Tweak your Ruby consoles

– 2011 –

```
loop{$><<">>";gets
$><<'=>';p eval$_}
```

J-_-L

rbjl.net

Tweak your Ruby consoles

Part 1:
Useful irb libraries / irbtools

Part 2:
ripl: The successor of irb?

Part 3:
Fancy Ruby shells

J-_-L

Tweak your Ruby consoles

Part 1:
Useful irb libraries / irbtools

J-_-L

Interactive Ruby Extensions

- “Just add somes helpers”-approach
  - irbtools, utility_belt, ...
- Heavily modded
  - pry, irt, ...
- Alternative irb implementations
  - ripl, dietrb, ir, pry
- Advanced shells
  - rush, fresh, ...

J-_-L

utility_belt

- By Giles Bowkett and others
- Adds a bunch of helper methods
  - “grab-bag of tricks, tools, […] including convenience methods, language patches, and useful extensions”
- Includes pretty specific helpers
  - Amazon S3
  - OS X
- Monolithic

J-_-L

irbtools

- “Meta gem”
  - Installs a set of useful gems
- One doesn't have to use/load all of them
- Supports different loading methods
  - autoload, threads, require
- Define code to be executed when lib is loaded
- Nice to use default settings via:
  - `require 'irbtools'`

J-_-L

wirb + fancy_irb

- **Wirb:** Result Syntax Highlighting
  - Support for generic and nested objects
  - Tries to correctly highlight standard Ruby objects
  - Tokenizer can be used without IRB
  - Currently monolithic – no plugin system
- **FancyIrb:** Colors everywhere!
  - Colorizes errors, prompts and `stdout` + `stderr`
  - “Hash Rocket” output

J-_-L

hirb + boson

- **Hirb:** Mini View Framework
  - Apply views based on an object's class
  - Provides various table and menu view helpers
    - Activated by default: ActiveRecord tables
  - Also provides pager for long output
- **Boson:** Command Framework
  - Like `rake` and `thor`, but:
  - Commands can be run from system shell or irb!
  - Use any library with boson
  - Supports repositories

J-_-L

You don't want to miss them...

interactive_editor

every_day_irb

clipboard

methodfinder

ori

zucker

J-_-L

You don't want to miss them...

ray

.vim

rq

cat

rrq

ls

copy

mf

reset

paste

.ri

.m

OS

RubyEngine

J-_-L

irbtools – future

- Add more gems that make one's life easier!
- Stay flexible – one should be always free to only use the libraries she/he wants
- More usage of hirb
  - More tables, more menus, and maybe... colorize them?
- Integrate some libraries into boson?

J-_-L

Tweak your Ruby consoles

Part 2:
`ripl`: The successor of `irb`?

J-_-L

IRB is getting old...

- 5000+ lines of (old!) code
  - That do... what?
- Hard to extend...
  - ...and to find the right spot to do so
- Default commands (e.g. sub-sessions) cannot be deactivated...
  - ...and the majority of `irb` users does not use them

J-_-L

but you only need...

```
loop{$><<">>";gets
$><<'=>';p eval$_}
```

J-_-L

Rewriting IRB – Approaches

- **irb2** by Yehuda Katz
- **ir** by James Tucker
- **dietrb** by Eloy Durán
- **pry** by John Mair
- **ripl** by Gabriel Horner

J-_-L

ripl

- “ruby interactive print loop”
- Light: Less then 300 lines of code
- Highly customizable via plugins
  - Plugins can alter shell behaviour, …
  - commands, ...
  - and start-up actions (e.g. command line options)
- Like IRB, it uses ~/.irbrc and ~/.irb_history
- Core plugins for history, readline support, auto-completion

J-_-L

ripl: plugin system

- No monkey patching involved!
- Plugins can hook into almost every method
  - get_input, prompt, eval_input, print_result, …
- Plugins create Ripl sub-modules and overwrite the specific method
  - Plugins call super, to pass the chain to the next plugin
- The last “plugin” is the ripl core:
  - `API` module

J-_-L

A simple ripl plugin

```ruby
module Ripl::CustomErrors
  def print_eval_error(err)
    if handler = config[:custom_errors][err.class]
      handler.call(err)
    else
      super
    end
  end
end

Ripl::Shell.include Ripl::CustomErrors
Ripl.config[:custom_errors] = {}
```

J-_-L

ripl “base” plugins

- multi_line
  - Different implementations
  - “Smart” compact history
- auto_indent
  - Actually indenting the closing end properly!
- color_result
  - Colorizes results using wirb
  - Can easily be changed to awesome_print or coderay

J-_-L

ripl... plugins are the way to go!

- **play:** play back and record input into ripl
- **debug:** automatically debugs a failed eval
- **irb:** fakes some irb behaivour
- **commands:** adds irb-like commands
- **i18n:** localized ripl
- **color_error**
- **color_streams:** stdout + stderr
- **rocket:** hash rocket
- **profiles:** adds a --profile option
- `gem list -r ripl`

J-_-L

Interactive Ruby Application Shells

- **rails:** the well-known web-framework
- **rack:** the well-known webserver interface
- **sinatra:** the other well-known web-framework
- **padrino:** the insider's tip web-framework
- **ronin:** the exploit development framework
- **hijack:** the ruby shell to any ruby process

J-_-L

Three reasons to actually use `ripl` instead of `irb` in every-day work

- It's well structured (**easily extendable**!)
- Proper auto-indention (“`end`”)
- Smart multi-line history

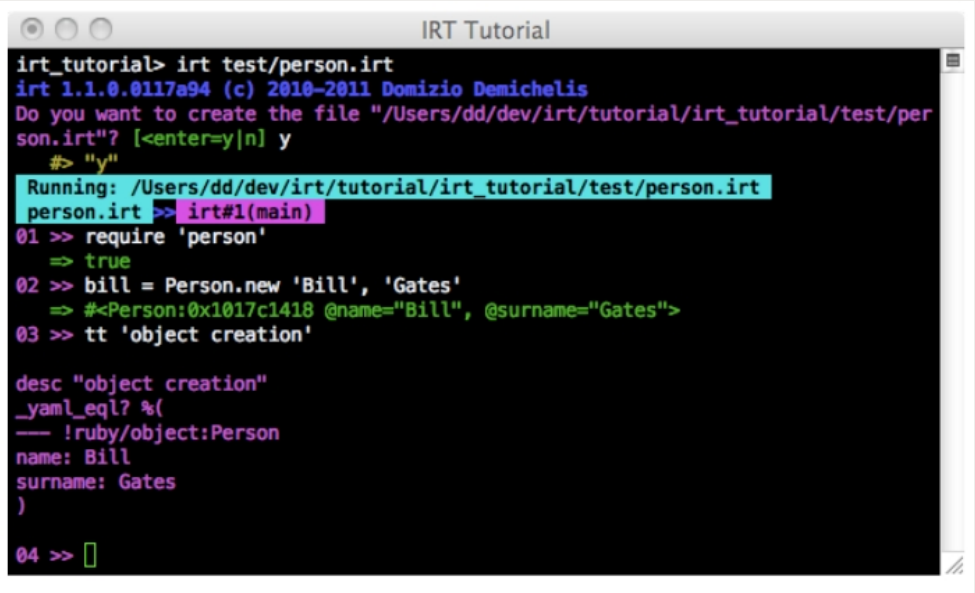
J-_-L

Tweak your Ruby consoles

Part 3:
Fancy Ruby shells

J-_-L

IRT



- Interactive Ruby Tools
  - “Improved irb and rails console with a lot of easy and powerful tools.”
- Concept of different sessions
  - interactive, inspecting, binding
- Record session steps and save them as tests
- Comes with a 23-pages tutorial pdf

J-_-L

- Monolithic
- Not a pure Ruby shell
  - Entered expressions get parsed before getting to ruby
- Goold live-help possibilities
  - Show / edit method definitions
- Run git, and rake from within pry

J-_-L

Ruby system shells

- **rush:** unix shell using ruby
  - `myproj['**/*.rb'].search(/^\s*class/).lines.size`
- **rubsh:** unix shell using ruby
  - `'*'.ls`
- **urchin:** unix-like commands
  - `mv image?.{png,gif} images`
- **rubish:** unix shell using ruby
  - `ls :l, "awk.rb", "sed.rb"`

J-_-L

Ruby system shells

- **fresh:** hybrid system / irb shell
  - fresh detects if command is system command or ruby
  - Almost no syntax mixing
    - You either use your system shell or irb
    - Exception: Store system result in a ruby variable
  - Based on ripl

J-_-L

Links

- J-_-L

  rbjl.net

  @rbjl

  https://github.com/janlelis
- **irbtools:** https://github.com/janlelis/irbtools
- **ripl:** https://github.com/cldwalker/ripl
- **ripltools:** https://github.com/janlelis/ripltools
- **pry:** https://github.com/banister/pry
- **fresh:** https://github.com/banister/pry
- Impress Template: hagetaka0

J-_-L