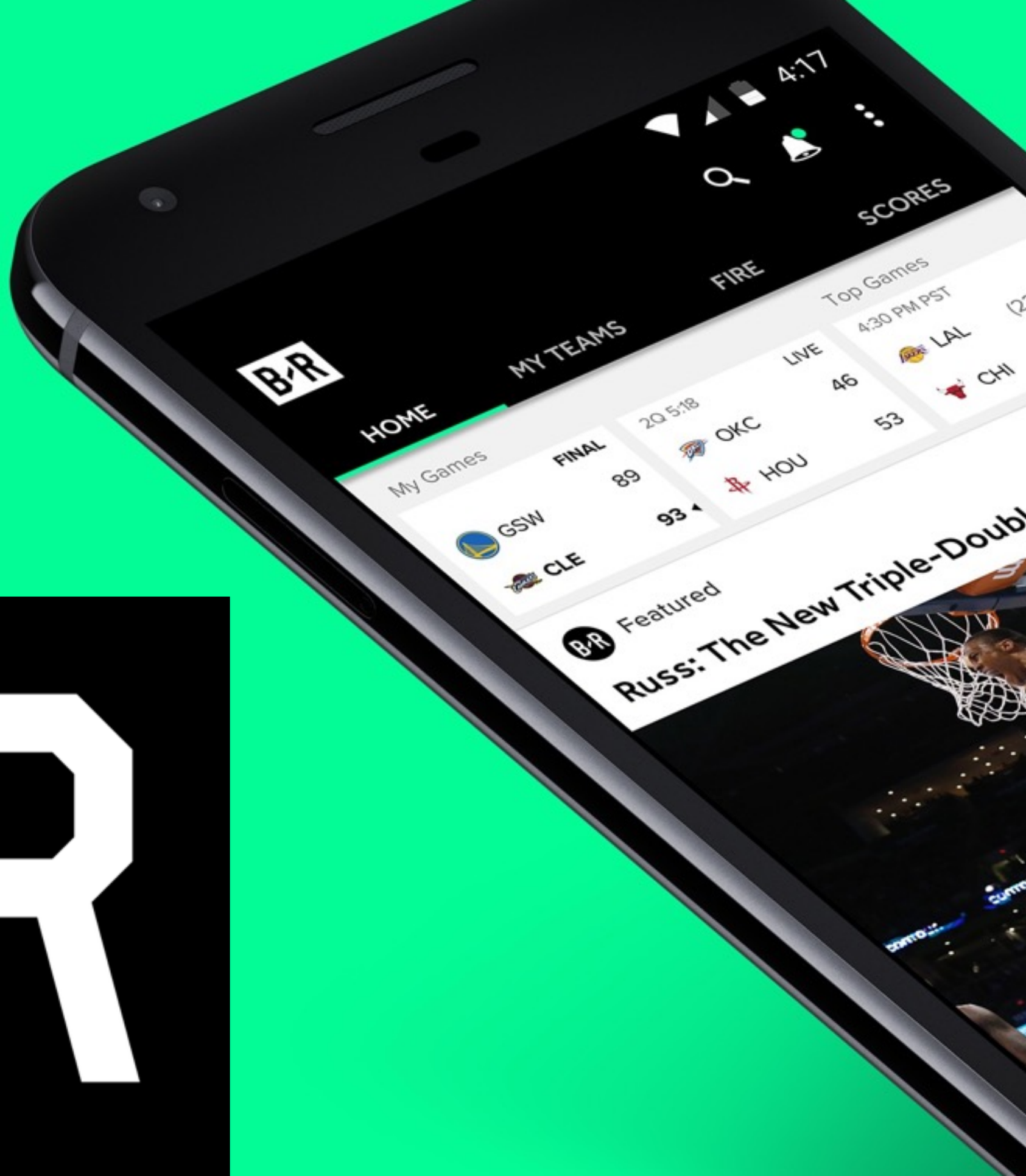
KEEPING CODE CONSISTENT

BEN MARX

@bgmarx

LEAD DEVELOPER

B-R



IN THE BEGINNING

OCTOBER 2014

DO WHAT FEELS RIGHT

PHOENIX README

MESSY ELIXIR CODE

```
def for(url) do
  url = prepend_http(url)

  {_, response } =
    url
    |> discover
    |> HTTPoison.get([], @redirect)
  case response do
    %HTTPoison.Response{status_code: 200, body:
body} ->
      decoded_body =
        Poison.decode!(body)
        |> strip_newline_chars

      cond do
          is_twitter?(url) ->
            format_twitter_response(decoded_body)
          is_instagram?(url) ->
```

COLLABORATIVE BARRIERS

CREDO

https://github.com/rrrene/credo

```
def for(url) do
  url
  |> format_url
  |> get_embed_data
  |> format_by_content_type
end
```

```
defp format_url(url) do
 # standardizes url based on certain requirements
end
```

```
defp get_embed_data(response) do
  # attempts to extract embed data from external source
end
```

```
defp format_by_content_type(:content_type, response) do
  # formats responses and handles errors
end
```

TYPESPECS & DIALYZER

```
@type embed :: map()

@spec for(String.t) :: embed
def for(url) do
  url
  |> parse_response
  |> format_by_content_type
end
```

PERPEXLING ERRORS

Function handle_cast/2 has no local return

The return type tuple() in the specification of init/1 is not a subtype of 'ignore' | {'ok',_} | {'stop',_} |{'ok',_,'hibernate' | 'infinity' | non_neg_integer()}, which is the expected return type for the callback of 'Elixir.GenServer' behaviour

EQUIVALENT

```
@spec insert_changeset(content,
%{}) :: Ecto.Changeset.t
```

```
@spec insert_changeset(content,
%{}) :: struct
```

Silly me. It turns out (unsurprisingly so) that I was wrong, and Dialyzer was right, all along. It would keep telling me my -spec was wrong, and I kept believing it wasn't. *I lost my fight, Dialyzer and my code won*. This is a good thing, I believe.

http://learnyousomeerlang.com/dialyzer

DOCUMENTATION

```
@doc """
Attempts to fetch and extract embed
data for a url and format it
according to the defined content type
"""
@spec for(String.t) :: embed
def for(url) do
  url
  |> get_embed_data
  |> format_by_content_type
end
```

```
@doc """
Attempts to fetch and extract embed data for a
url and format it according to the defined
content type
"""
@spec fetch_and_extract_embed(String.t) :: embed
def fetch_and_extract_embed(url) do
  url
  |> get_embed_data
  |> format_by_content_type
end
```

TESTING

HOW TO MEASURE COVERAGE?

EXCOVERALLS

https://github.com/parroty/excoveralls

```
$ MIX_ENV=test mix coveralls
...
----------------
COV    FILE                                        LINES RELEVANT   MISSED
100.0% lib/excoveralls/general.ex                     28        4        0
 75.0% lib/excoveralls.ex                             54        8        2
 94.7% lib/excoveralls/stats.ex                       70       19        1
100.0% lib/excoveralls/poster.ex                      16        3        0
 95.5% lib/excoveralls/local.ex                       79       22        1
100.0% lib/excoveralls/travis.ex                      23        3        0
100.0% lib/mix/tasks.ex                               44        8        0
100.0% lib/excoveralls/cover.ex                       32        5        0
[TOTAL]  94.4%
----------------
```

NEW CODE TESTED?

lib/excoveralls/html.ex

78.9 coverage 38 SLOC

```
0     defmodule ExCoveralls.Html do
1       @moduledoc """
2       Generate HTML report of result.
3       """
4
5       alias ExCoveralls.Html.View
6
7       @file_name "excoveralls.html"
8
9       defmodule Line do
10        @moduledoc """
11        Stores count information and source for a sigle line.
12        """
13
14        defstruct coverage: nil, source: ""
15      end
16
17      defmodule Source do
18        @moduledoc """
19        Stores count information for a file and all source lines.
20        """
21
22        defstruct filename: "", coverage: 0, sloc: 0, hits: 0, misses: 0, source: []
23      end
24
25      @doc """
26      Provides an entry point for the module.
27      """
28      def execute(stats, options \\ []) do
29  1     ExCoveralls.Local.print_summary(stats)
30
31  1     source(stats, options[:filter]) |> generate_report
32      end
33
34      @doc """
35      Format the source code as an HTML report.
36      """
37  2   def source(stats, _patterns = nil), do: source(stats)
38  1   def source(stats, _patterns = []),  do: source(stats)
39      def source(stats, patterns) do
```


overview
lib/ excoveralls.ex 97.5
lib/excoveralls/ circle.ex 93.3
lib/excoveralls/ conf_server.ex 88.9
lib/excoveralls/ cover.ex 50
lib/excoveralls/ exceptions.ex 0
lib/excoveralls/ html.ex 78.9
lib/excoveralls/html/ safe.ex 100
lib/excoveralls/html/ view.ex 100
lib/excoveralls/ local.ex 100
lib/excoveralls/ path_reader.ex 100
lib/excoveralls/ post.ex 80
lib/excoveralls/ poster.ex 85.7
lib/excoveralls/ settings.ex 100
lib/excoveralls/ stat_server.ex 100
lib/excoveralls/ stats.ex 100
lib/excoveralls/ stop_words.ex 100
lib/excoveralls/ sub_apps.ex 100
lib/excoveralls/task/ util.ex 100
lib/excoveralls/ travis.ex 88.9
lib/mix/ tasks.ex 92.5

STATIC ANALYSIS

TYPESPECS

DOCUMENTATION

TESTING

CODE REVIEWS

SERVICE REVIEW

DATABASE PERF

LOGGING

MONITORING

CI HOOKS

FOCUS ON THE CODE

HOW TO MEASURE SUCCESS?

ONGOING PROCESS

HABITUAL

&

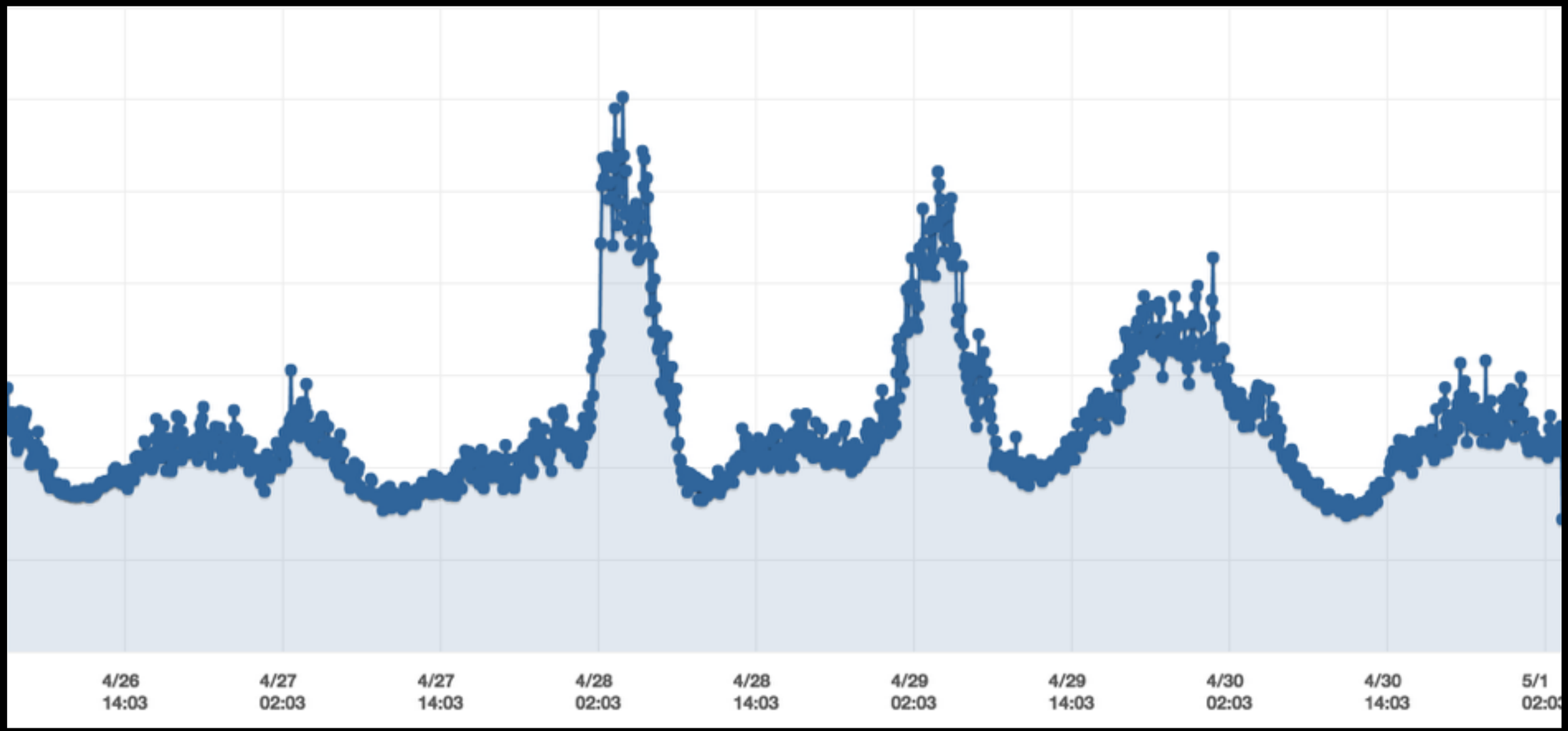
PREDICTABLE

NFL DRAFT

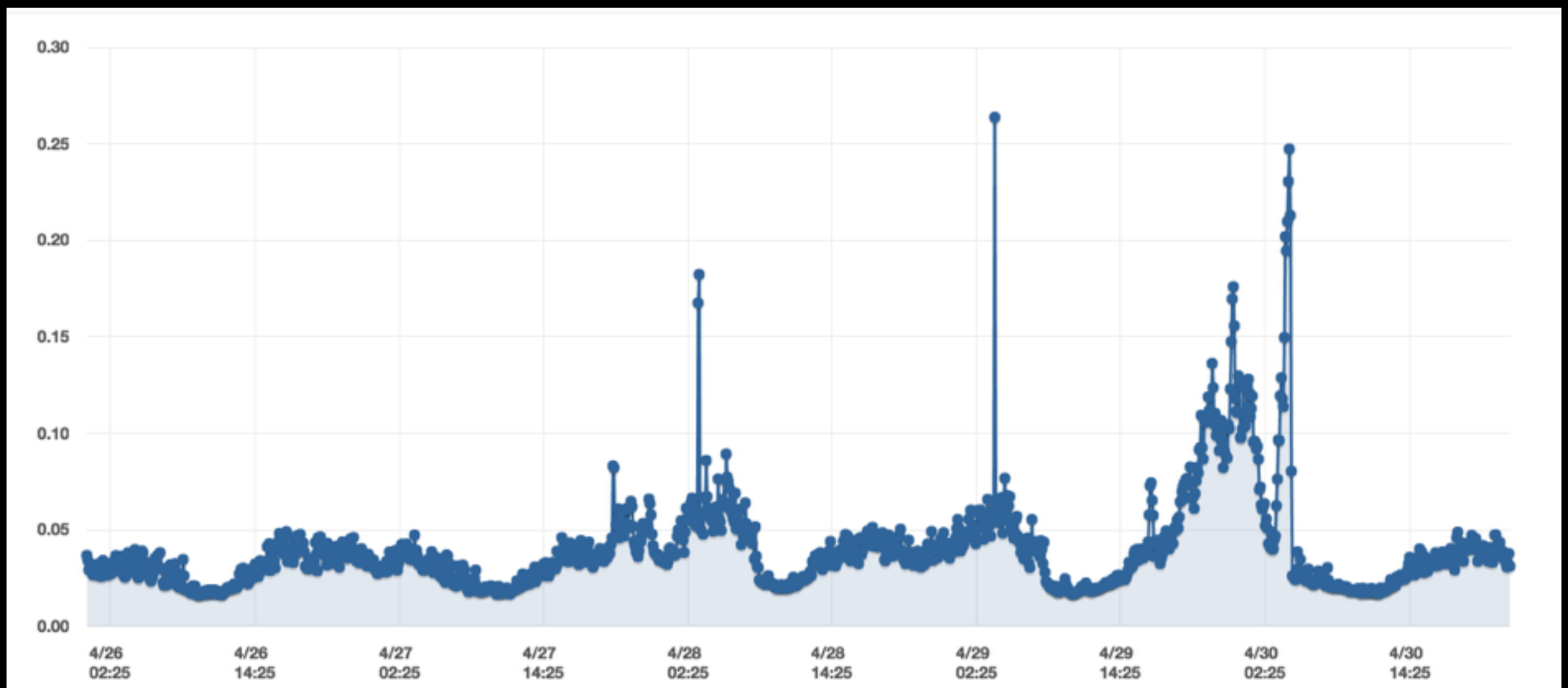
B/R STATE OF THE UNION

HIGHEST CONCURRENTS

SPIKES


4/26
14:03
4/27
02:03
4/27
14:03
4/28
02:03
4/28
14:03
4/29
02:03
4/29
14:03
4/30
02:03
4/30
14:03
5/1

LATENCIES


0.30
0.25
0.20
0.15
0.10
0.05
0.00
4/26 02:25
4/26 14:25
4/27 02:25
4/27 14:25
4/28 02:25
4/28 14:25
4/29 02:25
4/29 14:25
4/30 02:25
4/30 14:25

STRESS FREE

NUMBER OF CONTRIBUTORS

APPLICATION	NUMBER OF CONTRIBUTORS	PROJECT AGE
MONOLITH	59	BEGINNINNG OF TIME

APPLICATION	NUMBER OF CONTRIBUTORS	PROJECT AGE
FIRST ELIXIR APP POWERS CLIENTS	23	2.5 YEARS
OEMBED	12	2 YEARS
METADATA	10	8 MONTHS

SPEND TIME DEVELOPING

ADOPTING ELIXIR

FROM CONCEPT TO PRODUCTION

JOSÉ VALIM

BRUCE TATE

BEN MARX

QUESTIONS?

@bgmarx