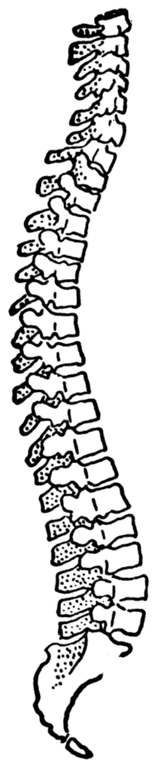
LET'S DO SPA

MAREK PIASECKI

WHAT IS SPA?

SINGLE PAGE APPLICATIONS



TAKE CARE OF YOUR BACKBONE

A COMPONENT

Knows how to display itself.

Knows how to behave.

Has configurable properties.

EVERYTHING* IS A COMPONENT

*that user sees and is interacting with

REACT **VS** WEB COMPONENTS

PROTIP #1

KEEP THE COMPONENT ABSTRACTION

KEEP THINGS TOGETHER

BASE ON STATE

DON'T MODIFY DOM BY HAND

MAKING COMPONENT ITERATIONS

1. Make component mockup with hardcoded state

2. Style component

3. Define interface actions

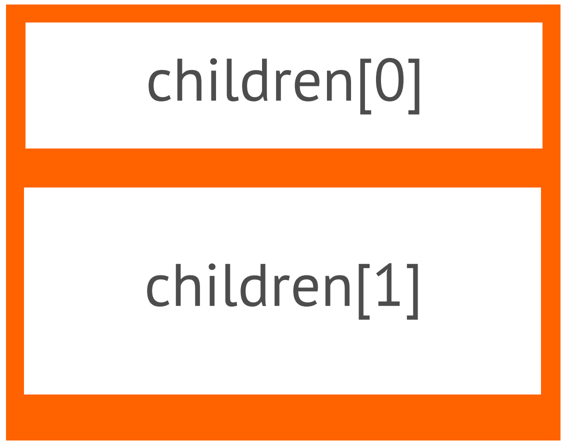
4. Provide endpoints
   - for data
   - for actions

READING AND WRITING DATA

IS SOMETHING VERY DIFFERENT

LAYOUT IS ALSO A COMPONENT

```
<HomeLayout>
  <TopBar />
  <HomeStream />
</HomeLayout>
```



COMPLETE STRUCTURE FOR FRONTEND

components/

pages/

services/

mixins/

app.coffee

GETTING DATA RAILS WAY

$$\begin{tikzcd}
\text{router} \arrow[r] & \text{controller} \arrow[r] \arrow[d, dash] & \text{serializer} \\
 & \text{DB} &
\end{tikzcd}$$

GETTING DATA

serializer

DB

GETTING DATA CONVENTIONS

match '*/*path*' => '*serializations/routing#router*', via: *:get*

/endpoint/:id
/endpoint/_/method

_ is a special sign

Serializer knows its scope.

PROTIP #5

USE ATTRIBUTES FROM SERIALIZER

TO OPTIMIZE QUERY WITH SELECT

DON'T USE PAGINATION

USE ORDER_QUERY

DON'T NEST OBJECTS IN JSON

SEND RELATIONS ELEMENTS AS SEPARATE COLLECTIONS

WRITING DATA CONVENTIONS

```
match '/*path' => 'actions#router', via: :post
```

```
/component_endpoint/action_name
```

```
class ComponentEndpoint

  def action_name
    # do action
  end

end
```

MY SERVICES

DISPATCHER

```
@Dispatcher = _.extend {}, Events
```

USE LOOSE COUPLING

STORE

```
Store.update \
  comments: { 123: { id: 123, body: 'Some comment...', author_ids: [1] },
  author: { 1: { id: 1, name: 'Marek' } }

comment = Store.get comments: 123

author = Store.get(author: comment.author_ids)[0]
```

MAKE SURE YOU HAVE ONLY ONE

INSTANCE OF PARTICULAR DATA IN THE APP

PROTIP #10

ONE OBJECT

IS A SPECIAL CASE OF COLLECTION

PROTIP #11

IT'S BETTER TO

HAVE DECORATORS IN THE FRONTEND

ALIASES

```
dictionary:
  author: 'users'
```

LCA

One render per Store update.

Fetcher

Collects queries for API and sends it in one.

MY MIXINS

ACTIONS HANDLER

```
actions:
  local:
    change_tab: (tab) ->
      # ...
  api:
    message_readed: (id) ->
      # ...
  realtime:
    message_readed: (id) ->
      # ...
```

SCOPE

Keeps information about subset of Store collection and listens for updates.

API STATE COMPONENT

Pairs component with its API endpoint. It uses Scope.
It's able to load more or newer elements of the collection.

ASC SUPLEMENT

Asks for more data for an object. Cooperates with Fetcher.

PROTIP #12

USE LO-DASH

INSTEAD OF UNDERSCORE

BONUS PROTIP

IDENTITY FUNCTION FOR NIL

```
class NilClass
  def method_missing(*)
    self
  end
end
```

PILOT

M.PIASECKI@PILOT.CO