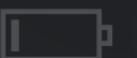


Let's Chat

Ruby and Elixir. An exploration of cross language communication

4

Kwantime



Hey,
I'm Garrett

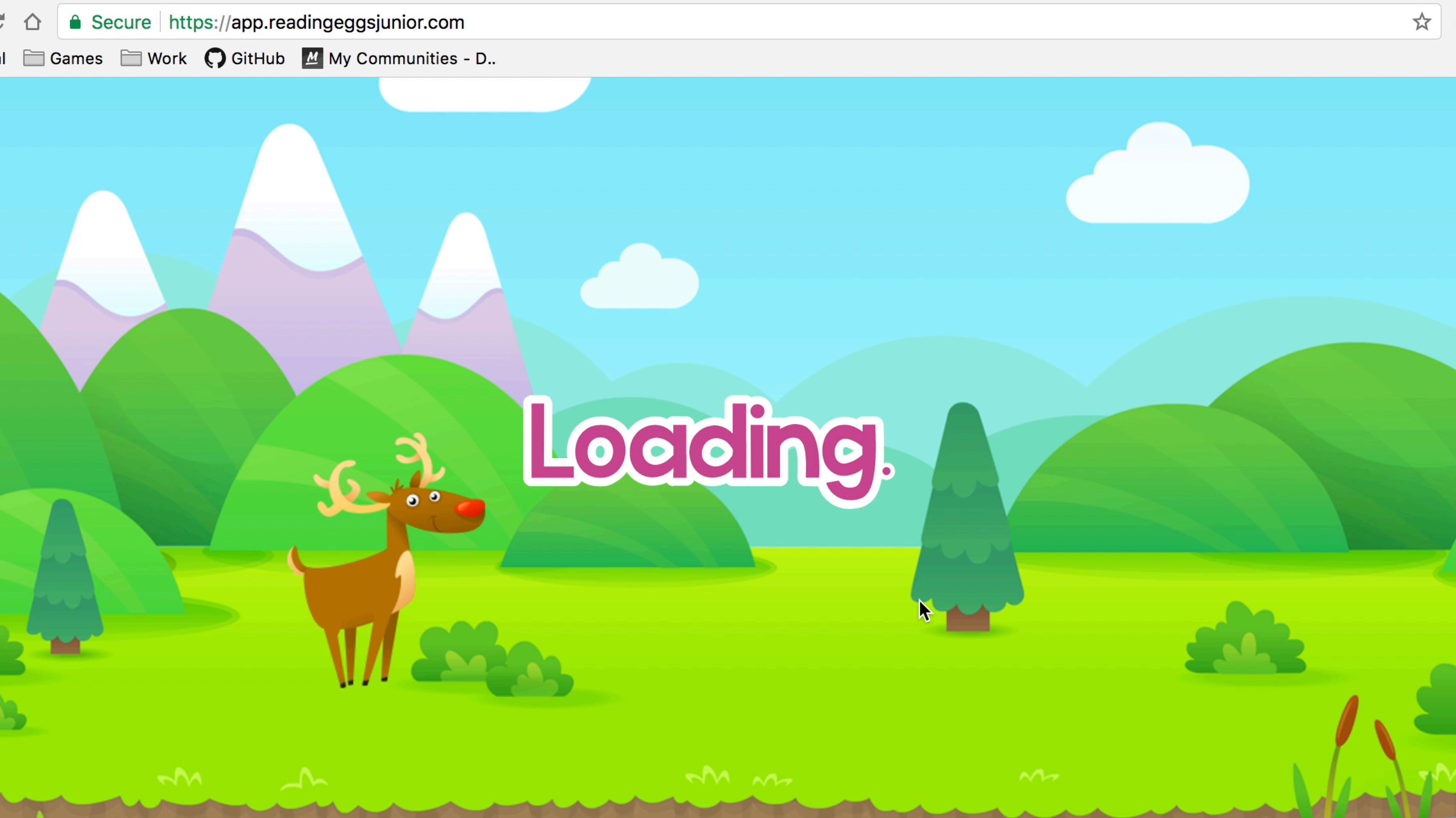
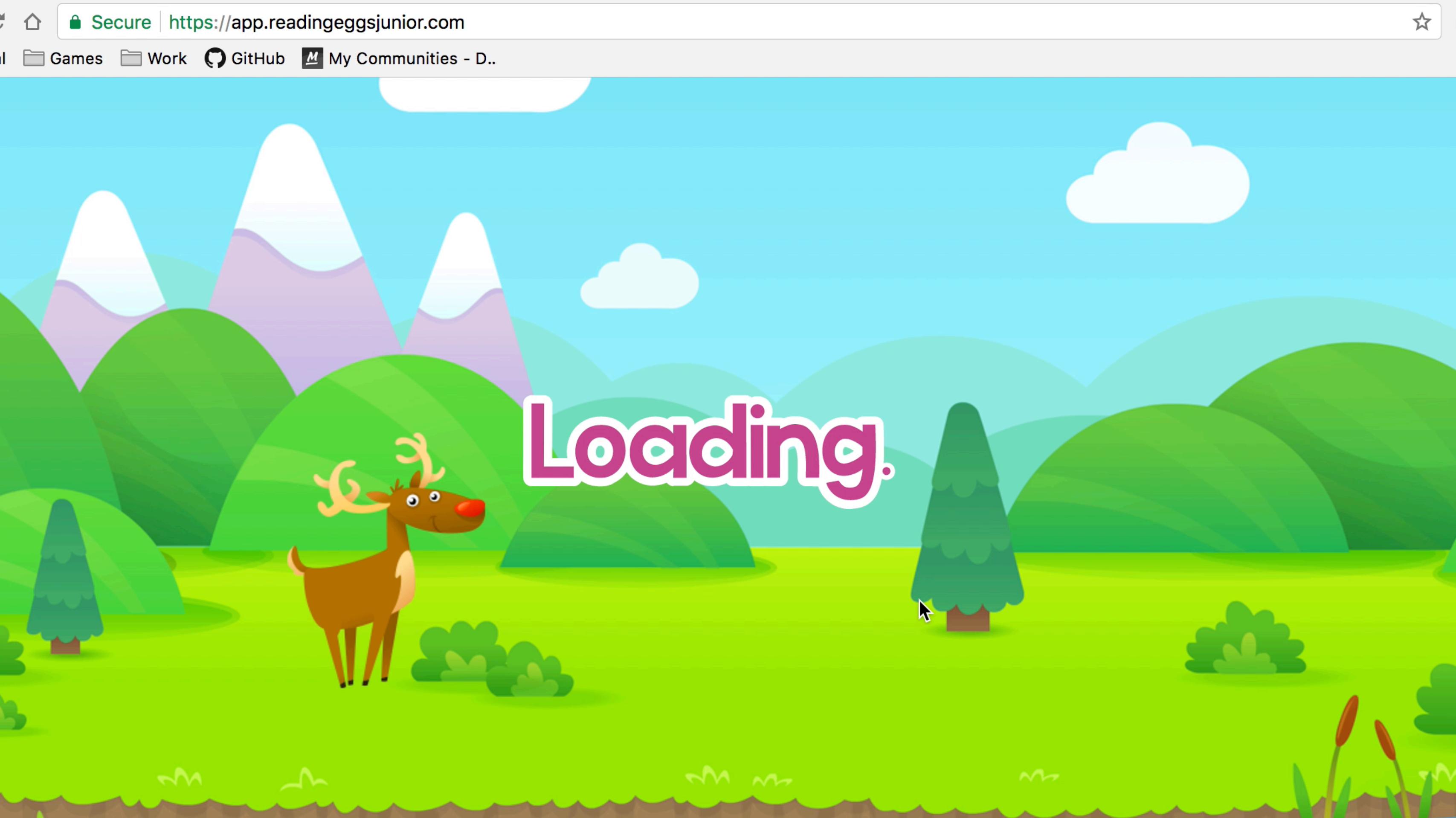


I WORK at Blake



We teach children





We use

Ruby & Elixir

Why did we move?

- *really* fast
- immutable data
- parallel / concurrent
- easy to isolate complexity into *applications* (microservices w/o http)
- encourages better design
- shiny
- does your taxes
- #1 hacker news

Agenda

- Refresher on **Ruby**
- Explain **Redis**
- Explore **Sidekiq**
- Introduce **Elixir**
- *Demo*
- Recap

Refresher on Ruby



Ruby - Hash

We use **keys** to unlock the **values**

Ruby - Hash



Ruby - Hash



my_house

- door_1 🛋 shower
- door_2 🛋 living_room
- door_3 🛋 bedroom

Ruby - Hash

```
my_house = {  
    door_1: "shower",  
    door_2: "living_room",  
    door_3: "bedroom",  
}
```

Ruby - Hash

```
person = {  
  name: "Garrett",  
  age: 26,  
  hearthstone_rank: 3,  
  fav_things: []  
}  
  
person[:hearthstone_rank]  
# => 3  
  
person[:fav_things] << "gummy bears"  
# => ["gummy bears"]
```

Ruby - Array

An Array is a list of items in order

Ruby - Array



Ruby - Array

0 1 2 3 4 5 6 7 8 9

dog	cat	snake	bever	hawk	mouse	deer	fly	moose	goose
-----	-----	-------	-------	------	-------	------	-----	-------	-------

Ruby - Array

```
a = []          # => []
a << "abc"      # => ["abc"]
a              # => ["abc"]
a.pop          # => "abc"
a              # => []
```

Ruby - Array

```
a = []          # => []
a.push("abc")   # => ["abc"]
a              # => ["abc"]
a.pop          # => "abc"
a              # => []
```

Ruby - Array

Appending to a list

```
x << y
```

```
x.push(y)
```

Ruby - Array

Pulling items off a list

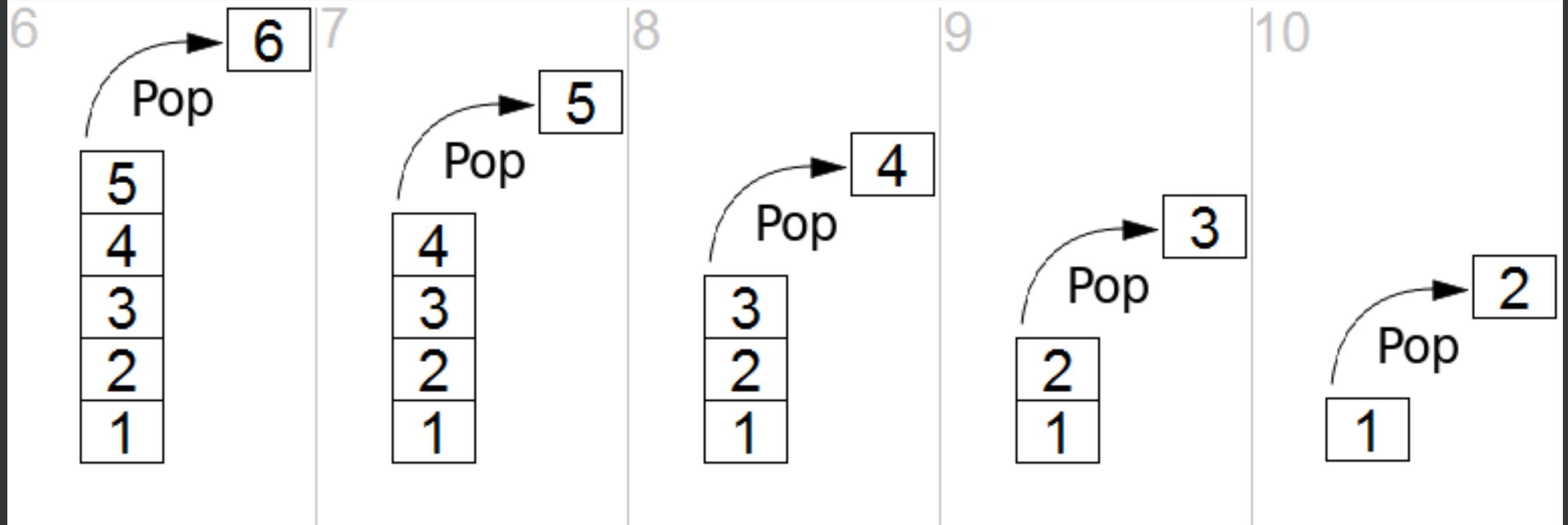
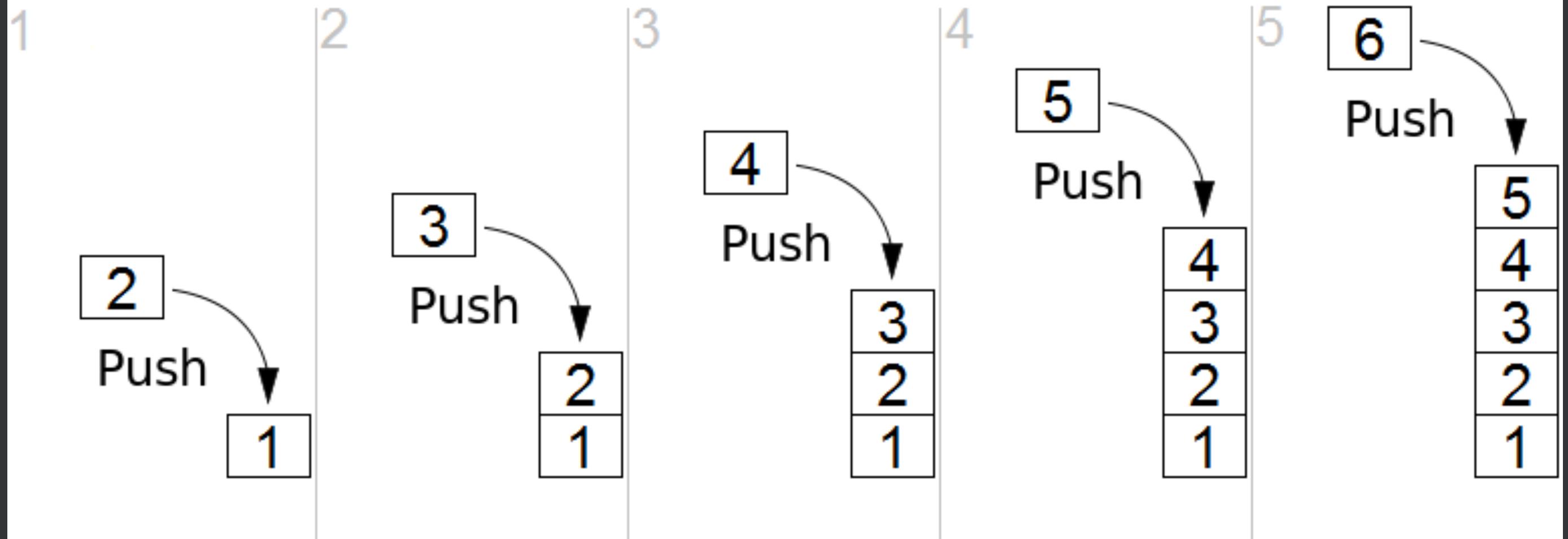
`x.pop(y)`

Ruby - Array

Push & Pop

`x.push(y)`

`x.pop(y)`



PushPop



**Well,
that's Ruby.**

Explain Redis



**Redis is a hash
in the cloud.**

```
$redis = {}

# setter
$redis[:thought] = "Where's the Elixir already?"
```

```
# getter
$redis[:thought]
# => "Where's the Elixir already?"
```

```
# setter
$redis[:drinks] << "little creatures"
$redis[:drinks] << "150 lashes"
```

```
# getter
$redis[:drinks]
# => ["little creatures", "150 lashes"]
```

```
$ redis-cli
```

```
set thought "Where's the Elixir already?"
```

```
get thought
```

```
# => "Where's the Elixir already?"
```

```
rpush drinks "little creatures"
```

```
rpush drinks "150 lashes"
```

```
lrange drinks 0 -1
```

```
# => 1) "little creatures"
```

```
# => 2) "150 lashes"
```

```
lpop drinks
```

```
# => "little creatures"
```

Redis Desktop Manager

Redis Desktop Manager

truth teller::db1::drinks X

LIST: drinks Size: 4 TTL: -1 Rename Delete

row	value
1	little creatures
2	150 lashes
3	{"drink_name": "Little Creatures", "strength": "5.2%"}

Add row Delete row Reload Value

Page 1 of 1 Set Page

Value: View as: JSON

```
{  
  "drink_name": "Little Creatures",  
  "strength": "5.2%"  
}
```

Save

2017-09-11 13:25:44 : Connection: truth teller > [runCommand] LLEN drinks > response received .
2017-09-11 13:25:44 : Connection: truth teller > [runCommand] LRANGE drinks 0 3
2017-09-11 13:25:44 : Connection: truth teller > [runCommand] LRANGE drinks 0 3 -> response received : Array
2017-09-11 13:25:49 : Connection: truth teller > [runCommand] LRANGE drinks 2 2
2017-09-11 13:25:49 : Connection: truth teller > [runCommand] LRANGE drinks 2 2 -> response received : Array
2017-09-11 13:25:49 : Connection: truth teller > [runCommand] LSET drinks 2 ---VALUE_REMOVED_BY_RDM---
2017-09-11 13:25:49 : Connection: truth teller > [runCommand] LSET drinks 2 ---VALUE_REMOVED_BY_RDM--- -> response received : +OK

2017-09-11 13:25:49 : Connection: truth teller > [runCommand] LREM drinks 0 ---VALUE_REMOVED_BY_RDM---
2017-09-11 13:25:49 : Connection: truth teller > [runCommand] LREM drinks 0 ---VALUE_REMOVED_BY_RDM--- -> response received :

Import / Export Connect to Redis Server System log

Explore Sidekiq

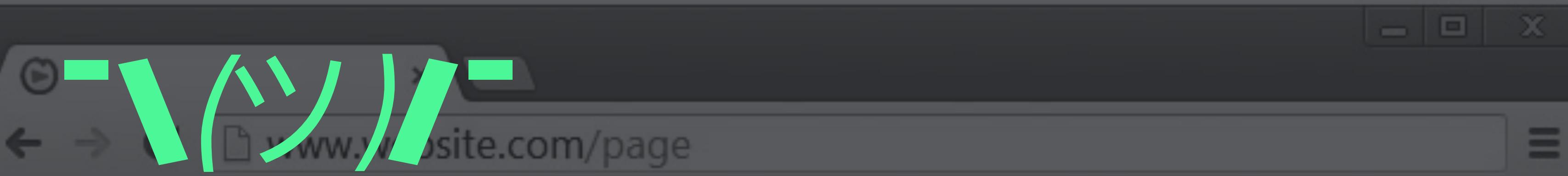
A black smartphone is shown from a top-down perspective, resting on a dark-colored computer keyboard. The phone's back panel features a prominent white 'Sidekiq' logo. The keyboard in front of the phone has visible keys for letters, numbers, and symbols. The background is a solid, light color.

Sidekiq

framework for background job processing

Why do I need a Sidekiq





408 Request Time-out

This request takes too long to process, it is timed out by the server. If it should not be timed out, please contact administrator of this web site to increase 'Connection Timeout'.

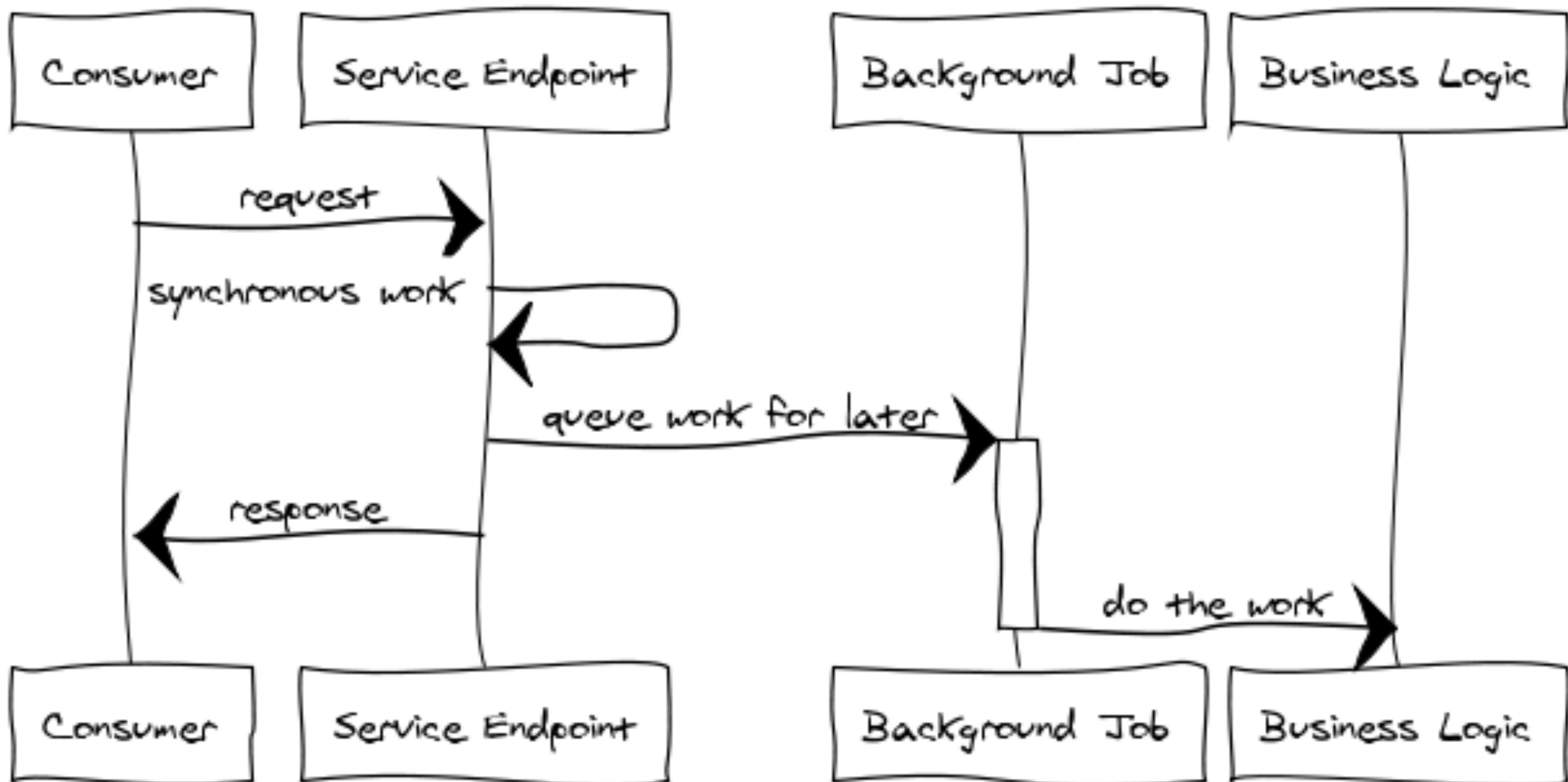
Powered By

[REDACTED] is not responsible for administration and contents of this web site!

Common Use cases

- Emails
- Expensive computation
- Imagine Processing
- Reports
- Etc..

Background Job



Sidekiq is built on 3 parts

- Client
 - push jobs into redis
- Redis
 - store the jobs
- Server
 - process the jobs

Client

```
MyWorker.perform_async({abc: "123", simple_as: "drm"})
```

Redis

```
xpush queue:default
 "{\"class\":\"MyWorker\", \"args\":[{\\"abc\\":\\"123\\", \\"simple_as\\":\\"drm\\"},\n  \\"retry\\":true, \\"queue\\":\"default\", \\"jid\\\":\"f03cd72fef59bfbbf4098890\", \n  \\"created_at\\":1503728661.8513222, \\"enqueued_at\\":1503728661.851379}"]
```

Server

```
class MyWorker
  include Sidekiq::Worker

  def perform(hash)
    puts hash.inspect # => {abc: "123", simple_as: "drm"}
  end
end
```

Client

```
class SomethingSingularOrPluarl < BigBaseController
  def create
    MyWorker.perform_async(superfriend_params)
    head :created
  end

  private
  def superfriend_params
    params.permit! # {abc: "123", simple_as: "drm"}
  end
end
```

Redis

Redis Desktop Manager

truth teller::d....::queue:default X

LIST: queue:default Size: 2 TTL: -1

Rename Delete Add row Delete row Reload Value

Page 1 of 1 Set Page

Value: View as: JSON

Save

{
 "class": "MyWorker",
 "args": [
 {
 "abc": "123",
 "simple_as": "drm"
 }
],
 "retry": true,
 "queue": "default",
 "jid": "dfe1d657c841e97ef71edcbc",
 "created_at": 1505120203.7369971,
 "enqueued_at": 1505120203.737372
}

2017-09-11 18:56:49 : Connection: truth teller > [runCommand] LLEN queue:default -> response received :
2017-09-11 18:56:49 : Connection: truth teller > [runCommand] LRANGE queue:default 0 1
2017-09-11 18:56:49 : Connection: truth teller > [runCommand] LRANGE queue:default 0 1 -> response received : Array
2017-09-11 18:56:53 : Connection: truth teller > [runCommand] LRANGE queue:default 1 1
2017-09-11 18:56:53 : Connection: truth teller > [runCommand] LRANGE queue:default 1 1 -> response received : Array
2017-09-11 18:56:53 : Connection: truth teller > [runCommand] LSET queue:default 1 ---VALUE_REMOVED_BY_RDM---
2017-09-11 18:56:53 : Connection: truth teller > [runCommand] LSET queue:default 1 ---VALUE_REMOVED_BY_RDM--- -> response received : +OK

2017-09-11 18:56:53 : Connection: truth teller > [runCommand] LREM queue:default 0 ---VALUE_REMOVED_BY_RDM---
2017-09-11 18:56:53 : Connection: truth teller > [runCommand] LREM queue:default 0 ---VALUE_REMOVED_BY_RDM--- -> response received :

Import / Export Connect to Redis Server System log

Server

```
class MyWorker
  include Sidekiq::Worker

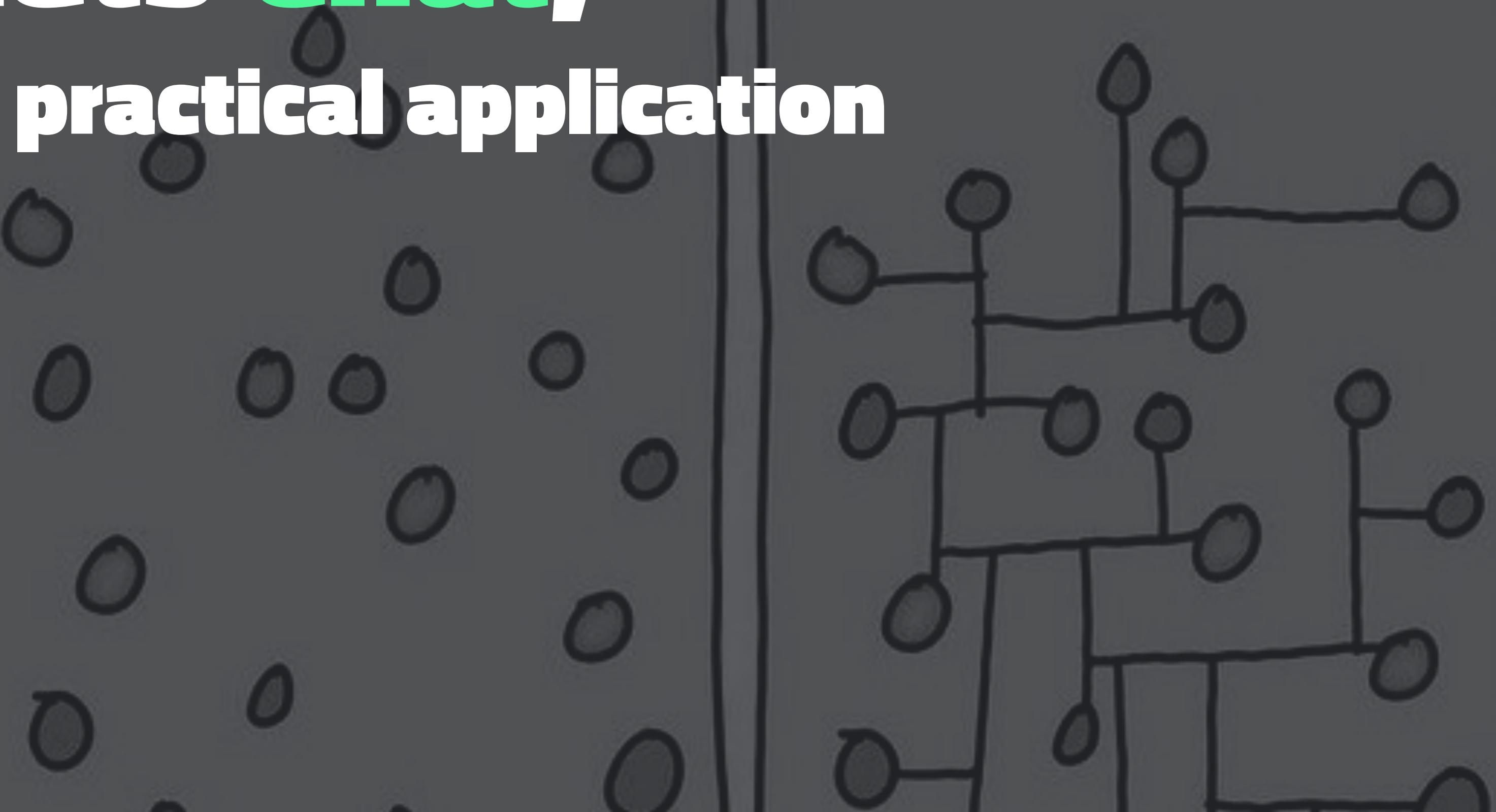
  def perform(hash)
    puts hash.inspect # => {abc: "123", simple_as: "drm"}
  end
end
```

Knowledge

Lets **Chat**,

A practical application

Experience

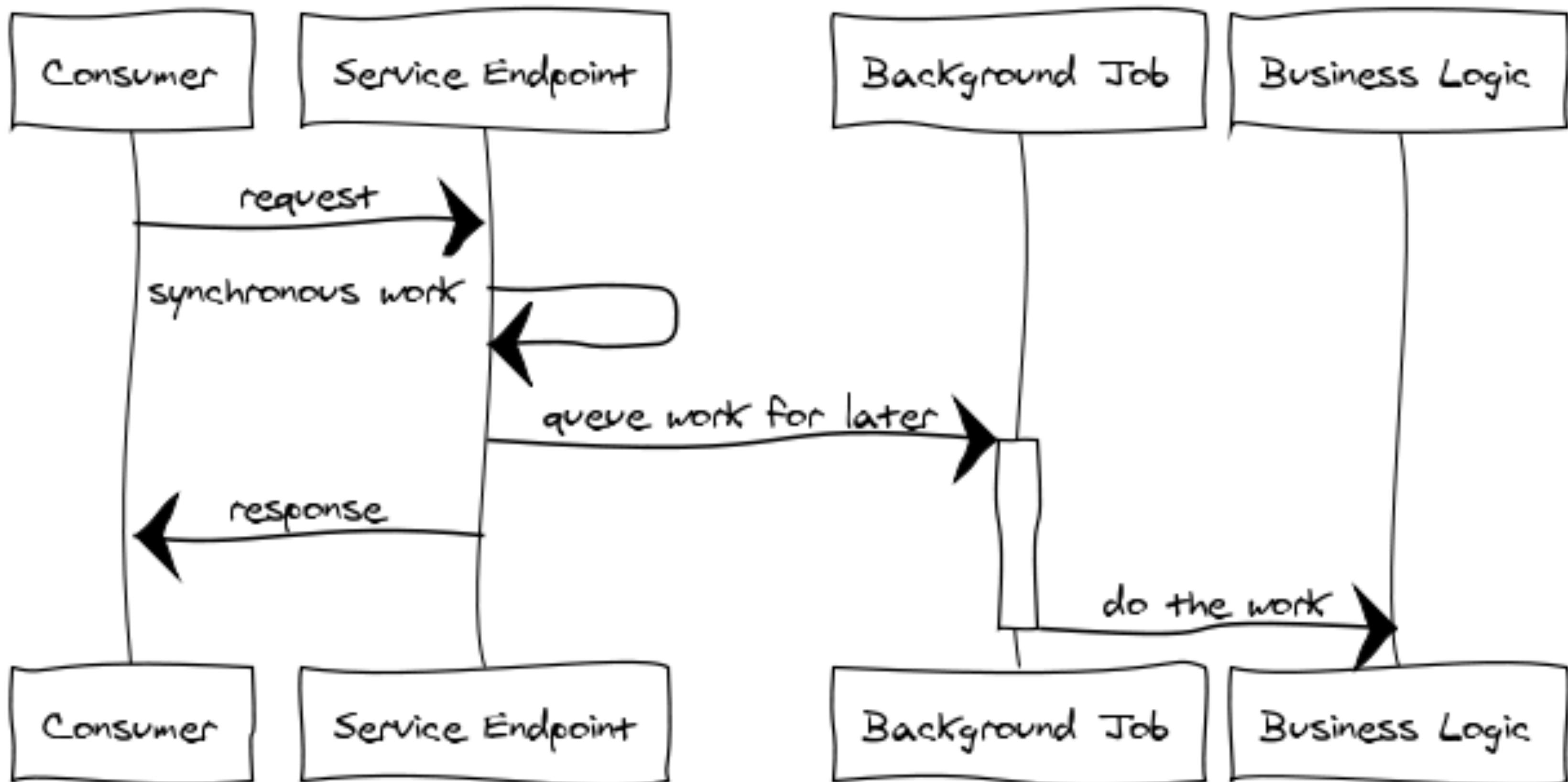


Sidekiq - how does it work?

```
# clients
rpush "redis:jobs" {assignee: "Audience Member", task: "Greet your neighbour"}
rpush "redis:jobs" {assignee: "Audience Member", task: "Take a drink"}
rpush "redis:jobs" {assignee: "Audience Member", task: "Heckle louder"}
rpush "redis:jobs" {assignee: "Audience Member", task: "Do Garrett's Taxes"}

# sidekiq
while true
  job = lpop "redis:jobs"
  if job
    AudienceMember.perform(job)
  end
end
```

Background Job



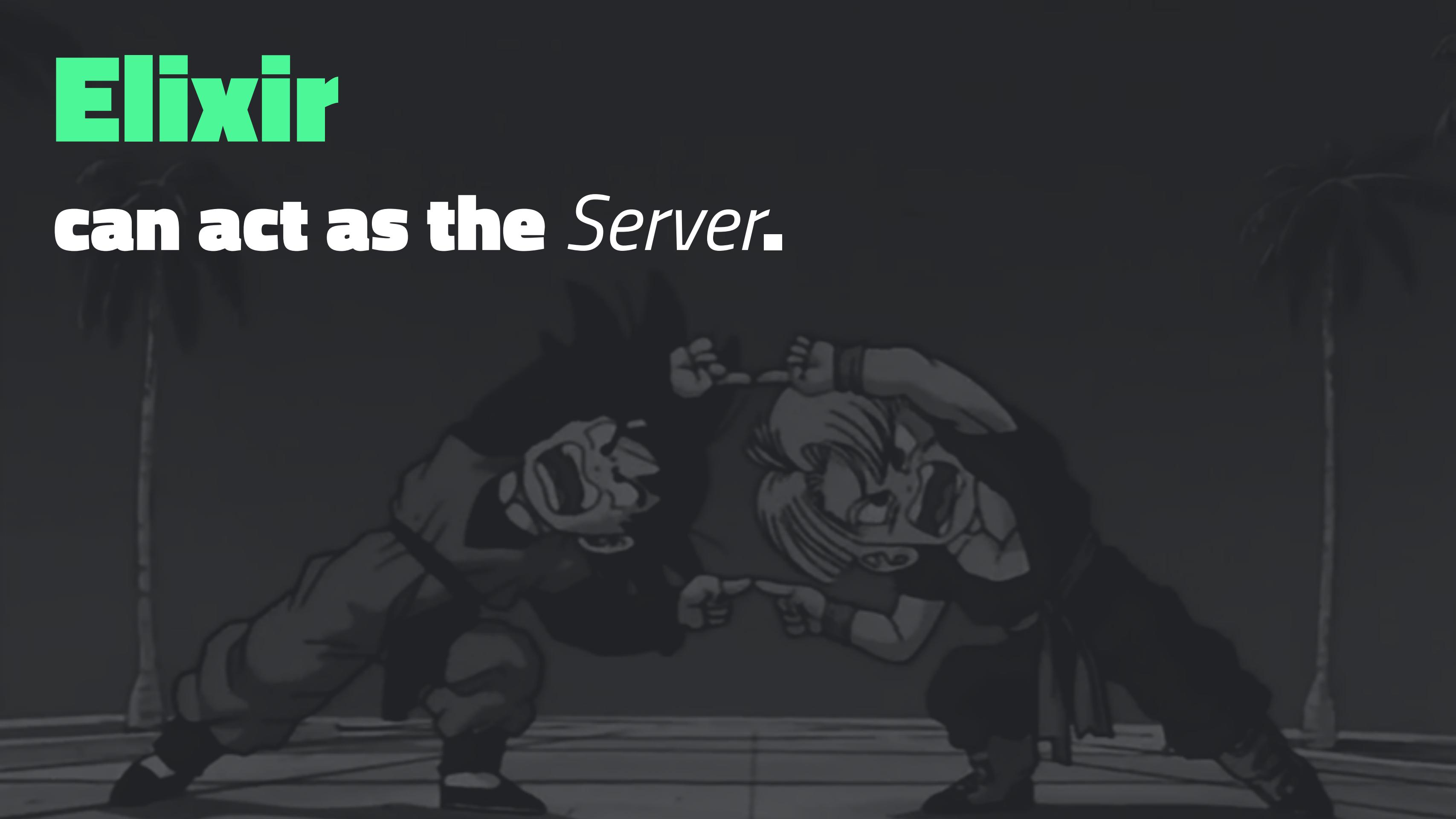
**Well,
thats Sidekiq.**

Introduce Elixir

A black and white photograph of a person from the waist up, wearing a dark, ribbed hoodie. The hoodie has the words "Introduce Elixir" printed across the chest in a large, bold, sans-serif font. The "Elixir" part is colored bright cyan. The background is a plain, light-colored wall.

Elixir

can act as the *Server*.





Exq

job processing library compatible
with **Sidekiq** for **Elixir**

Configure Exq

```
config :exq,  
  host: "localhost",  
  port: 6379,  
  namespace: "",  
  concurrency: 10,  
  queues: ["default"],  
  poll_timeout: 200
```

Ruby Sidekiq Server

```
class MyWorker
  include Sidekiq::Worker

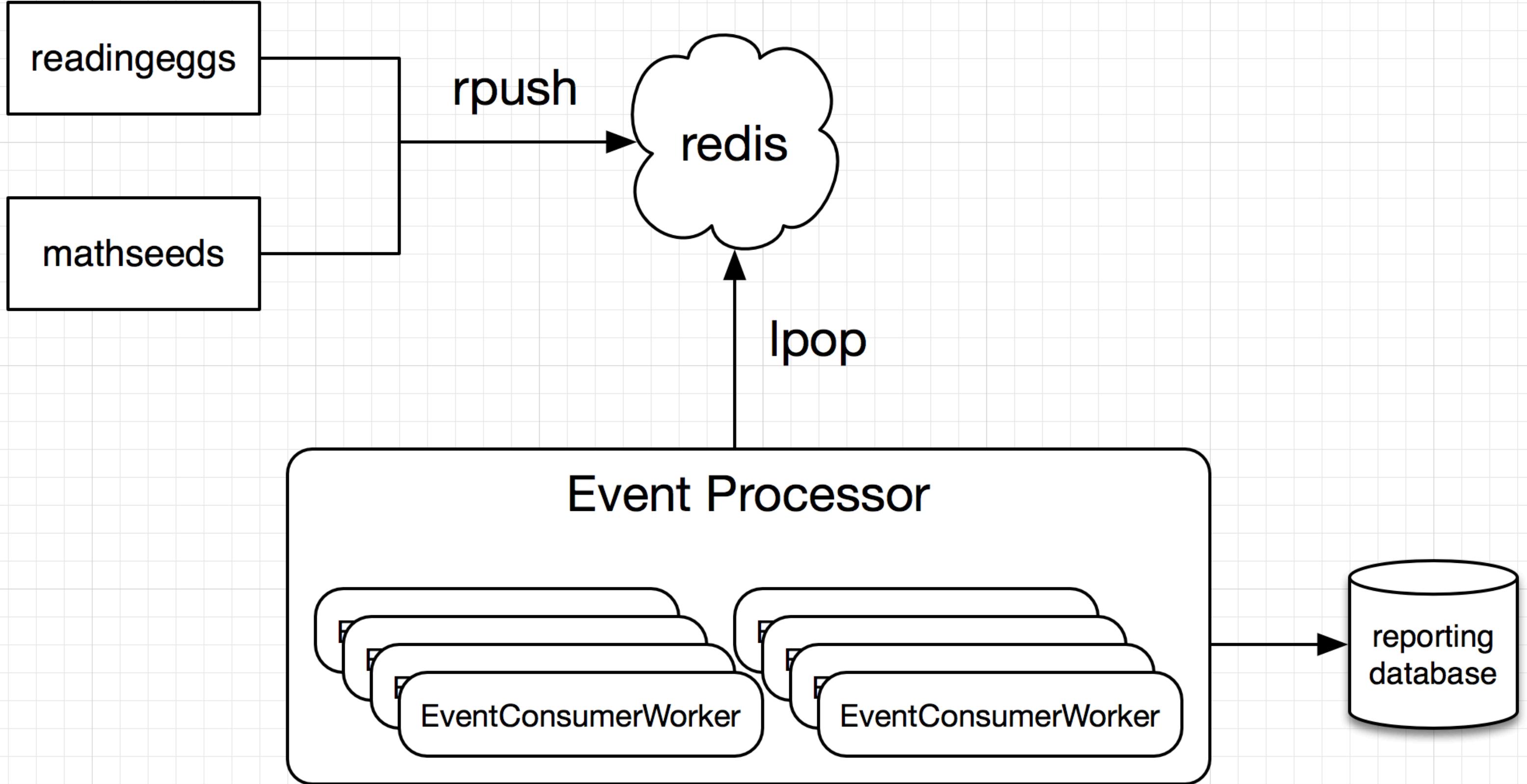
  def perform(hash)
    puts hash.inspect # => {abc: "123", simple_as: "drm"}
  end
end
```

Elixir Sidekiq Server

```
defmodule MyWorker do
  def perform(map) do
    IO.inspect(map) # => %{abc: "123", simple_as: "drm"}
  end
end
```

Demo Time





Recap

- Redis is a Hash
- We push jobs into a list
- We pull jobs off the list
- We can process the jobs with Ruby or Elixir
- Exq is the Elixir Sidekiq

Thanks

@gogogarrett

