win

Inside Matz Ruby

High level Ruby internals crash course.



RailsWay Con

Bio

- http://github.com/methodmissing
 - Ruby contractor
 - Currently building out a consumer forex platform @ Trade2Win.
- creator
 - scrooge, mysqlplus_adapter, mri_instrumentation
- contributor
 - Rails, mysqlplus, query_memcached, thinking-sphinx, em-spec



RailsWayCon

```
def users_of(:ruby)
 frequently()
 raise Opinion
rescue
 next_slide.call()
end
```

Have you looked at the source?

Mere mortal, with a toolbox

- Basic C skills *
- Understanding of the UNIX subsystem *
- Dtrace *
- GDB *
- GCC macro expansion *
- Add preferred insomnia beverage
- * RTFM

The Symbol Table

- Generic hash implementation bins && items
- Inits with 11 bins, > 5 items per bin spawns more for optimal density
- Table.has_many :bins
- Bin.has_maby:symbol_table_entries
- Strings or numbers as hash keys
- Global vars and classes stored in number table
- Collisions do happen ... expensive



Values and identifiers

- VALUE and ID is an unsigned long
- Special types eg. true, false, nil directly represented by a VALUE ... immediates
- The infamous nil with ID 4: #define Qnil 4

```
struct RBasic {
    unsigned long flags;
    VALUE klass;
    };
```

Types

- One per builtin type: Array, Symbol etc.
- Special hidden types
 - Mixed in modules
 - Syntax tree nodes
 - Variable scope

Casting Values

- Immediates (nil) looked up first for efficiency
- ROBJECT(123343434) → T_ARRAY

Type Flags

- Singleton class ?
- Marked for GC ?
- Tainted ?
- Any finalizers defined ?
- Instance variables defined on the object ?
- Frozen ?

Classes

- Instance variables + method table
- Super member
- Only for Kernel, class->super == NULL
- Array->super == Object

Objects

- Instance var table
- Object#method
 - Searches the method table of it's class
 - Fallback to superclass on 404
- Method cache busted when
 - MyClass.send(:extend, OtherModule)

Method Inclusion

- class A include B end
- Included class: internal type for module mixins
- Pointer to the same ivar && method table

Hash

- Based on the symbol tables
 - Iteration, insertion && removal
- Iteration level
 - Track, and act upon, any changes during loops
- Mixed in modules

String

- Min. buffer size is 128 bytes
 - 's' occupies 128 sizeof('s')
- static const char null_str[] = "";
 - New strings point to null_str for efficiency
- Locking mechanism
 - String#gsub
 - IO.read && IO#sysread buffers

Array

- Default capacity of 16 elements
 - [x] occupies sizeof(16 of x) sizeof(x)
- Locking mechanism
 - Array#sort(!)

Preallocation

- Arrays and strings
 - Similar structure : length && ptr members
- An initial and expandable capacity
- Slightly more space allocated to avoid reallocation overheads
- Supports sharing of values, where possible

Blocks

- Snapshots
- Optional arguments + a body
- Persistent local scope
- Linkable
 - block->prev for iterators

Frames

- An item on the method call stack
- Arguments count and values
- Receiver state eg. AClass#method_invoked
- Links back to the previous frame
- Node state

Threads

- SIGVTALRM signal reserved
- Virtual ring through th->prev && th->next
- Evenly timesliced @ 10ms
- Threads can block for
 - A single file descriptor
 - A select operation
 - Thread.join && Thread#value

Garbage Collection

- Process.has_many :heaps
 - Where objects are stored
 - Heap.has_many :slots

- Heap space maxed
 - GC runs to free some space
 - Allocates a new heap * 1.8 bigger

GC - Entries

- Entries
 - Wraps core classes, nodes, scope etc.
- Reference values
 - Flags + pointer to the next value
- Freelist is special global reference value

GC - Allocation and deallocation

- Object deallocation
 - Immediates ignored
 - Class frees method table, File close it's descriptor etc.
- New object allocation
 - Force GC if no freelist
 - Reference value assigned from the freelist
 - Freelist updated to reflect the next free value

GC - Mark phase

- Object references it's klass, members etc.
 - Module marks it's methods and ivar table
- Ignores immediates
- Each object recursively marks others
- Sets the marked flag for reference values
- Applicable to
 - Global vars
 - Variables currently on the stack



GC - Sweep phase

- Iterates through all objects
- Deletes objects not previously marked
- Defers sweeping if finalizer registered

Questions?

Follow methodmissing @ github or twitter. Thanks!