DRb

Distributed Ruby

What is DRb

- Distributed objects like Cobra and Java's RMI
- No IDL or ceremony to use
- Magic happening under the covers

Simple server

```
require 'drb/drb'
$SAFE = 1 # disable eval and friends
SERVER_URI = "druby://127.0.0.1:8000"

# simple class
class TimeServer
  def get_current_time; Time.now; end
end

# bind to a port
DRb.start_service(SERVER_URI, TimeServer.new)
puts "listening on #{SERVER_URI}"
DRb.thread.join
```

Client

```
require 'drb/drb'
require 'pp'

SERVER_URI = "druby://127.0.0.1:8000"

timeserver = DRbObject.new_with_uri(SERVER_URI)
pp timeserver #<DRb::DRbObject @uri= ..>
puts timeserver.get_current_time
```

What we saw

- service discovery is done through urls
- PORO - Server is a Plain old Ruby class
- A Stub is created
- Data is marshalled across
- It is a sync RPC call. not async by default

What Next

- We were returning a standard ruby class. What about a custom class ?

- Let us try returning a object - what can go wrong ?

Demo

Skipped topics

- Rinda
- ACL for DRb
- Ruby safe levels
- DRb over SSL
- GC and DRb

Usecase (not all of them are practical)

- For building RPC apis
- For building a distributed logging service. has the semantics of a method call
- A deployment bot. A daemon which is running on all nodes of a cluster listening for commands
- A background job worker
- map-reduce using Rinda. eg. starfish