

JRuby in Action

Ruby, Rails, GUIs, and More



Except where otherwise noted, the content of this presentation is licensed under the Creative Commons Attribution-Share Alike 3.0 United States License (<http://creativecommons.org/licenses/by-sa/3.0/us/>).

The JRuby Guys

- Charles Oliver Nutter and Thomas Enebo
- JRuby for 4+ years
- Sun Microsystems for 2+ years
- JVM languages

Ruby

- Dynamically typed
- Pure object-oriented
- C-language implementation
 - Green threads
 - “Fast enough” == “a bit slow”
- 1.8.6/7 is current
- 1.9 in development
 - Focus on performance, character encodings

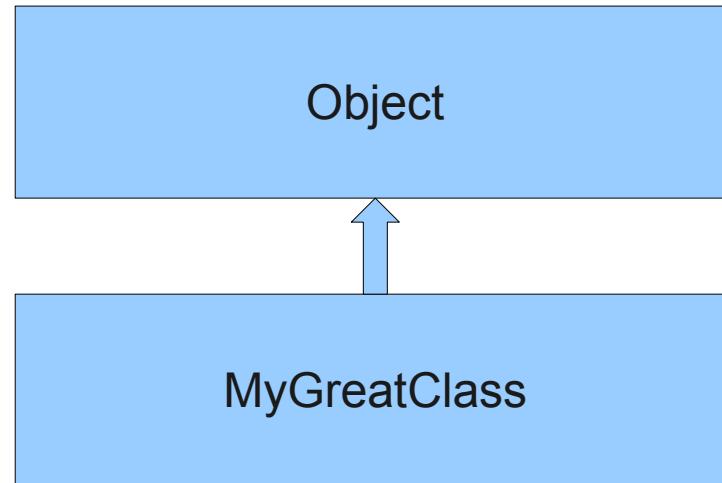
JRuby

- Ruby on the JVM
- Nearly complete Ruby compatibility
- Interop with Java libraries
- Improvements on original
 - Native threading
 - Faster by most measurements
- 1.1.6 this week (RC1 out now)
- 1.8.6 compatible, 1.9 in progress

Ruby Tutorial

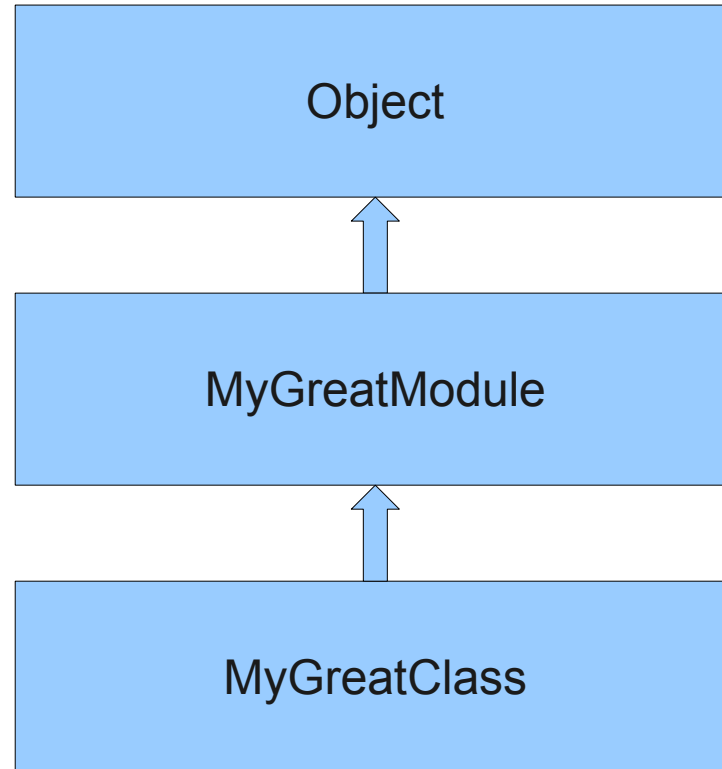
- Walkthrough of all the key Ruby features
- Interactive! Please jump in with questions
- Free-form, live-coded
- Ruby is fun, Ruby tutorial should be fun!

Modules



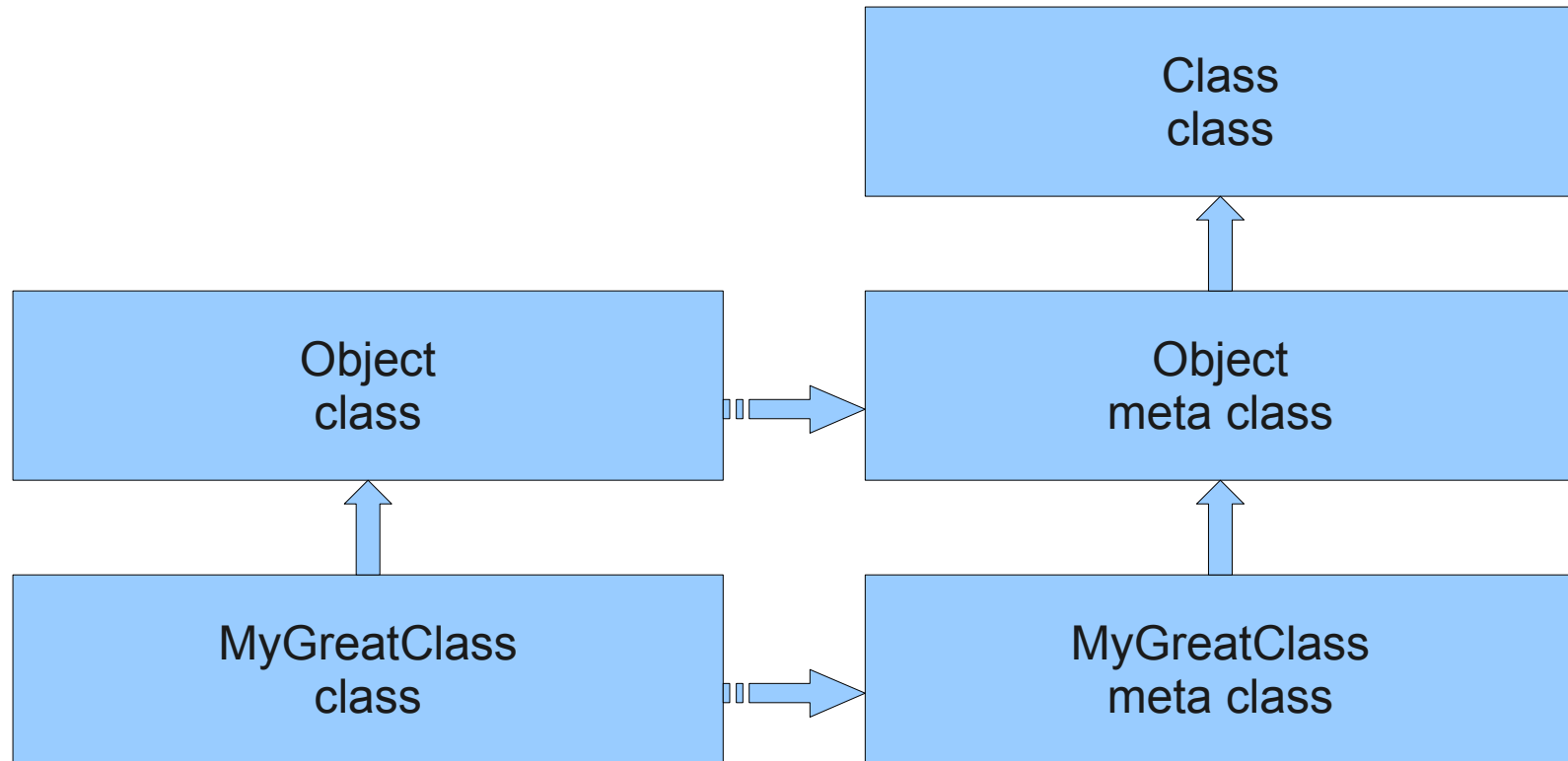
```
class MyGreatClass < Object
...
end
```

Modules



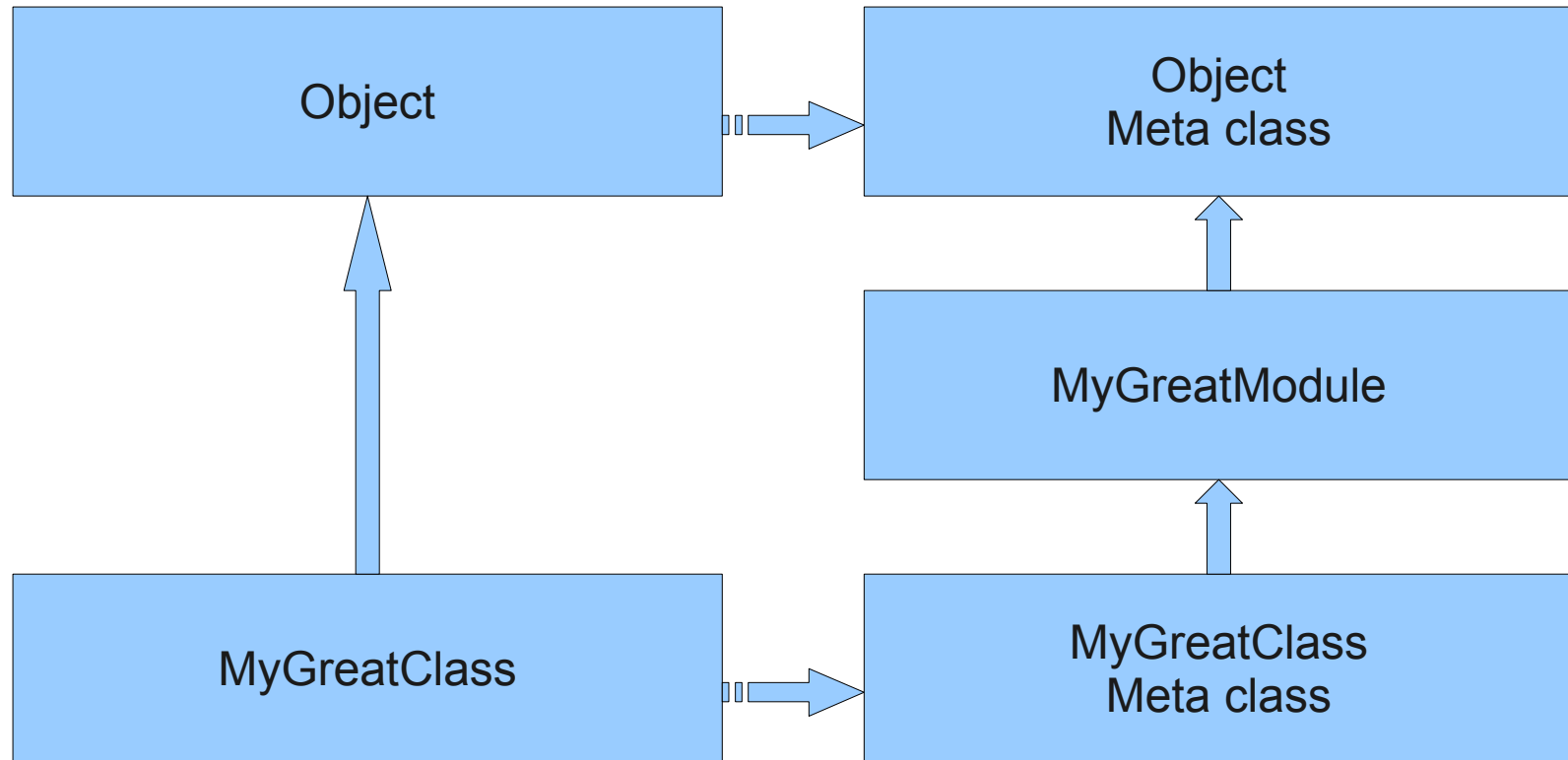
```
class MyGreatClass < Object  
  include MyGreatModule  
end
```

Modules



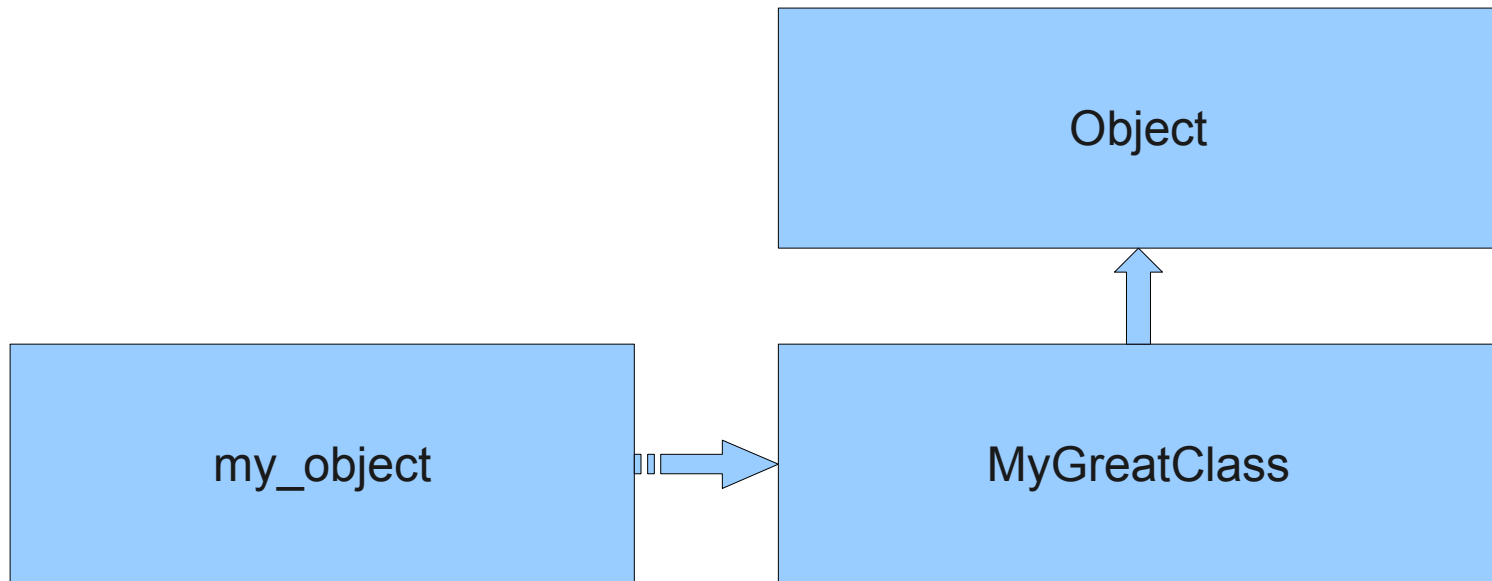
```
class MyGreatClass < Object
  extend MyGreatModule
end
```

Modules



```
class MyGreatClass < Object  
  extend MyGreatModule  
end
```

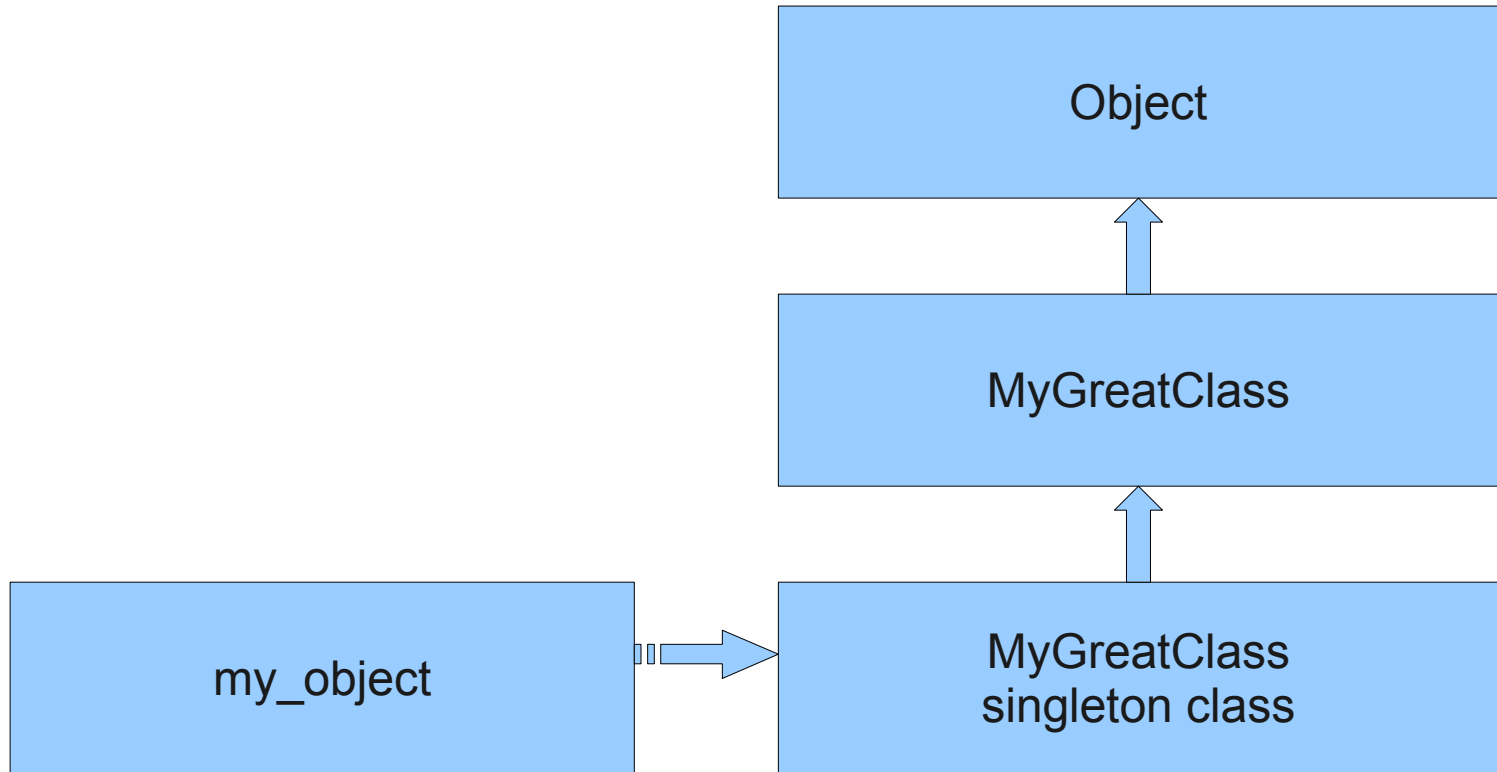
Singletons



```
class MyGreatClass
end
```

```
my_object = MyGreatClass.new
```

Singletons



```
class MyGreatClass  
end
```

```
my_object = MyGreatClass.new  
class << my_object  
end
```

More Cool Ruby: FFI

- Foreign Function Interface
- Call C libraries from Ruby code
- Works on Ruby and JRuby
- Map simple types, structs, callbacks
- DSL for function binding

JVMScript (working title)

- A DSL for generating JVM bytecode
- Multiple Ruby techniques employed
- Goal: structured assembly for JVM

What is Ruby on Rails?

- A Full-stack MVC web development framework
- Open Source (MIT), Many Contributors
- Written in Ruby
- Single-Threaded design (2.2 has MT-mode)
- First released in 2004 by David Heinemeier Hansson
- Super-hyped! :) :(

Rails Precepts

- Convention over Configuration
 - Why punish the common cases?
 - Encourages standard practices
 - Everything simpler and smaller
- Don't Repeat Yourself (DRY)
 - Framework written around minimizing repetition
 - Repetitive code harmful to adaptability
- Agile Development Environment
 - No recompile, deploy, restart cycles
 - Simple tools to generate code quickly
 - Testing built into framework

Why Rails?

- Greatly simplified web development
 - “Less Rails code than Java app configuration”
 - Instant applications: working code in minutes
- Makes small apps trivial to create
- Ruby is an excellent language
- Still growing in popularity in leaps and bounds

The Rails Stack

- ActiveRecord
 - ORM on steroids
- ActionPack
 - View and controller support
- ActiveResource
 - Restful resources
- Actionmailer
 - Email
- Activesupport
 - Unicode, json, miscellaneous helpers

Rails Demo

Let's write a simple blog app

JRuby on Rails

- Java is Everywhere
 - Every OS/Hardware platform you can think of
 - Probably already on a server near you
- Less political resistance
 - “JRuby is just another Jar file”
 - No need to install additional software on your servers
- Wider database support
- Decent performance

JRuby on Rails (Warbler)

- Java-style deployment to Application Server
- Bundles a Rails application into a WAR file
- Hand WAR file to production staff
- Rails App deployed!



http://wiki.jruby.org/wiki/JRuby_Rack

JRuby on Rails (GlassFish v3 gem)

- Entire App Server in a 4Mb gem
- Ruby-style deployment to GF Application Server
 - `gem install glassfish`
 - `glassfish <Rails app dir>`
- Configurable number of listeners all in one process
- Under active development



Parts is Parts

- Script Java technology you need into your Rails app
 - Call “legacy” Java from existing project
 - Get additional choices

Swing GUI in JRuby

- Builder approach (ex: Rubeus)
 - Sort of a “model 1” 1998 JSP approach
 - Great for simple forms
 - Difficult for long-term, large apps
- Tool-driven approach (ex: MonkeyBars)
 - Tool manages UI layout
 - Logic is MVC-driven
 - UI and logic evolved, maintained separately
 - Easier to scale to large apps