2020/02/06
OOP

Architectures for Modern Web Front Ends

Stefan Tilkov
@stilkov

Lucas Dohmen
@moonbeamlabs

INNOQ

Annoying your app users in 10 easy steps

1.
Forbid the use of the back and forward buttons

2.

Send them to the home page when they hit "refresh" …

3.

… or at least ensure the browser pops up a warning window

4.

Make sure they can’t open a second browser window

5.
Let them see UI decoration and ads first, content last

6.
Make sure they can't bookmark or send a link

7.
Don't let Google index anything

8.

Show users a picture of your app – it's surely better than nothing

9.

Disable assistive technologies. Who needs a screen reader, anyway?

10.
Ensure non-functioning JavaScript gives them a blank page

History repeating …

Web

CORBA

WS-*

REST

What’s the client side analogy?

"Web service"[1)]

- Uses HTTP as transport
- Ignores HTTP verbs
- Ignores URIs
- Exposes single "endpoint"
- Fails to embrace the Web

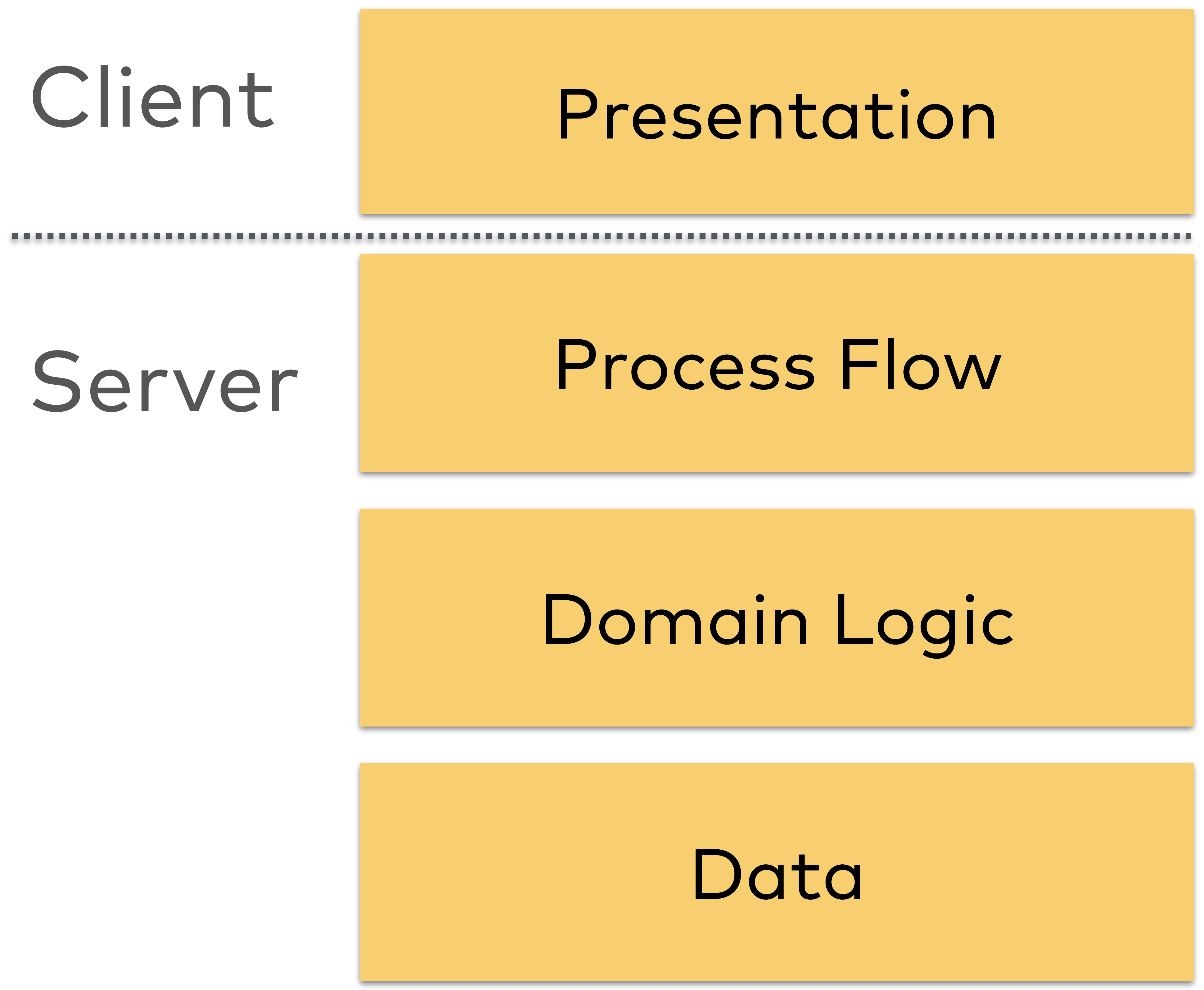
"Web app"[2)]

- Uses browser as runtime
- Ignores forward, back, refresh
- Does not support linking
- Exposes monolithic "app"
- Fails to embrace the browser

1) in the SOAP/WSDL sense

2) built as a careless SPA

The web-native way of distributing logic



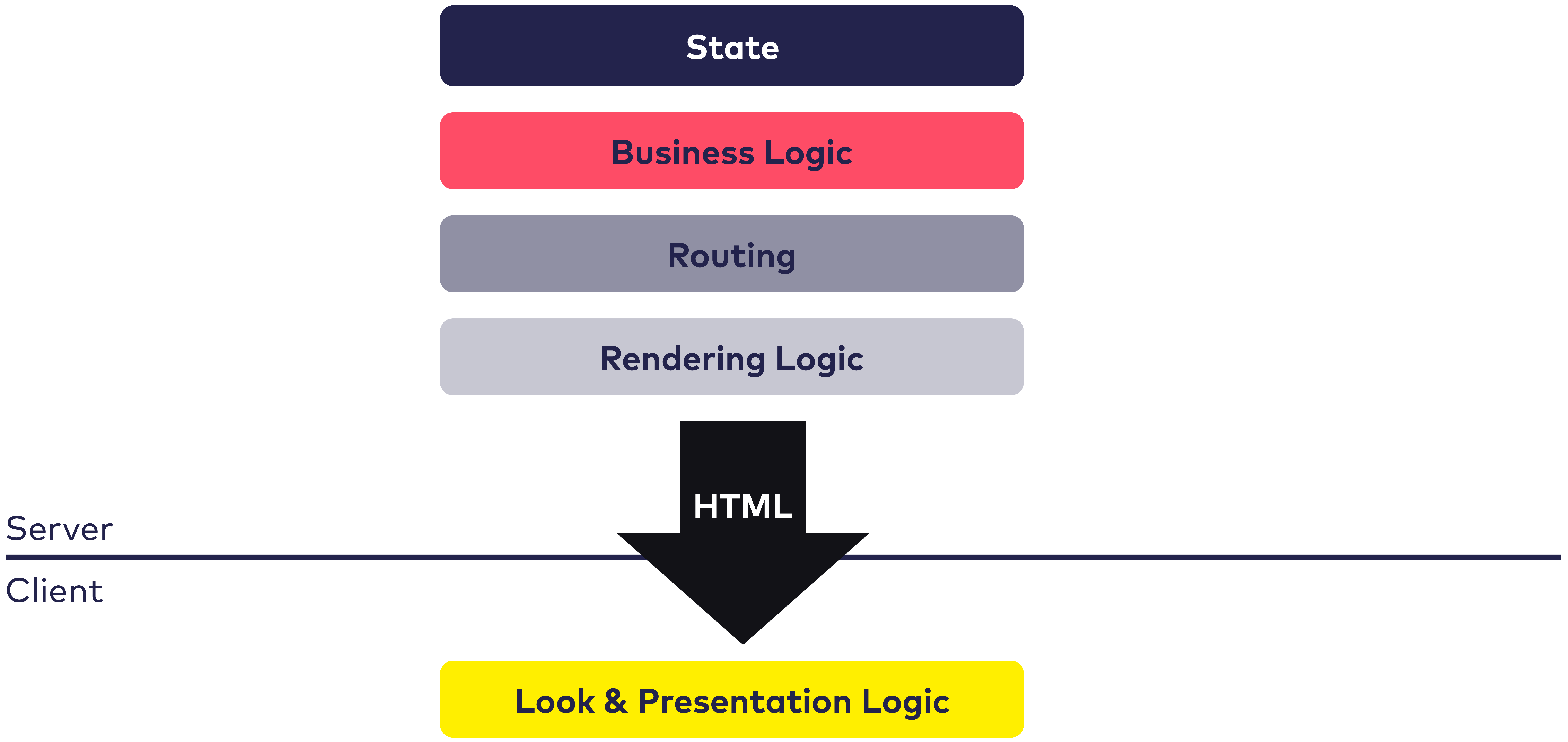
- Rendering, layout, styling on an unknown client
- Logic & state machine on server
- Client user-agent extensible via code on demand

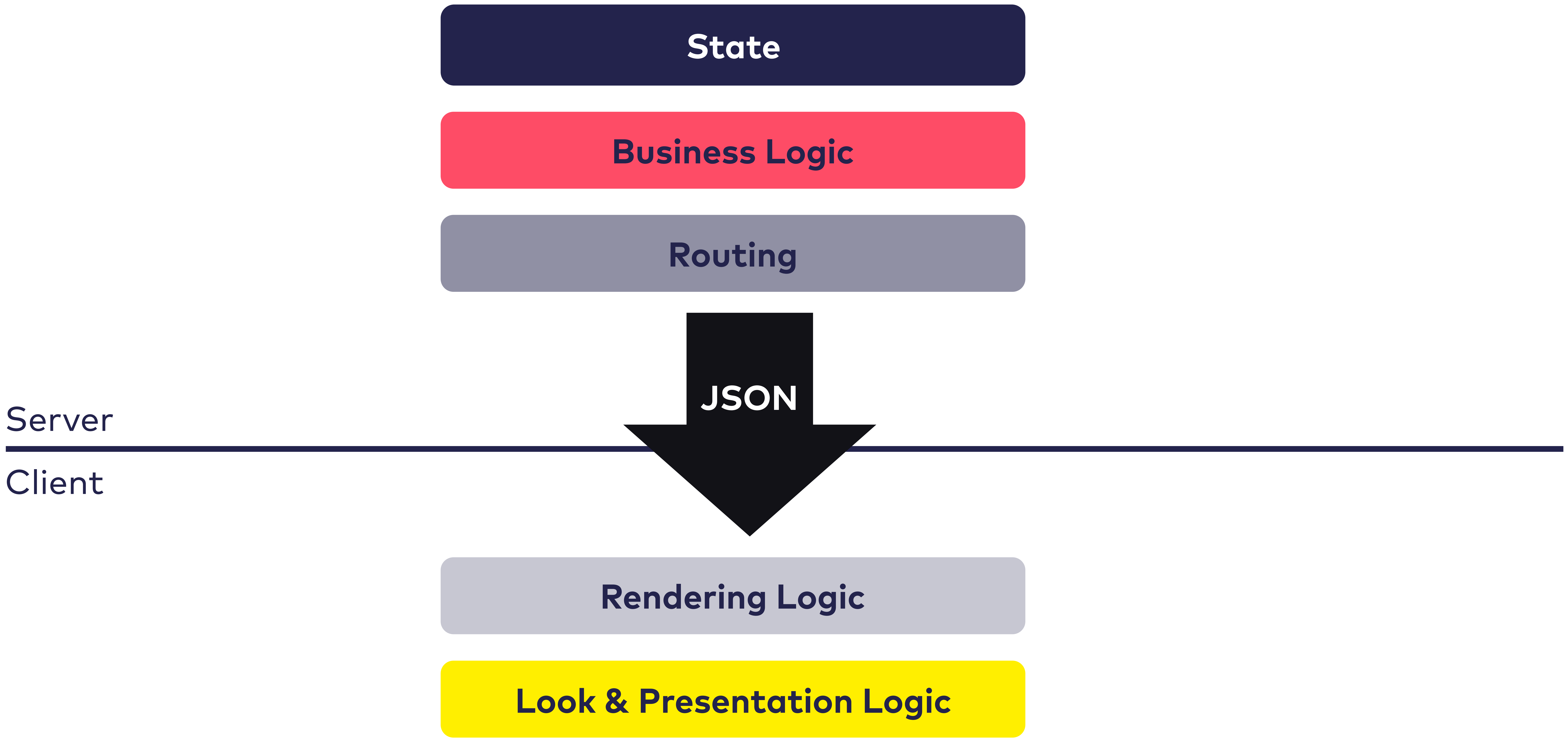
HTML & Hypermedia

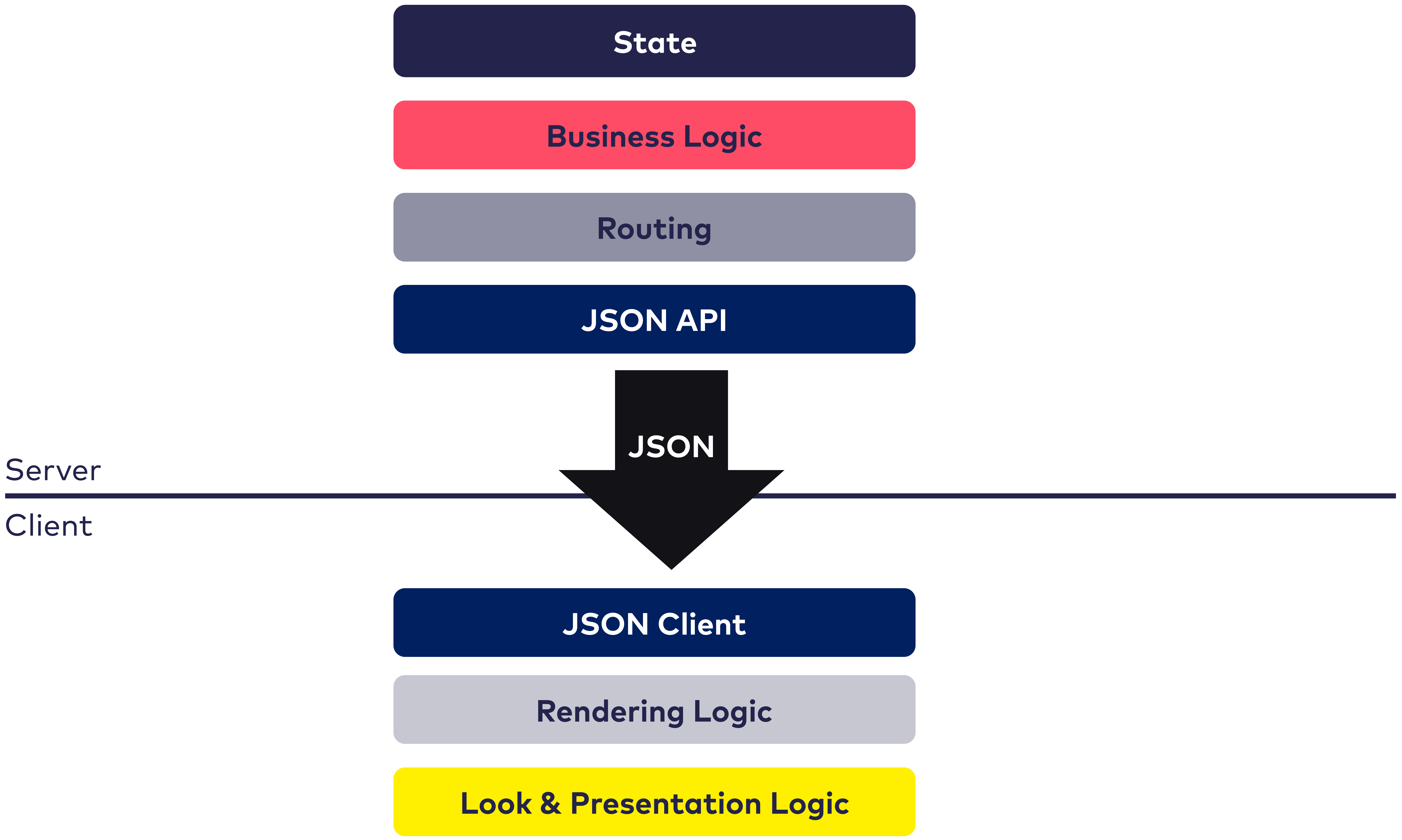
- In REST, servers expose a hypermedia format
  - Option 1: Just invent your own JSON-based, incomplete clone
  - Option 2: Just use HTML
- Clients need to be RESTful, too
  - Option 1: Invent your own, JS-based, buggy, incomplete implementation
  - Option 2: Use the browser

A great REST hypermedia API is very similar to a simple, server-sided rendered web application

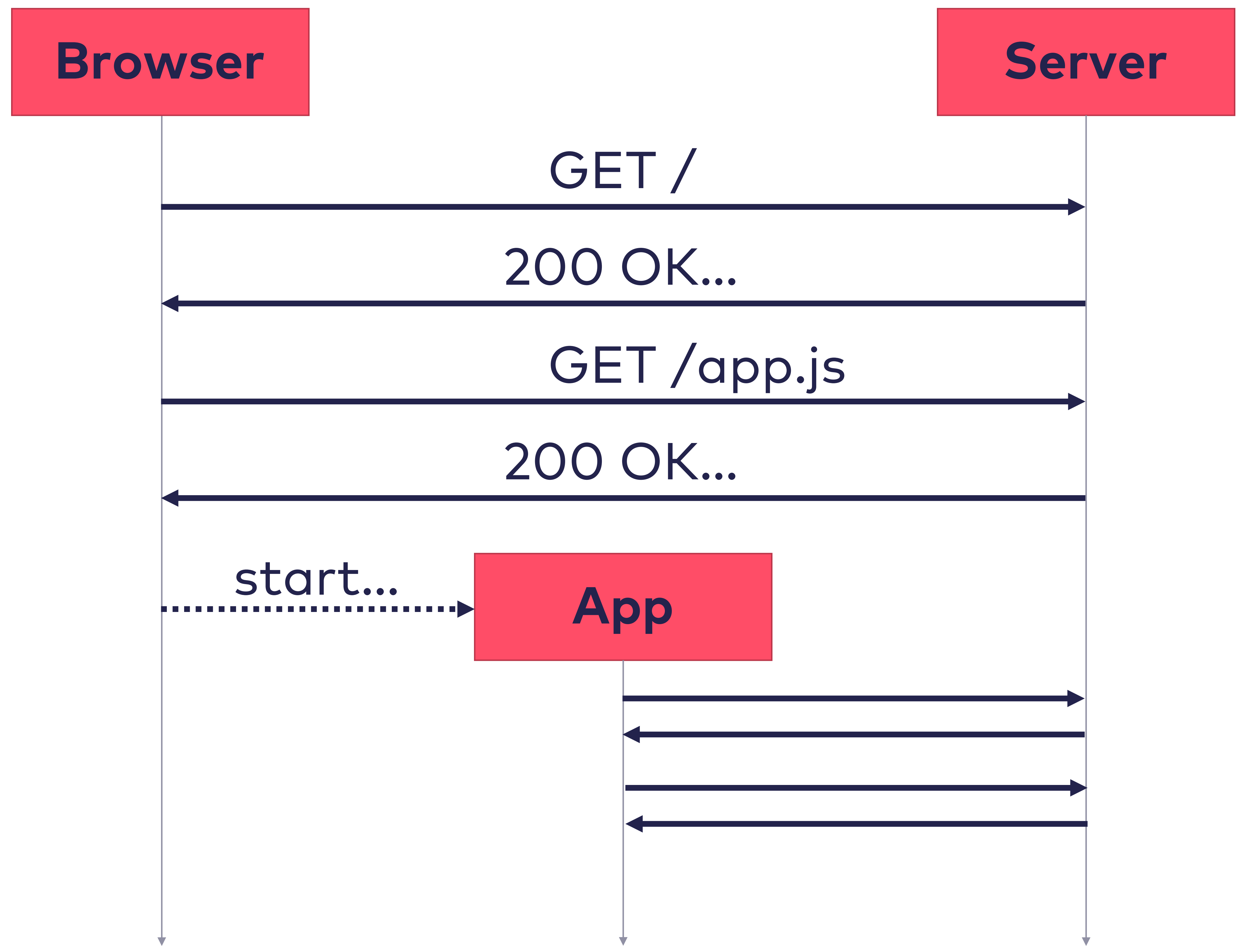
The role of JS in modern Web applications


State
Business Logic
Routing
Rendering Logic
HTML
Server
Client
Look & Presentation Logic


State
Business Logic
Routing
JSON
Server
Client
Rendering Logic
Look & Presentation Logic


State
Business Logic
Routing
JSON API
JSON
Server
Client
JSON Client
Rendering Logic
Look & Presentation Logic

Why Routing?

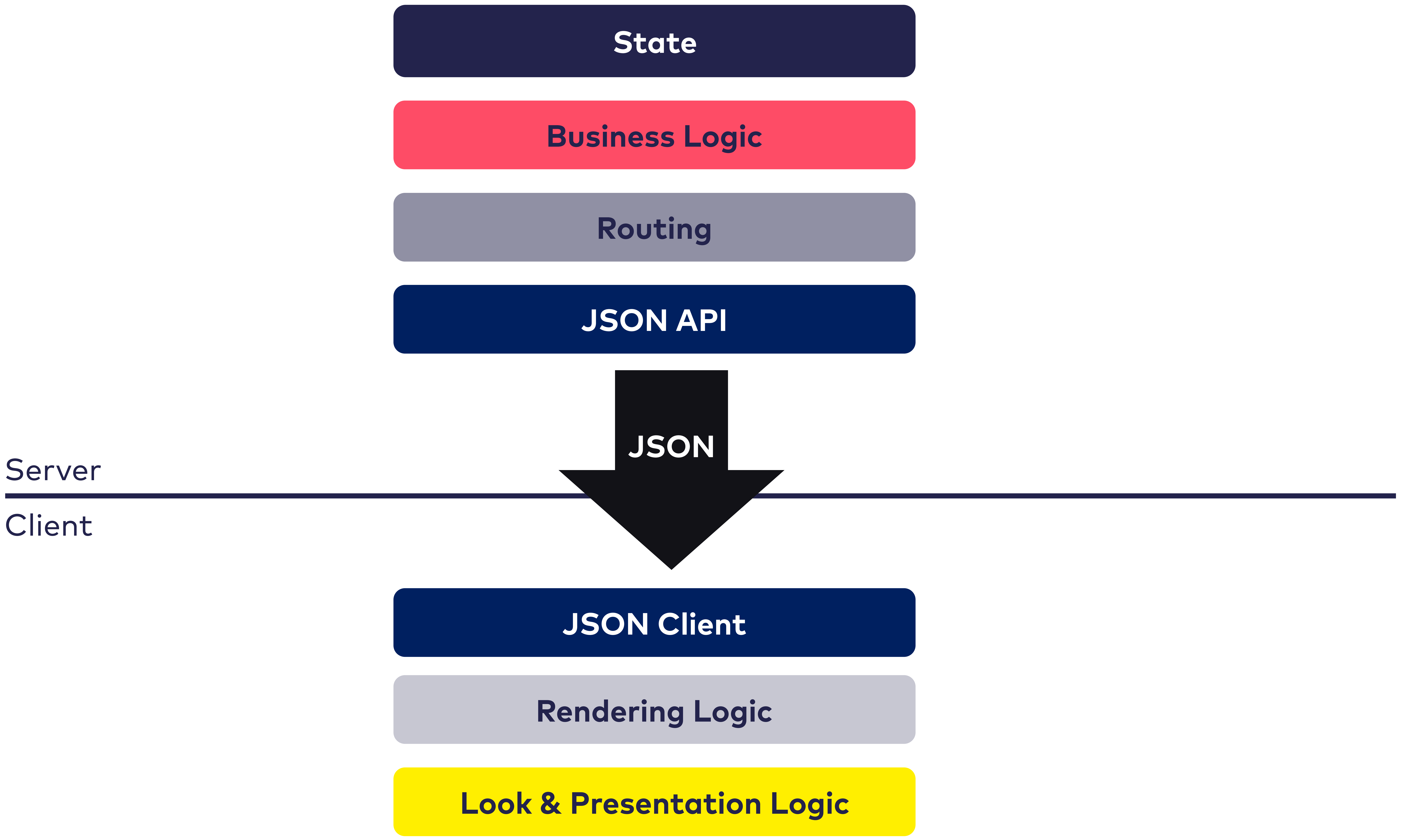


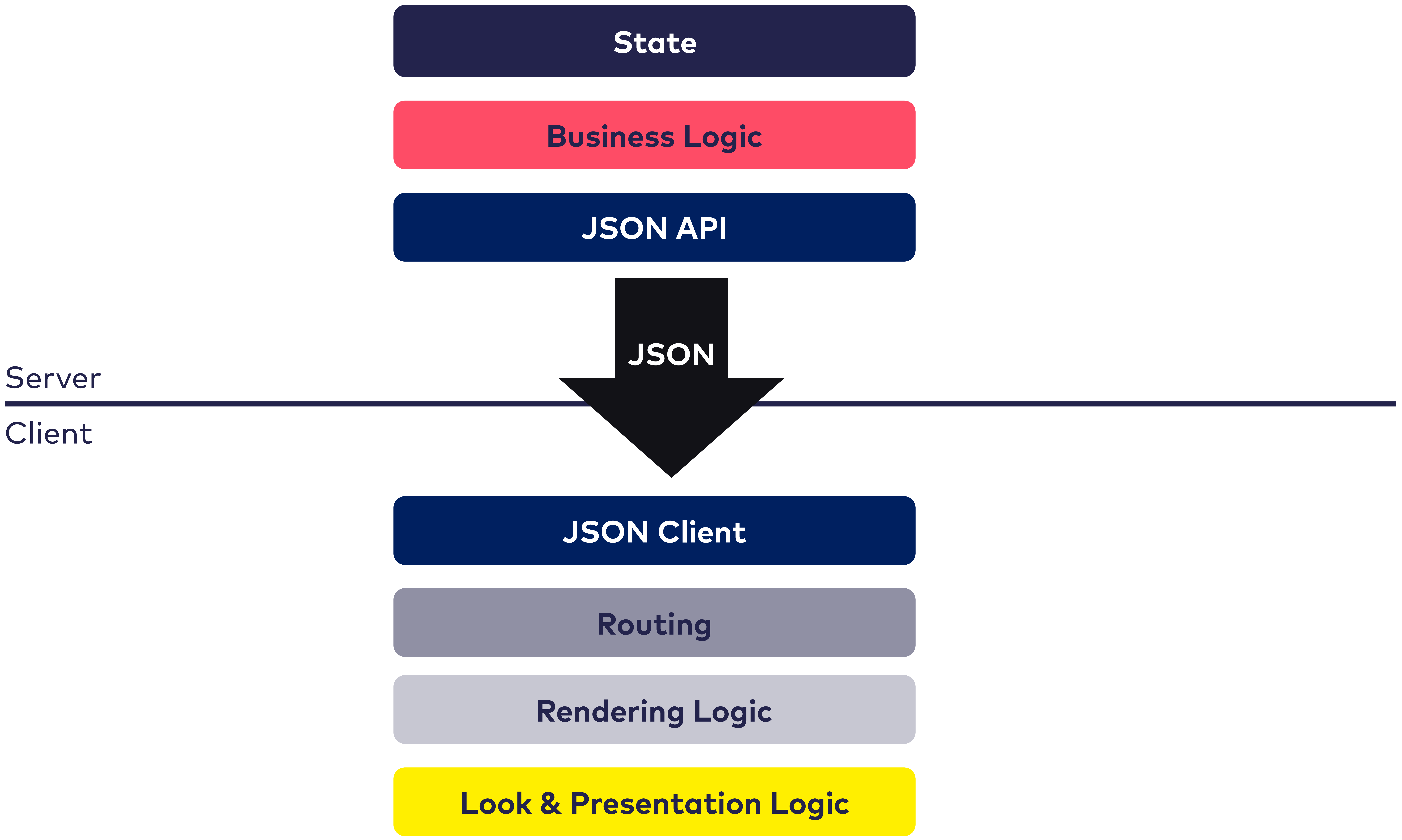
Bookmarks?

Deep links?

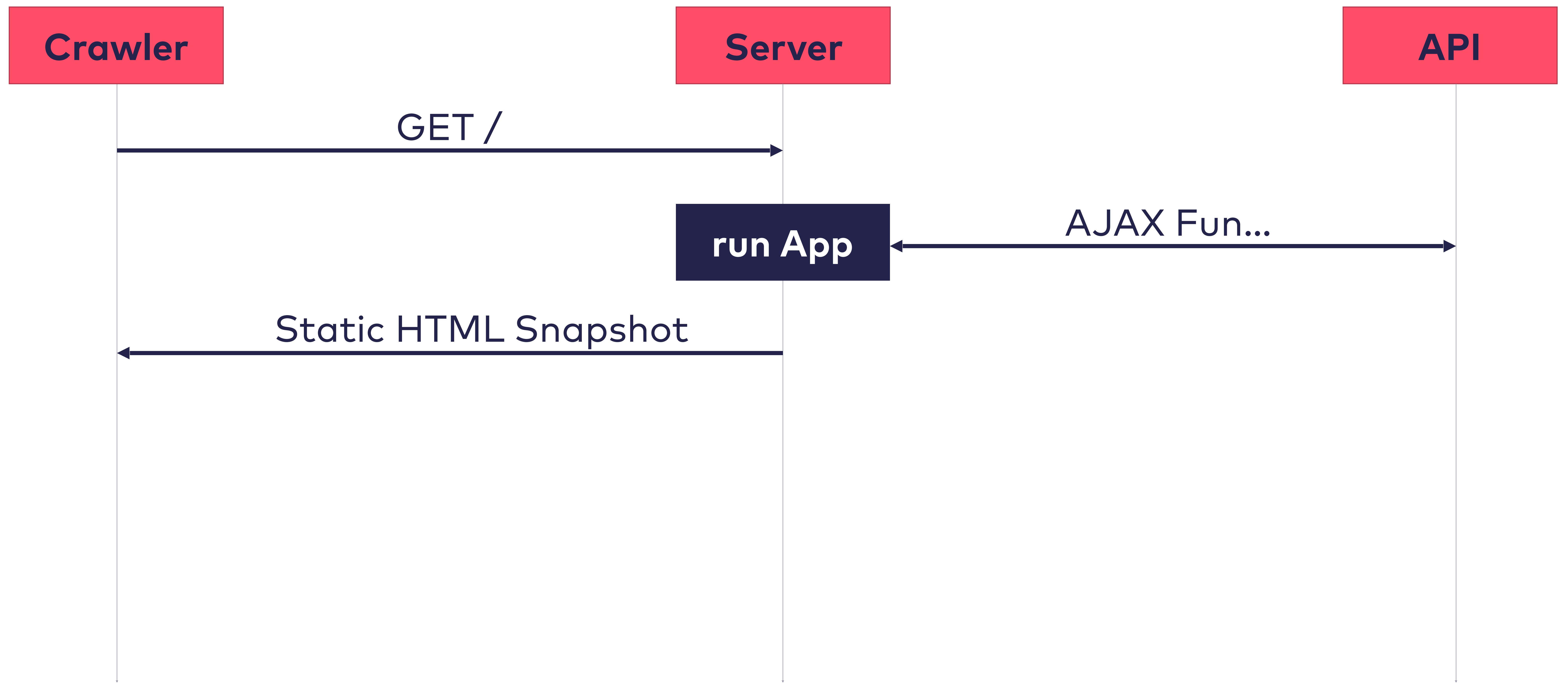
Reload?

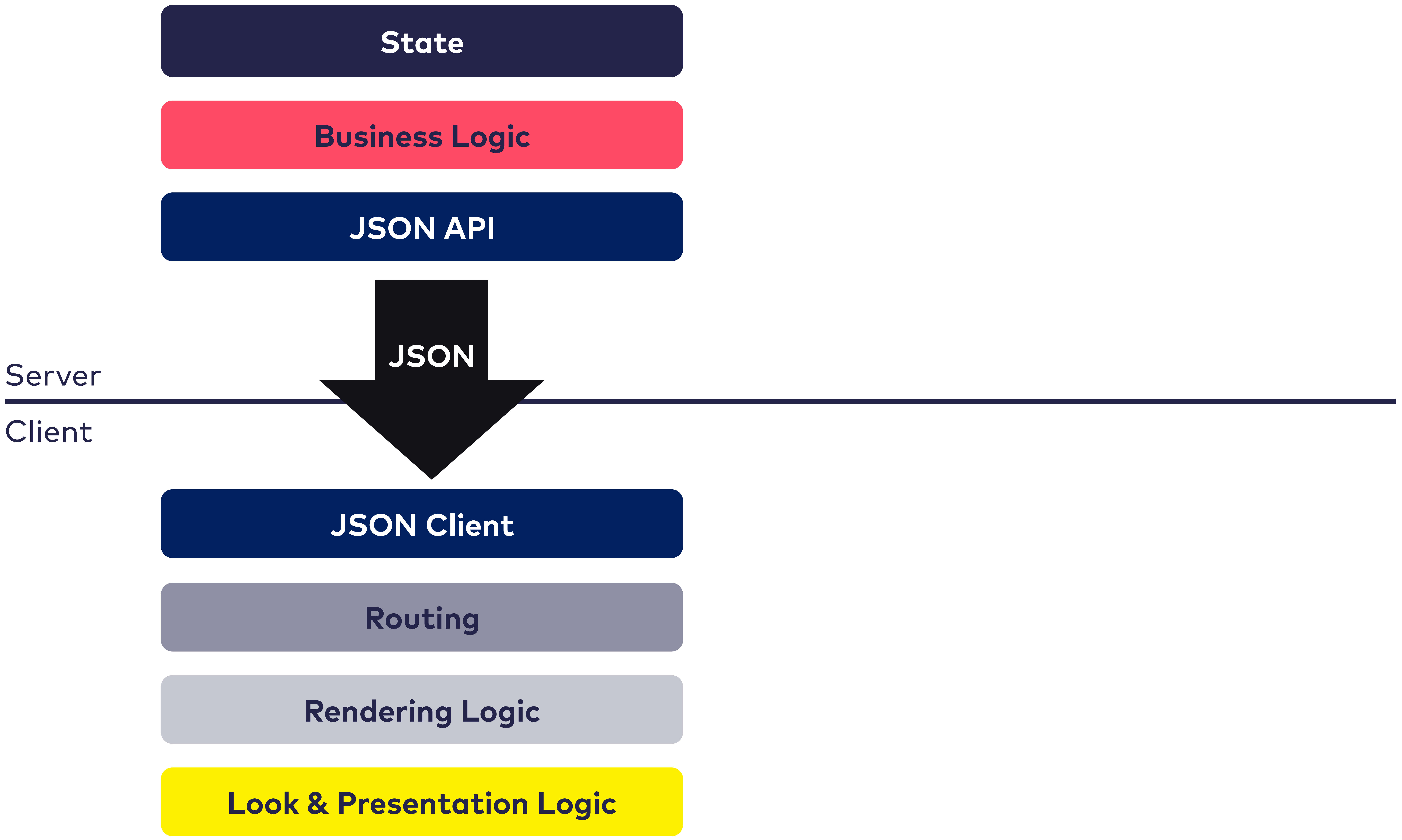
Solution:
Store some app state in the URI!

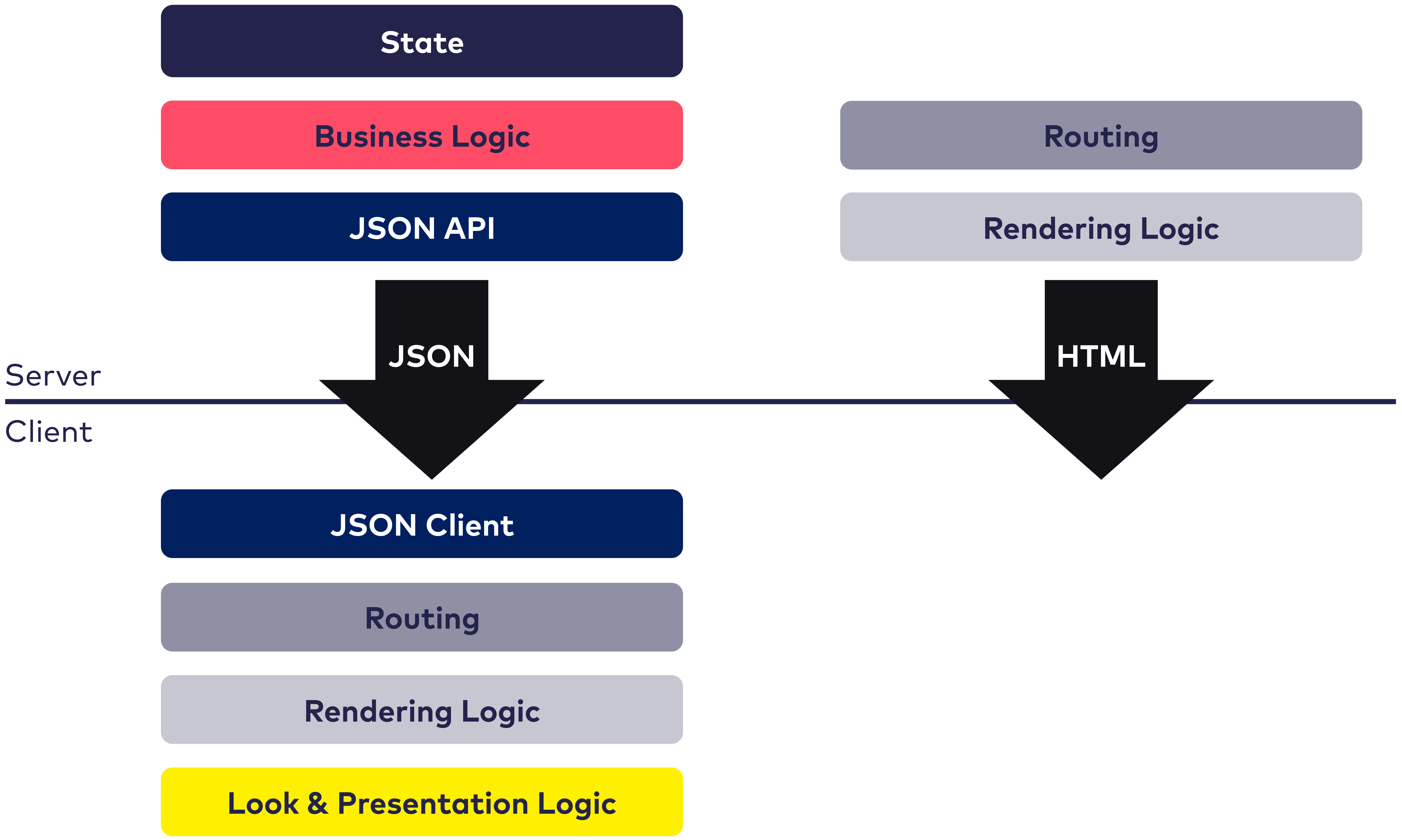

State
Business Logic
Routing
JSON API
JSON
Server
Client
JSON Client
Rendering Logic
Look & Presentation Logic

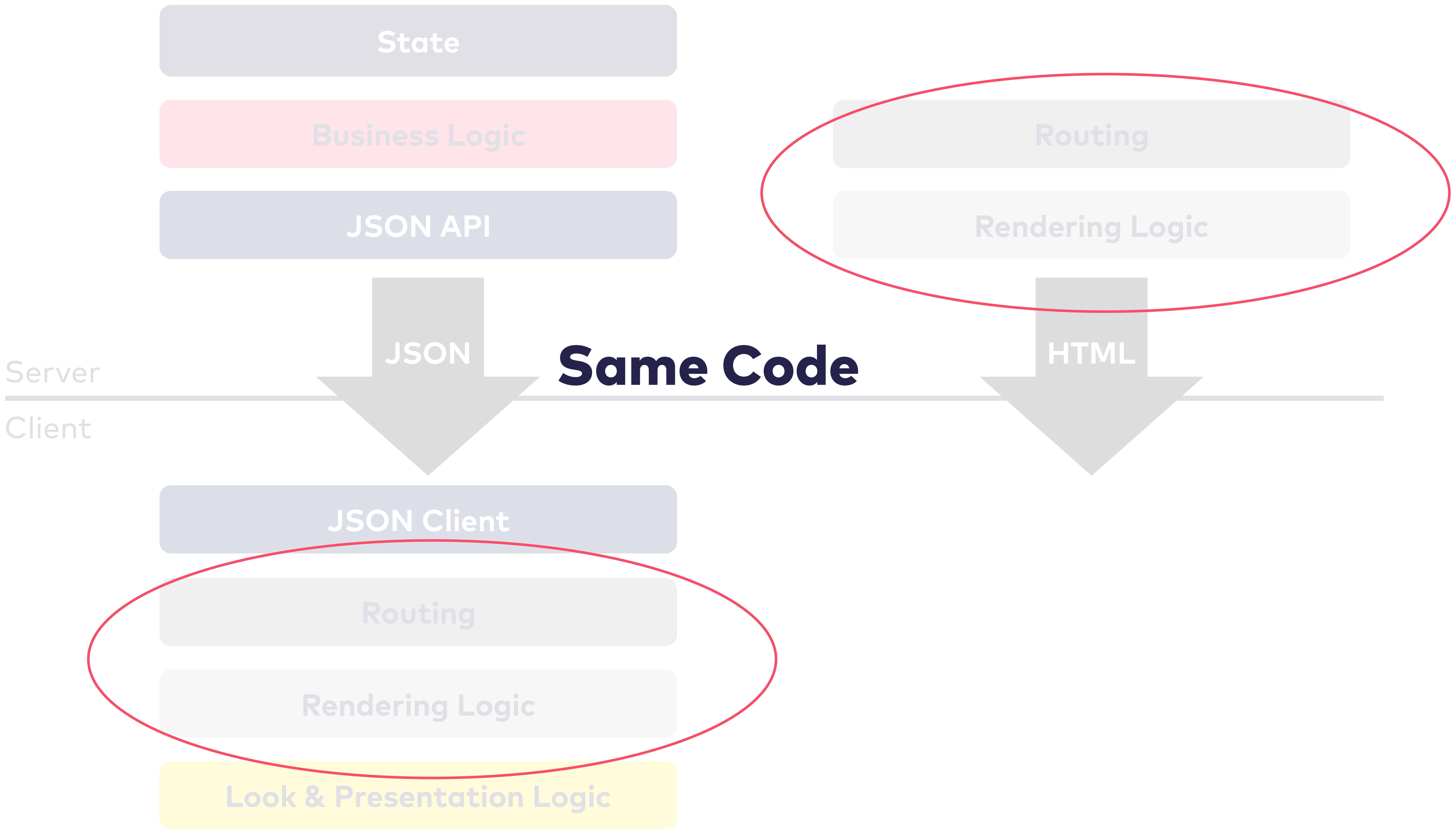

State
Business Logic
JSON API
JSON
Server
Client
JSON Client
Routing
Rendering Logic
Look & Presentation Logic

Searchability




State
Business Logic
JSON API
JSON
Server
Client
JSON Client
Routing
Rendering Logic
Look & Presentation Logic


State
Business Logic
JSON API
Routing
Rendering Logic
JSON
HTML
Server
Client
JSON Client
Routing
Rendering Logic
Look & Presentation Logic

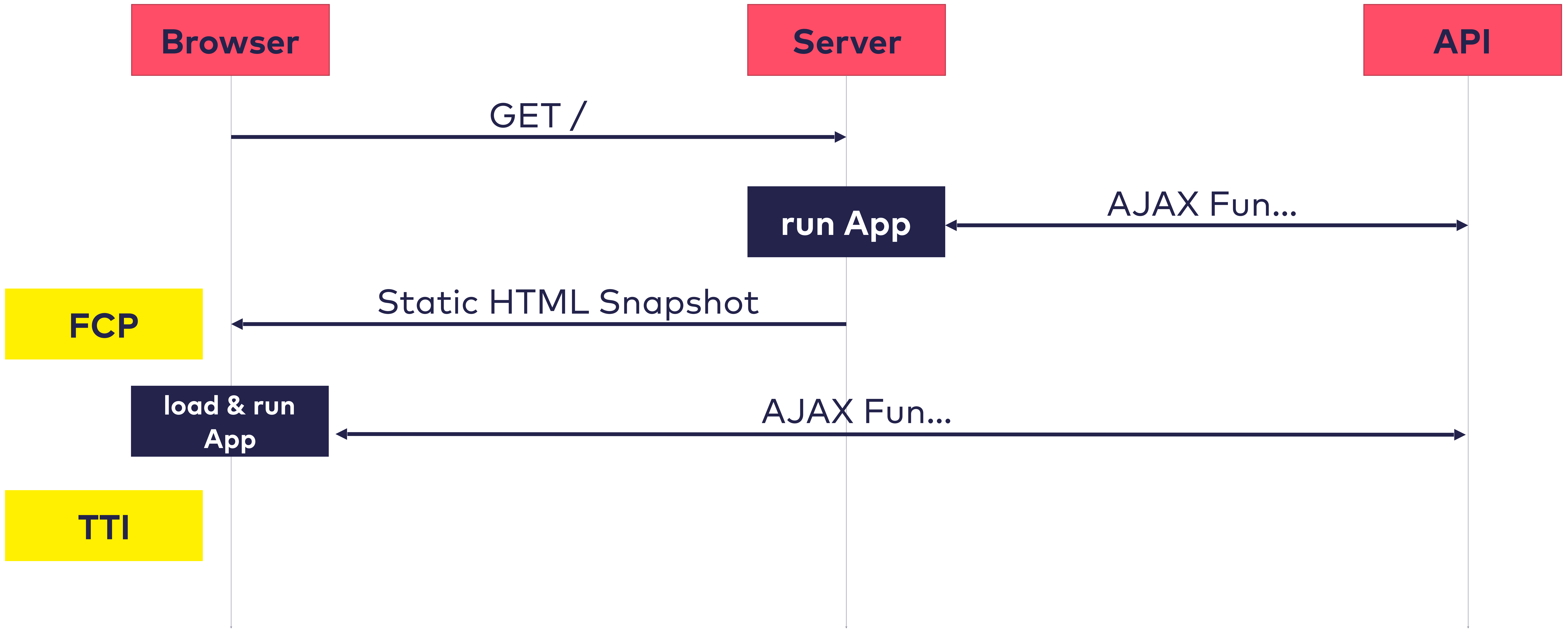

State
Business Logic
JSON API
Routing
Rendering Logic
JSON
HTML
Server
Client
JSON Client
Routing
Rendering Logic
Look & Presentation Logic

Same Code

"All your users are non-JS users while they're downloading your JS"

Jake Archibald, developer advocate for Google Chrome

Prerendering

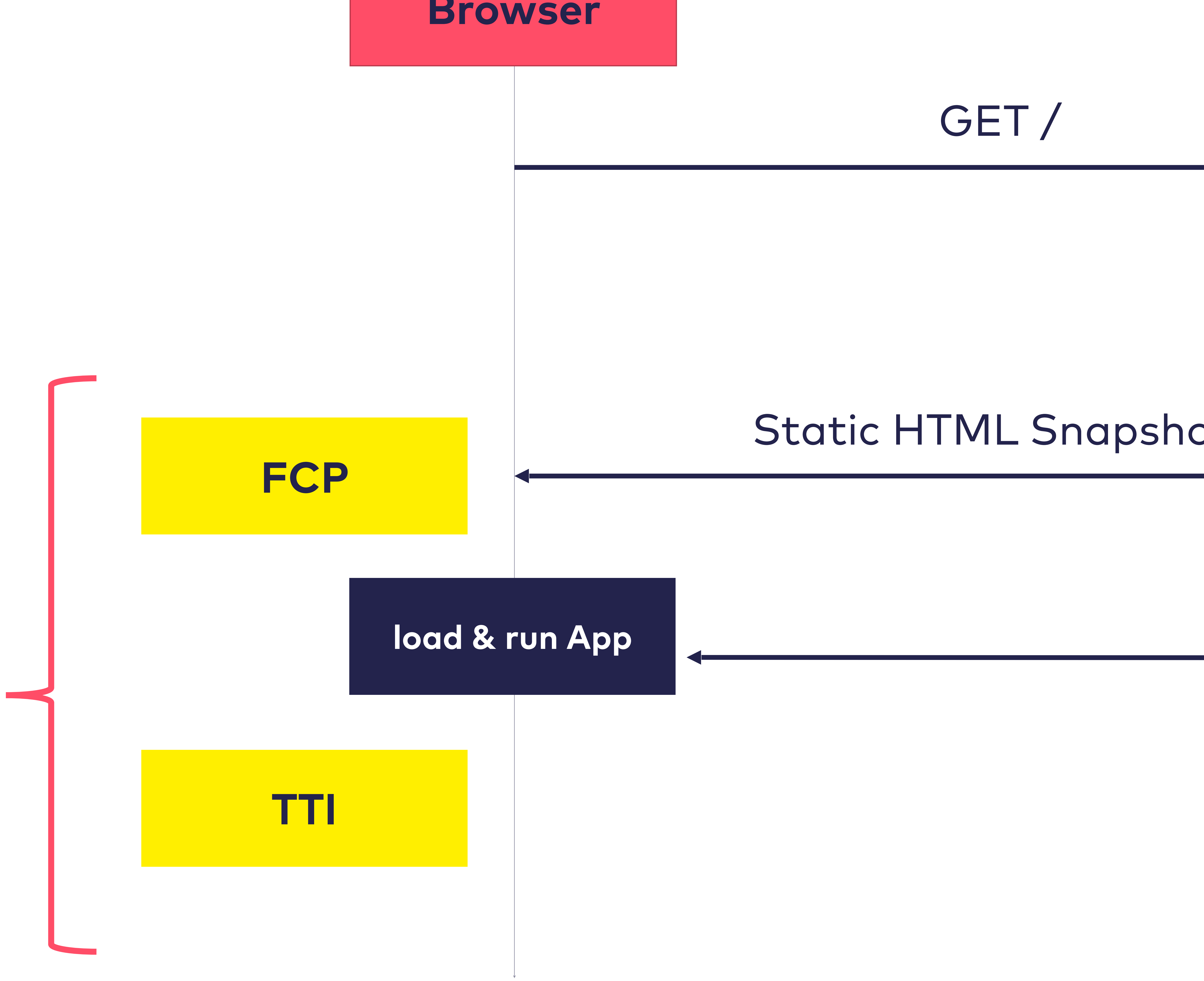


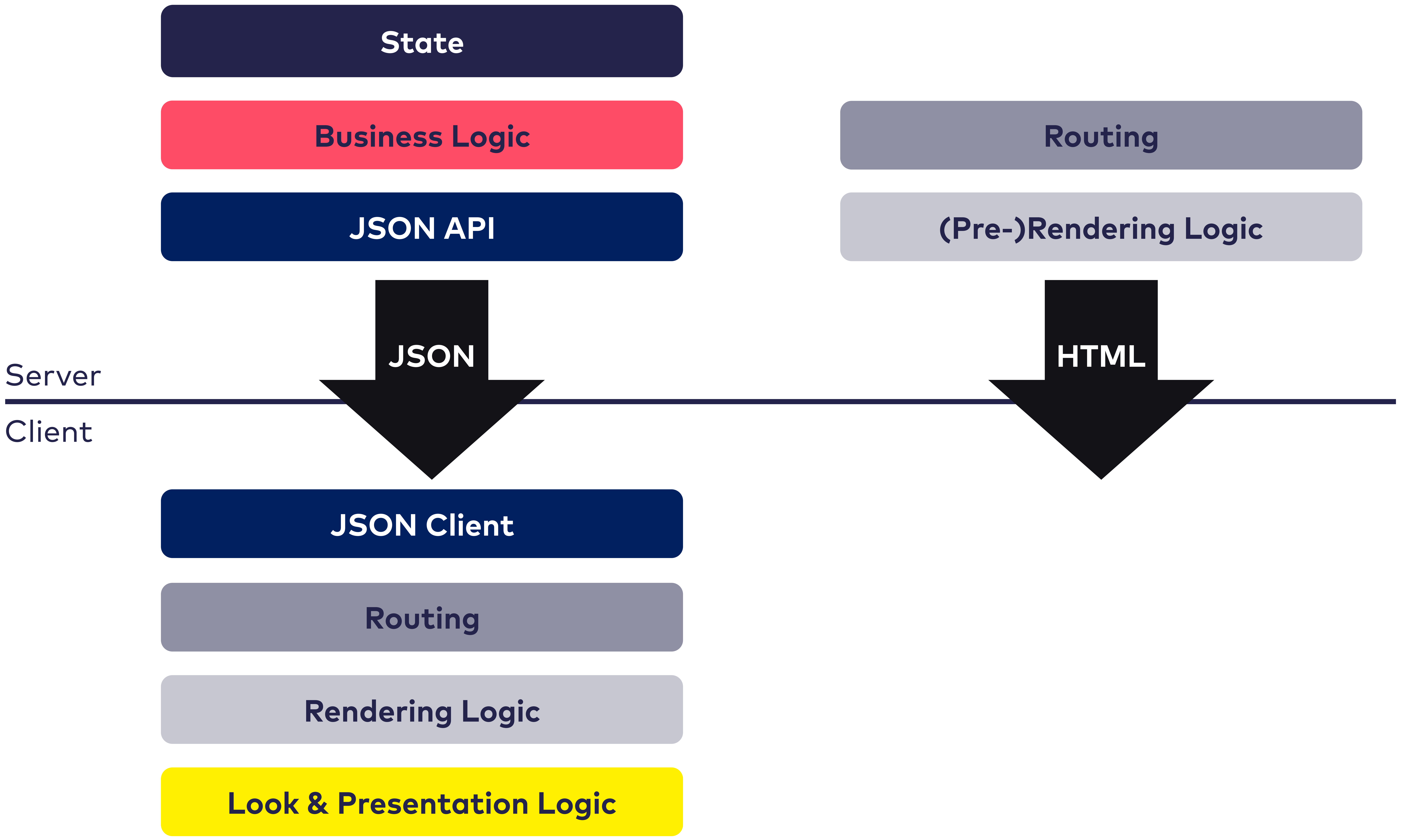
Hydration

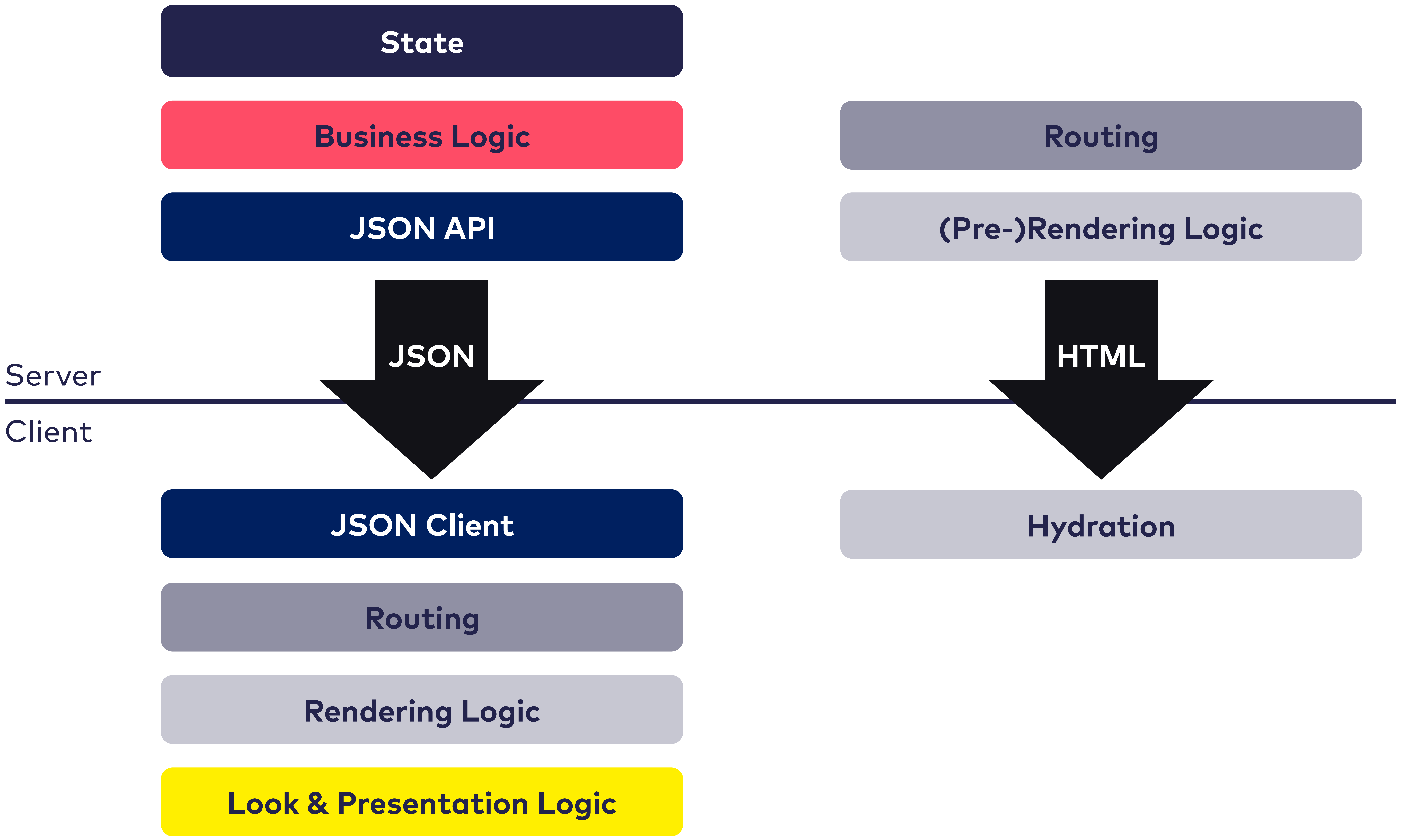
How to simulate readiness?

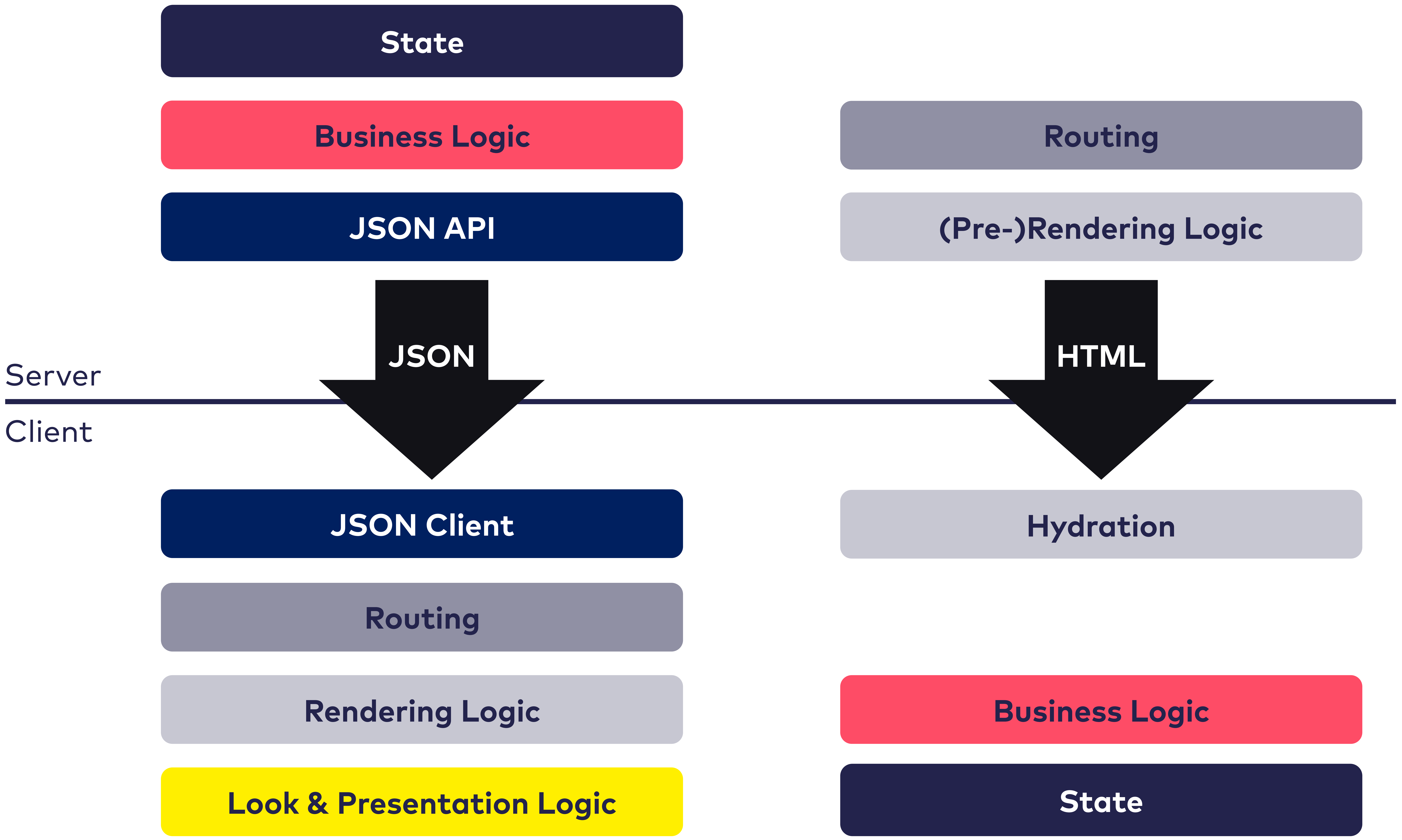
What about Events (Clicks etc)?

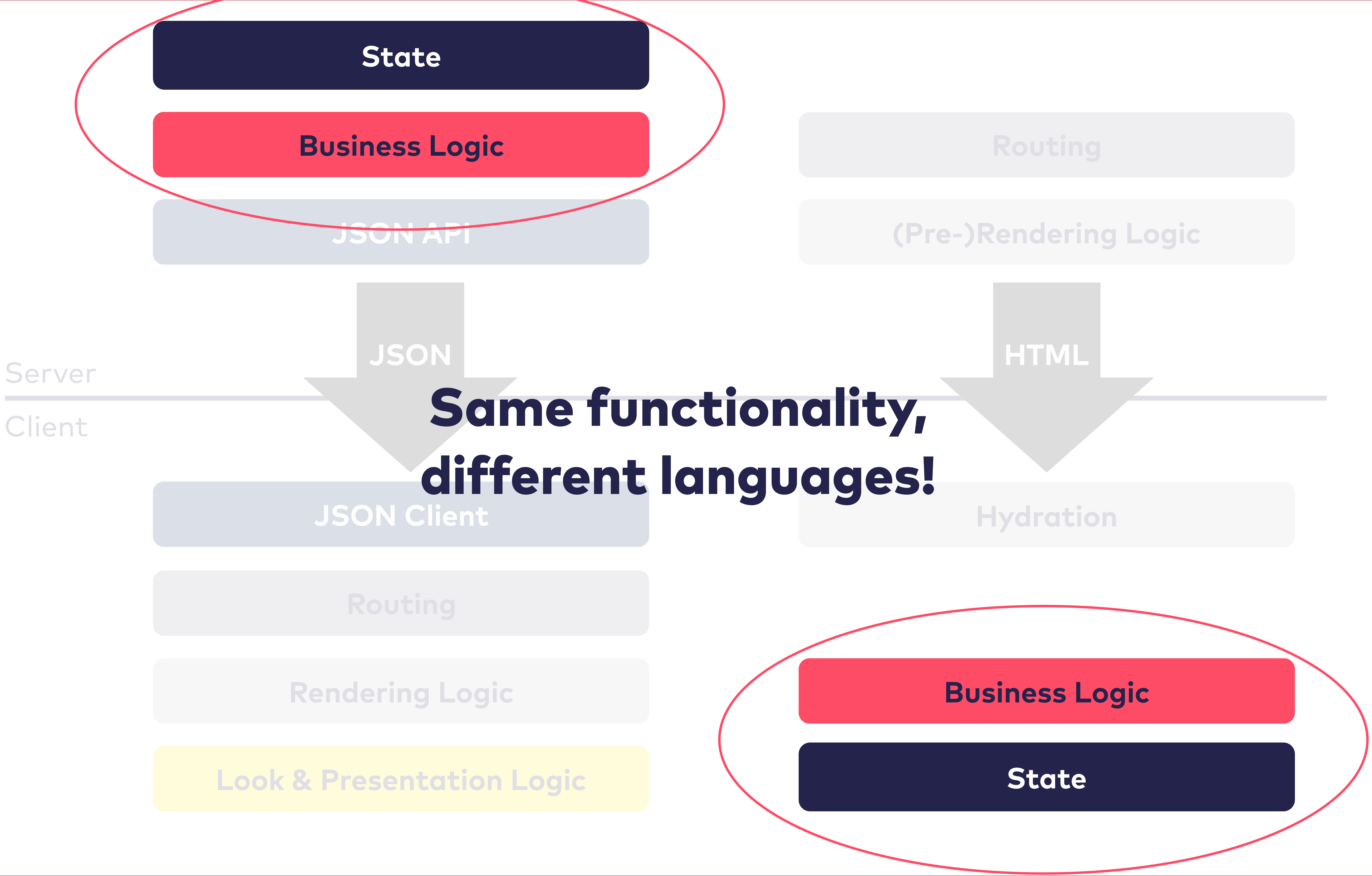
How to match server-side HTML to client-side DOM?



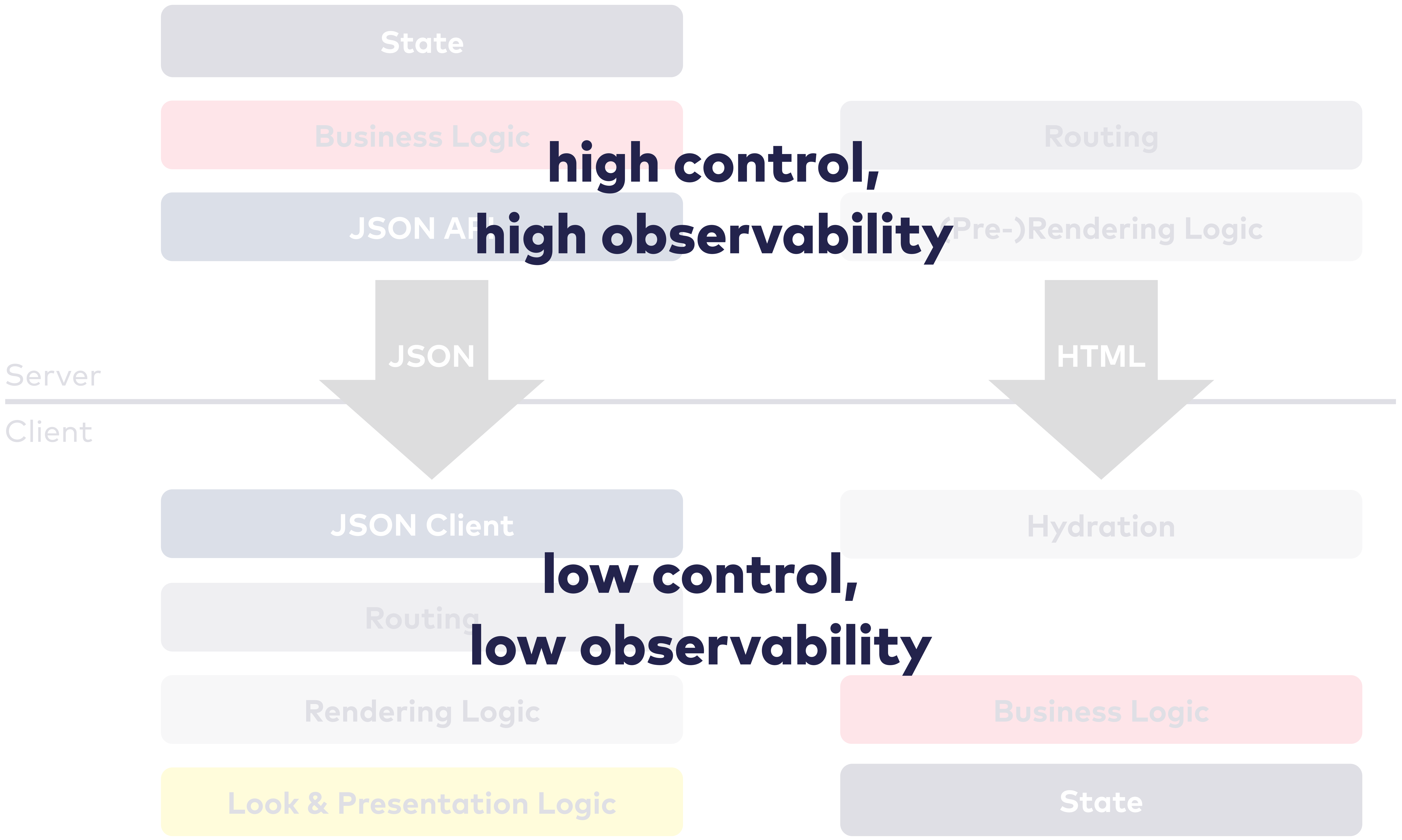

State
Business Logic
JSON API
JSON
Server
Client
JSON Client
Routing
Rendering Logic
Look & Presentation Logic
Routing
(Pre-)Rendering Logic
HTML


State
Business Logic
JSON API
JSON
Server
Client
JSON Client
Routing
Rendering Logic
Look & Presentation Logic
Routing
(Pre-)Rendering Logic
HTML
Hydration


State
Business Logic
JSON API
JSON
Server
Client
JSON Client
Routing
Rendering Logic
Look & Presentation Logic
Routing
(Pre-)Rendering Logic
HTML
Hydration
Business Logic
State



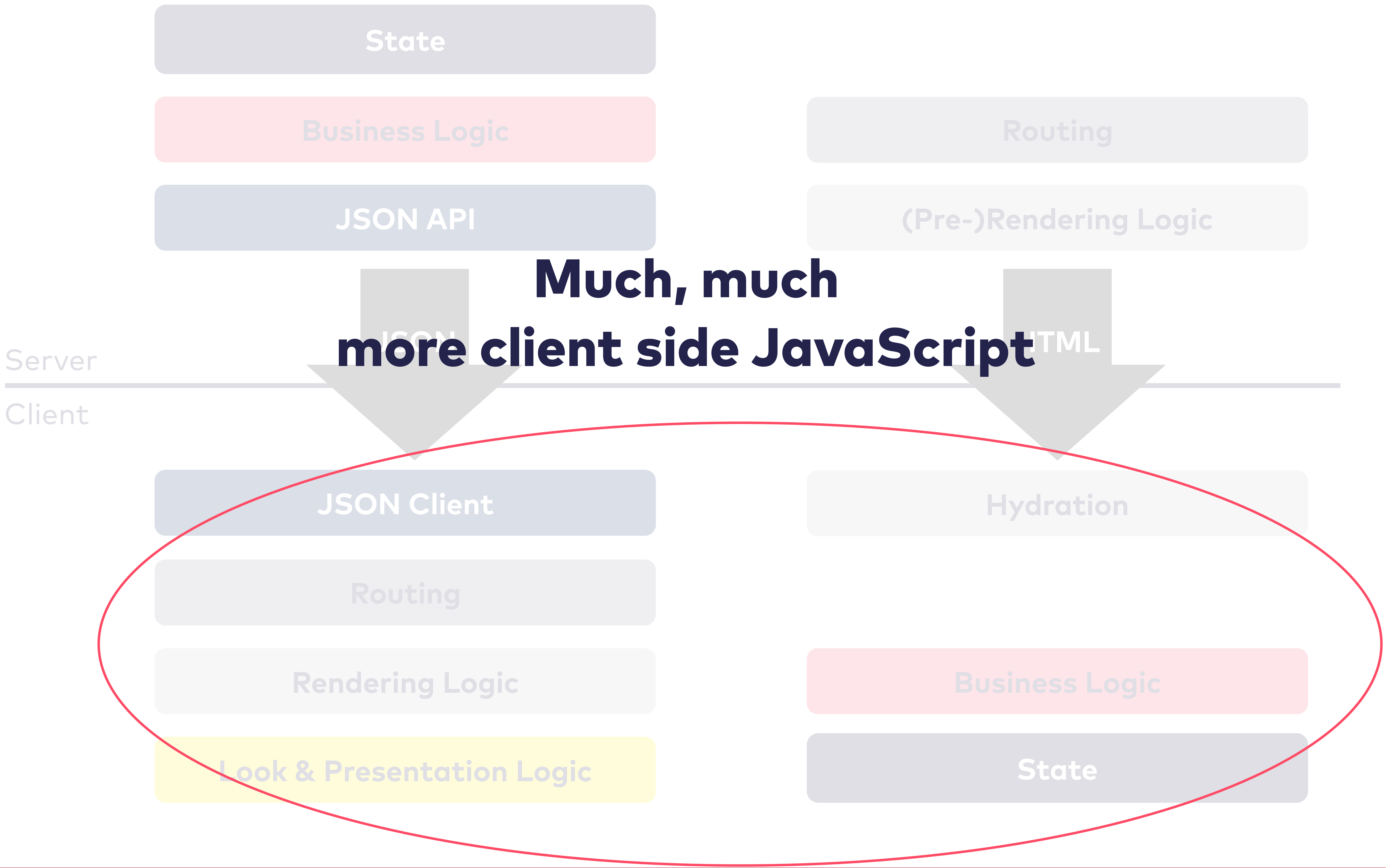
Same functionality, different languages!



high control,
high observability

low control,
low observability

Much, much more client side JavaScript



Resilience

Modern API in JS

```
customElement.define(
  "my-element",
  MyElement
);
```

Firefox 63: It works

Chrome 69: Exception

Modern API in CSS

```
.item {
  display: contents;
}
```

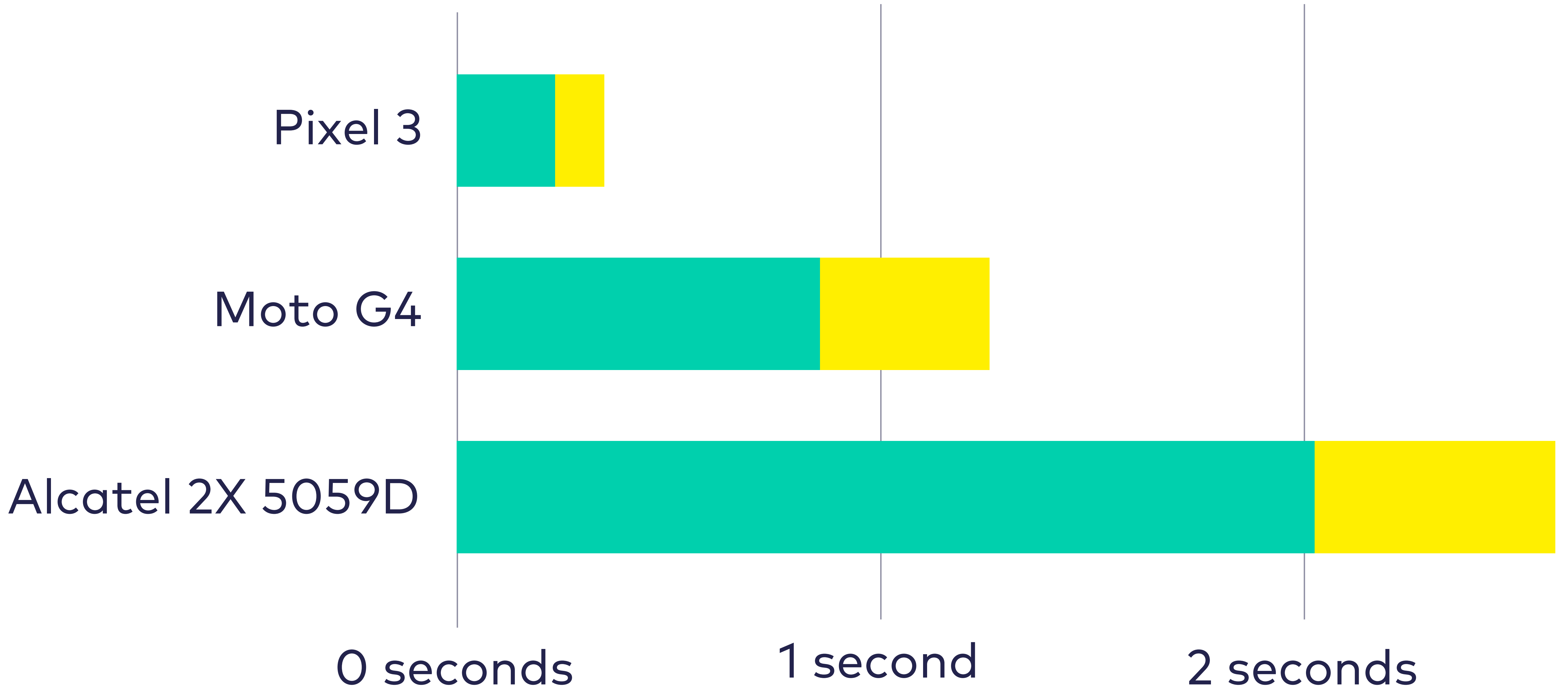
Firefox 63: It works

Chrome 69: Skips that line

“JavaScript is the most expensive part of your page”

Addy Osmani, Speed team lead for Google Chrome

Cost of JavaScript on Reddit.com



🟩 Main thread   🟨 Worker thread

The cost of JavaScript in 2019

Test your app on
real, low-cost devices and slow networks

(No, an emulator is not enough)

DuckDuckGo — Privacy, simpl
duckduckgo.com
Privacy, simplified.
DuckDuckGo
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
top
Filter
Default levels

```
> for(let i=0; i<1000000000; i++) {
      Math.pow(i, i);
  }
```

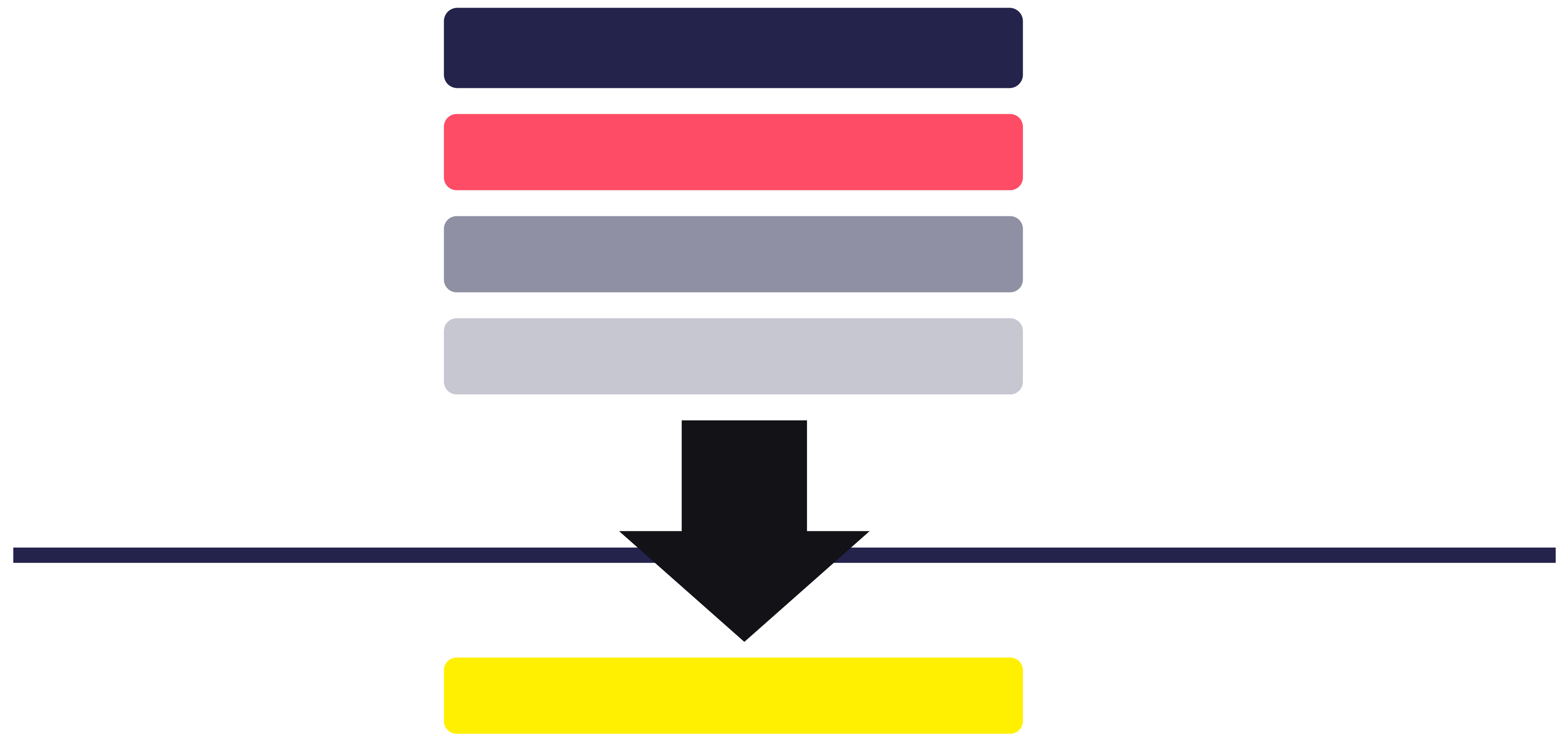
RAGE CLICKS

"15% of users tried to interact sometime between onload and interactive."

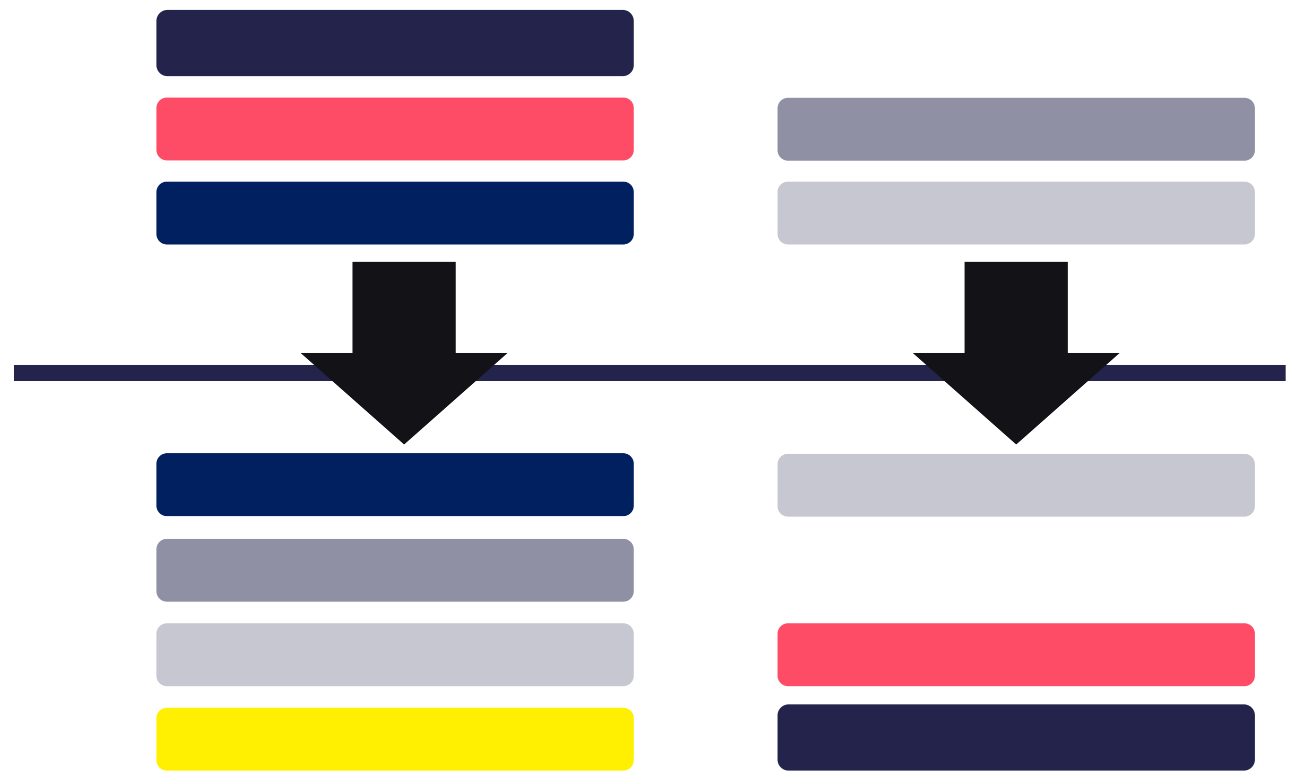
Akamai: Metrics That Matter

Hydration is not
a progressive enhancement,
it's an **uncanny valley**

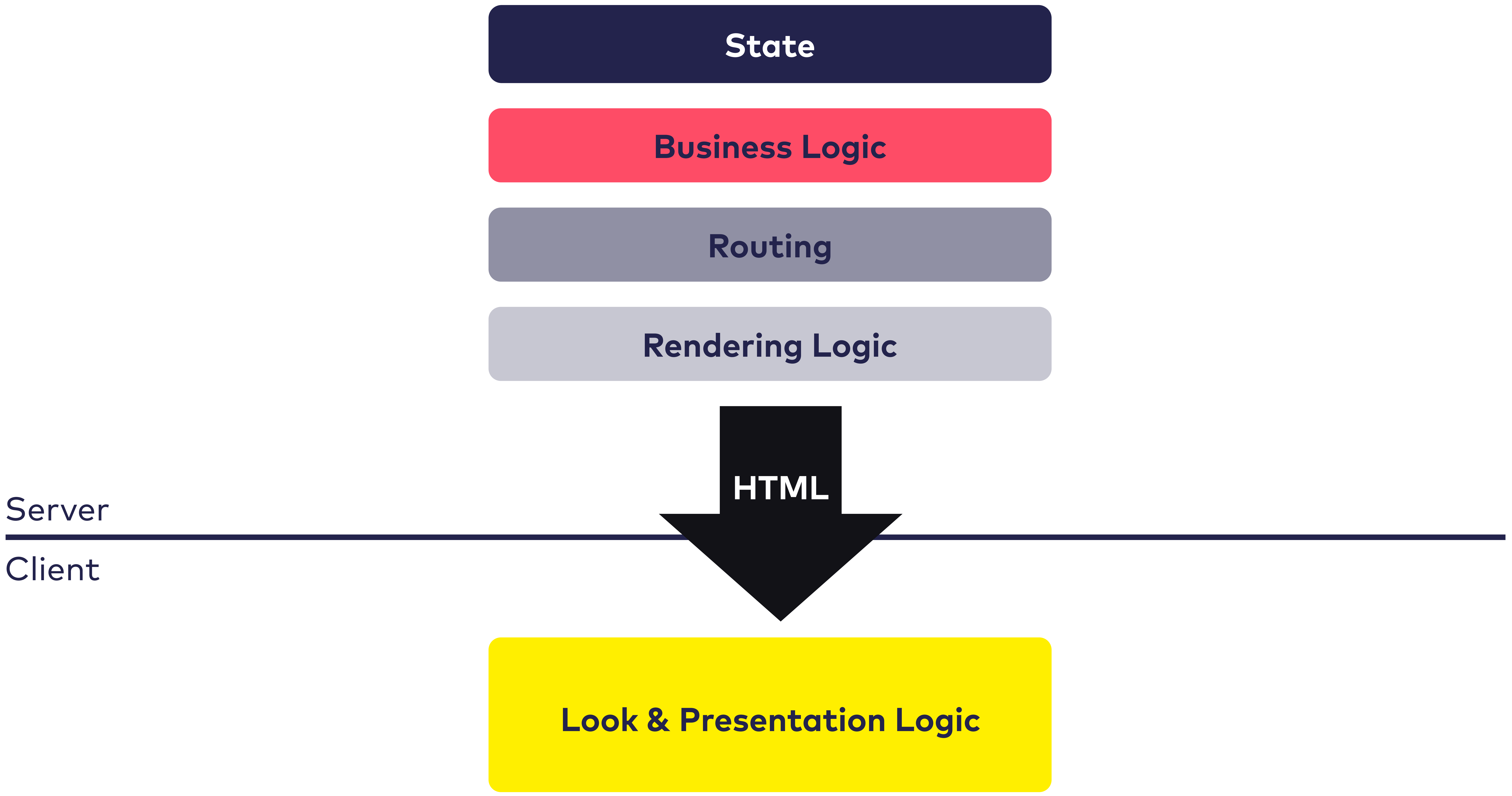
Now what?

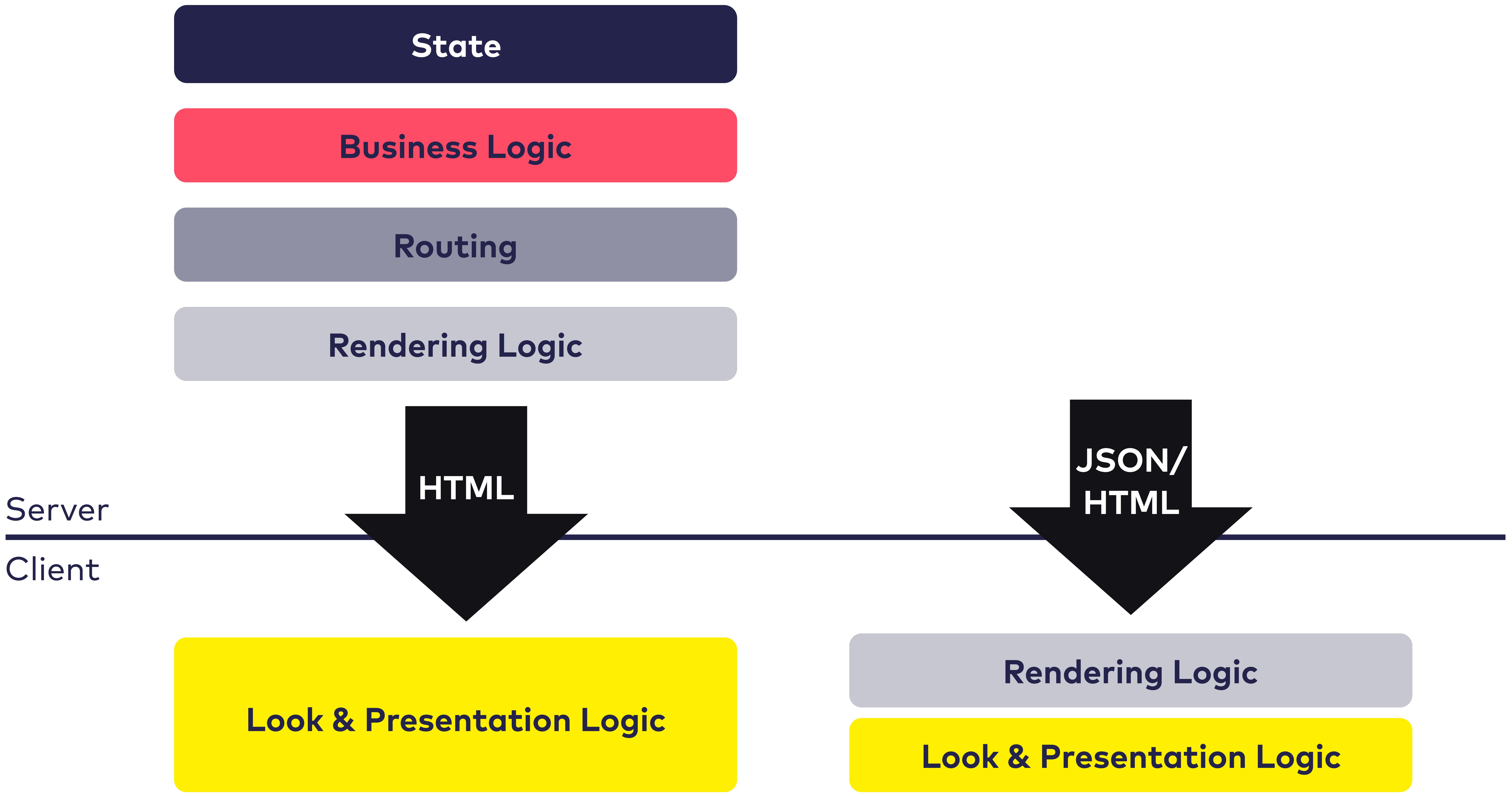


- Server-side state handling
- Simpler
- More resilient & observable
- Smaller client footprint
- Better performance



- Client-side state handling
- Better offline support
- Closer to desktop model
- Better performance

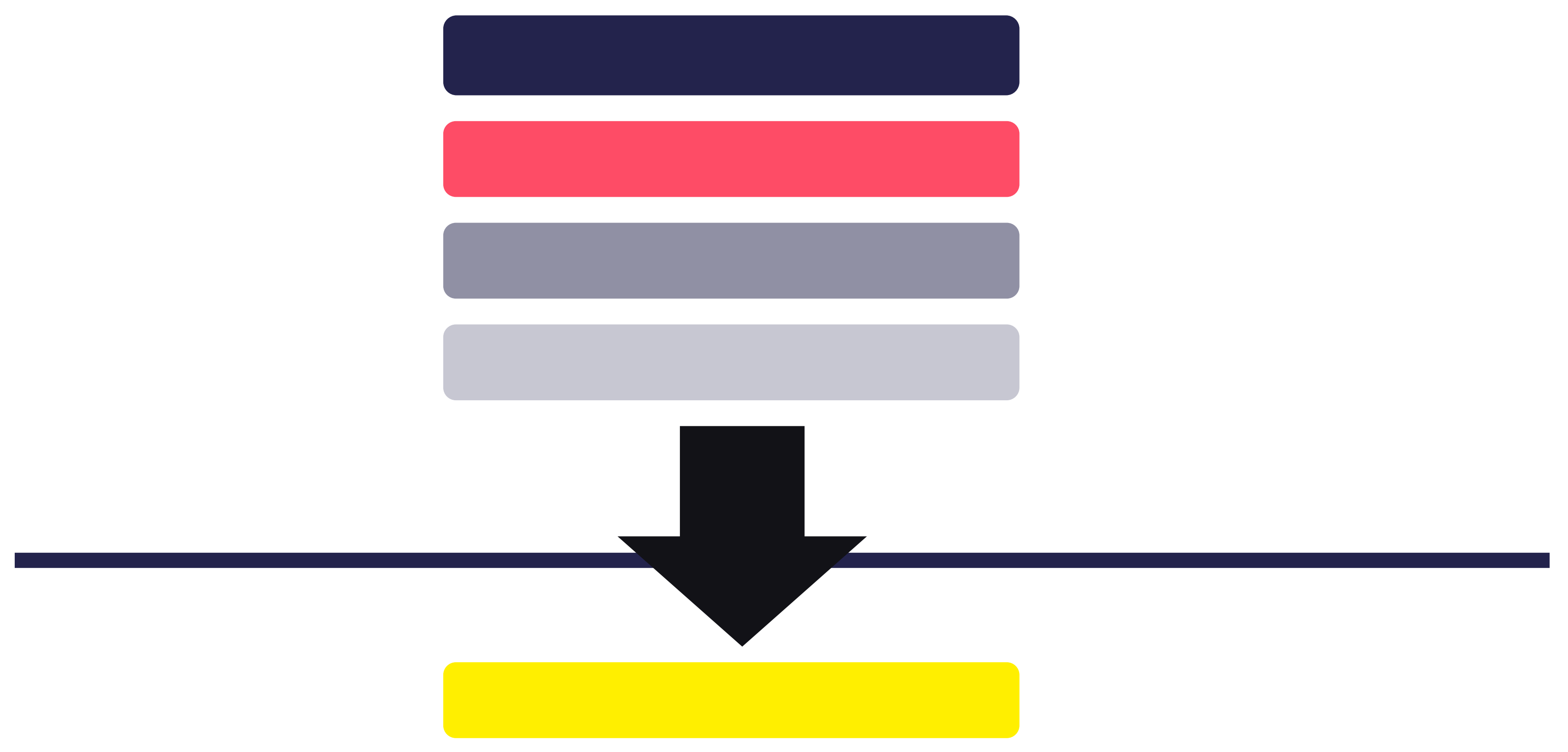

State
Business Logic
Routing
Rendering Logic
HTML
Server
Client
Look & Presentation Logic


State
Business Logic
Routing
Rendering Logic
HTML
JSON/
HTML
Server
Client
Look & Presentation Logic
Rendering Logic
Look & Presentation Logic

Let's use the **technologies from SPAs**,
but keep the **architecture of the Web.**

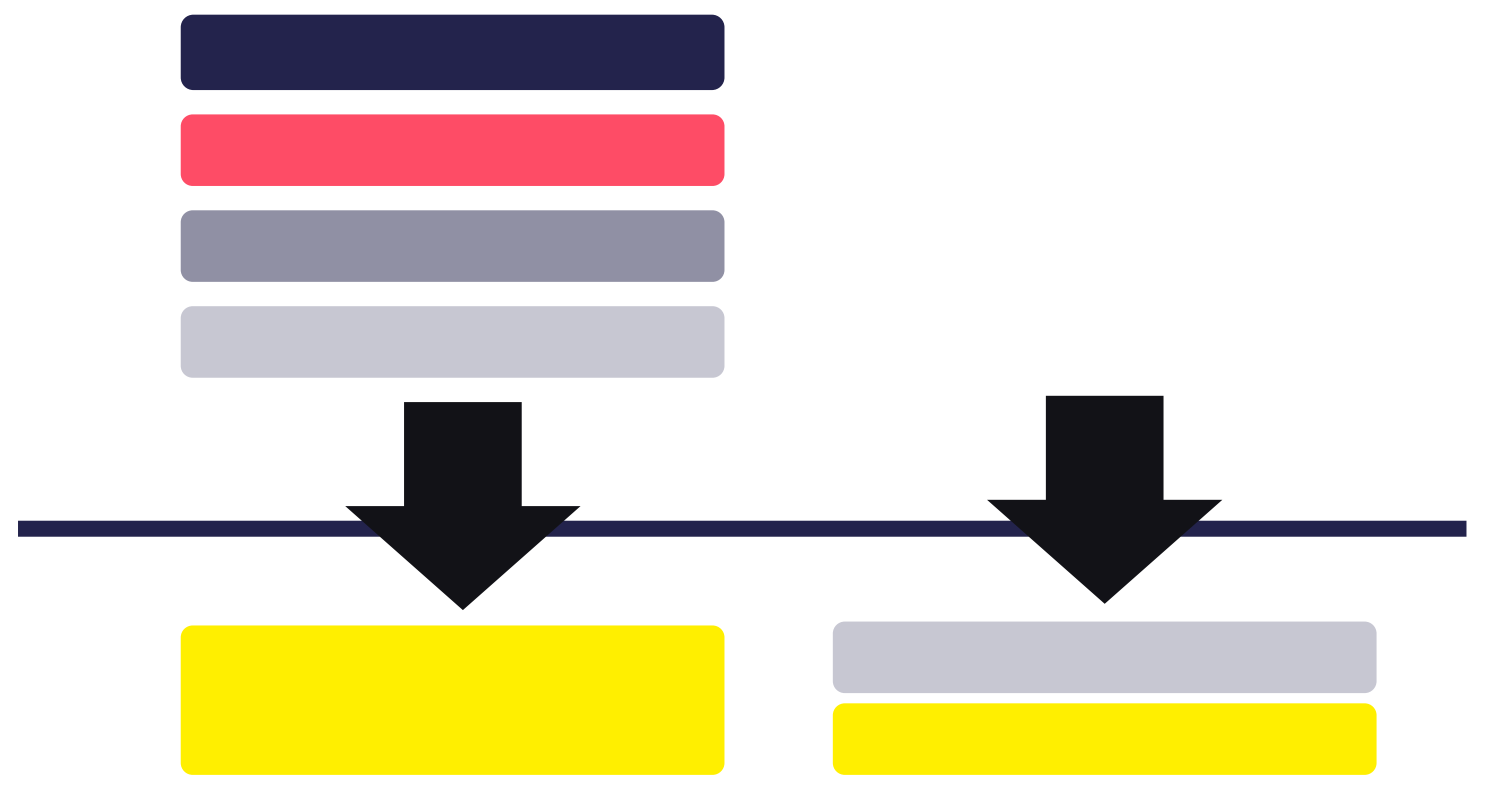
- Large number of users
- Basic UX needs
- Support for past, present and future devices

Pure SSR



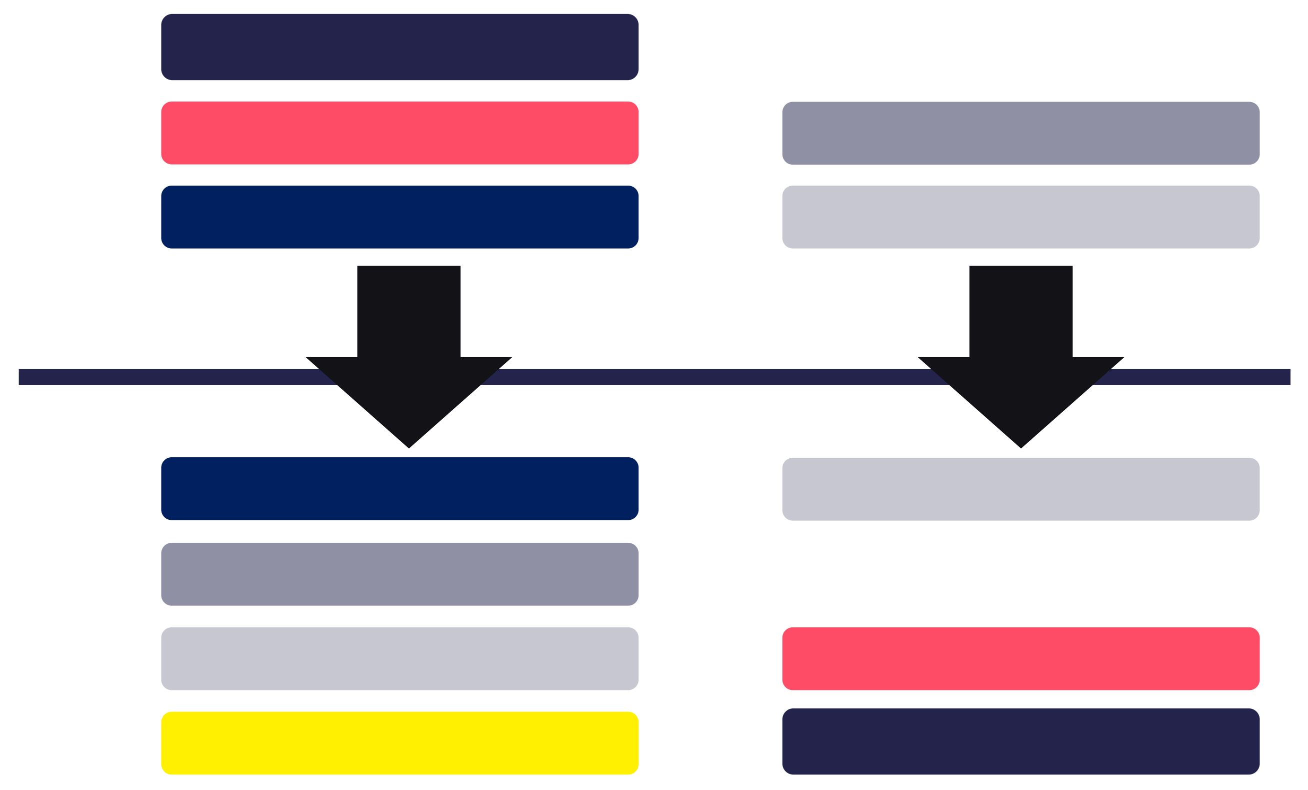
- Like SSR, but with
  - more UX needs
  - Complex component state
  - Basic offline support

SSR+RC



- Complex global client state
- Offline support
- Controlled device landscape

Pure SPA



Thanks! Questions?

INNOQ
www.innoq.com

Stefan Tilkov
stefan.tilkov@innoq.com
+49 170 4712625
stilkov

Lucas Dohmen
lucas.dohmen@innoq.com
+49 151 75062496
moonbeamlabs

innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim am Rhein
Germany
+49 2173 3366-0

Ohlauer Str. 43
10999 Berlin
Germany
+49 2173 3366-0

Ludwigstr. 180E
63067 Offenbach
Germany
+49 2173 3366-0

Kreuzstr. 16
80331 München
Germany
+49 2173 3366-0

Hermannstrasse 13
20095 Hamburg
Germany
+49 2173 3366-0

innoQ Schweiz GmbH

Gewerbestr. 11
CH-6330 Cham
Switzerland
+41 41 743 0116