

CAUTION

OUT-OF-ORDER STREAMING

THE FUTURE OF WEB DEVELOPMENT

842368

JULIAN BURR @ UXDX ONLINE 2025

BIGGEST UX REVOLUTION SINCE JAVASCRIPT



suspense

/sə'spens/

noun — a state or feeling of excited or anxious uncertainty about what may happen.

Example: "come on, John, don't keep me in suspense!"

SERVER

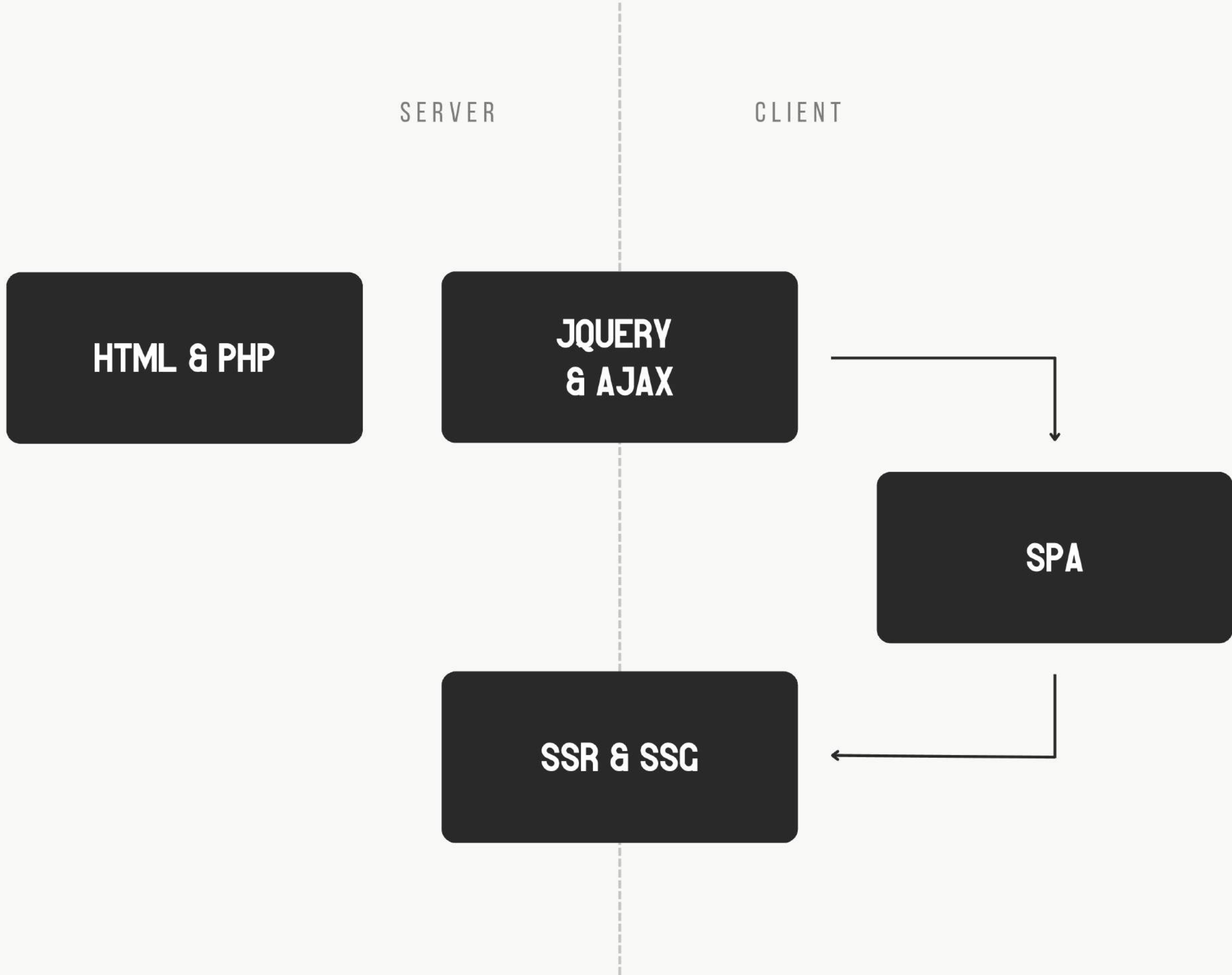
CLIENT

HTML & PHP

JQUERY
& AJAX

SPA

SSR & SSG



SERVER

CLIENT

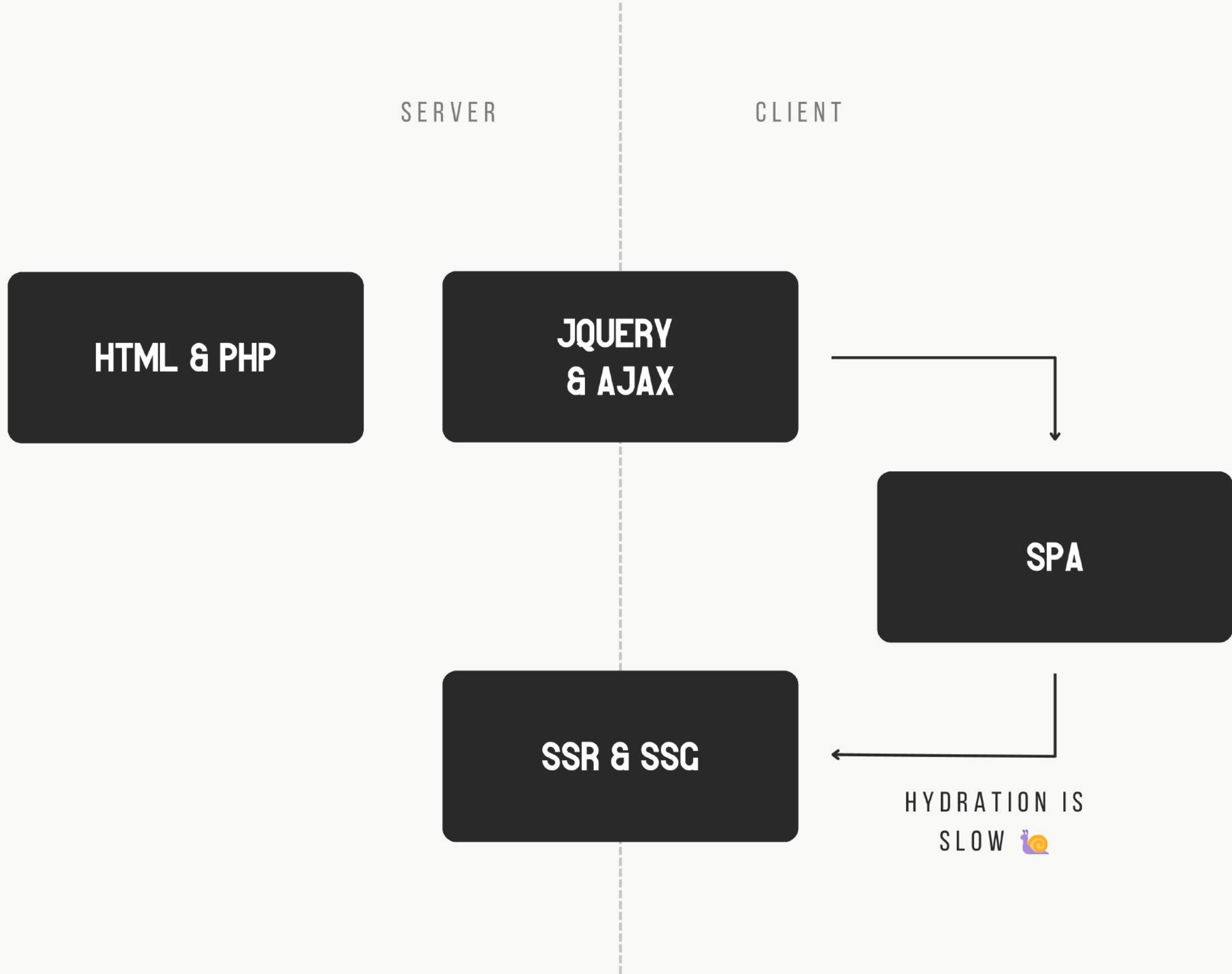
HTML & PHP

JQUERY
& AJAX

SPA

SSR & SSG

HYDRATION IS
SLOW 🐌



SERVER

CLIENT

HTML & PHP

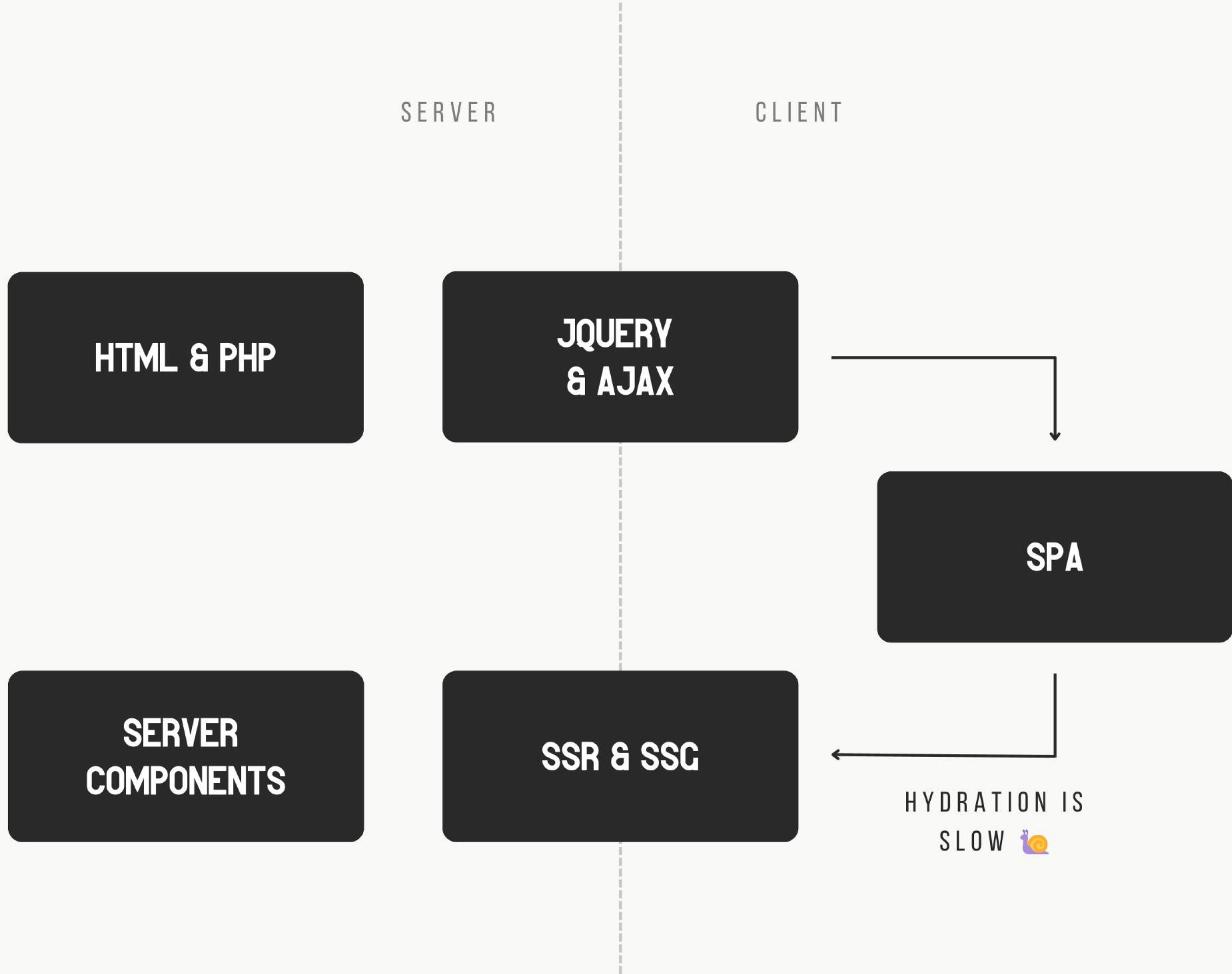
JQUERY
& AJAX

SPA

SERVER
COMPONENTS

SSR & SSG

HYDRATION IS
SLOW 🐌



WHAT MAKES SERVER COMPONENTS POSSIBLE?

WHAT MAKES SERVER COMPONENTS POSSIBLE?



**SERVER-SIDE
RENDERING**

WHAT MAKES SERVER COMPONENTS POSSIBLE?



**SERVER-SIDE
RENDERING**



**INTRODUCE
STREAMING**

WHAT MAKES SERVER COMPONENTS POSSIBLE?



**SERVER-SIDE
RENDERING**



**INTRODUCE
STREAMING**



**OUT-OF-ORDER
STREAMING**

WHAT MAKES SERVER COMPONENTS POSSIBLE?

**LET'S WRITE
SOME CODE**

```

import express from "express";
import { renderToString } from "react-dom/server";

const app = express();
app.use(express.static("out"));

app.get<{}, any, any, { id: string }>("/", async (req, res) => {
  const app = (
    <>
      <Header />
      <Title id={req.query.id} />
      <Details id={req.query.id} />
      <Similar id={req.query.id} />
    </>
  );

  let content = `<!doctype html><html><body>`;

  const contents = await Promise.all(
    app.props.children.map(async (child: JSX.Element) => {
      const element = await child.type(child.props);
      return renderToString(element);
    })
  );

  content += contents.join("");
  content += `</body></html>`;

  res.setHeader("Content-Type", "text/html");
  res.send(content);
});

app.listen(process.env.PORT || 3000);

```

YOU CAN FIND ALL CODE EXAMPLES AT
[HTTPS://GITHUB.COM/JULIANBURR/
 TALK-OUT-OF-ORDER-STREAMING](https://github.com/julianburr/talk-out-of-order-streaming)



```

import { fetchTitle } from "../api/title.ts";

export async function Title({ id }: { id: string }) {
  const data = await fetchTitle({ id });
  return (
    <section>
      <img src={data.image} />
      <div>
        <h2>{data.name}</h2>
        <p>
          {data.datePublished.substring(0, 4)} · {data.genre.join(", ")}
        </p>
        <p>{data.description}</p>
      </div>
    </section>
  );
}

```

```

import { fetchDetails } from "../api/details.ts";

export async function Details({ id }: { id: string }) {
  const data = await fetchDetails({ id });
  return (
    <section>
      <h3>Cast</h3>
      <div>
        {data.cast.edges.map((edge: any) => (
          <div key={edge.node?.name?.id}>
            <img
              src={edge.node?.name?.primaryImage?.url}
              alt={edge.node?.name?.nameText?.text}
            />
            <div>
              <p>{edge.node?.name?.nameText?.text}</p>

```

YOU CAN FIND ALL CODE EXAMPLES AT
[HTTPS://GITHUB.COM/JULIANBURR/
 TALK-OUT-OF-ORDER-STREAMING](https://github.com/julianburr/talk-out-of-order-streaming)





```
// Introduce streaming

res.setHeader("Content-Type", "text/html");
res.write(`<!doctype html><html><body>`);

for (let child of app.props.children) {
  const element = await child.type(child.props);
  res.write(renderToString(element));
}

res.write(`</body></html>`);
res.end();
```

YOU CAN FIND ALL CODE EXAMPLES AT
[HTTPS://GITHUB.COM/JULIANBURR/
TALK-OUT-OF-ORDER-STREAMING](https://github.com/julianburr/talk-out-of-order-streaming)





```
// Introduce suspense
```

```
const app = (  
  <>  
    <Header />  
    <Suspense fallback={<TitleSkeleton />}>  
      <Title id={req.query.id} />  
      <Details id={req.query.id} />  
      <Similar id={req.query.id} />  
    </Suspense>  
  </>  
>);
```



```
export let suspended: any = {};  
  
let uuid = 0;  
export function Suspense({ fallback, children }: any) {  
  const id = uuid++;  
  suspended[id] = children;  
  
  return (  
    <div data-suspense-id={id} style={{ display: 'contents' }}>  
      {fallback}  
    </div>  
  );  
}
```

YOU CAN FIND ALL CODE EXAMPLES AT

[HTTPS://GITHUB.COM/JULIANBURR/
TALK-OUT-OF-ORDER-STREAMING](https://github.com/julianburr/talk-out-of-order-streaming)



```

for (let child of app.props.children) {
  const element = await child.type(child.props);
  res.write(renderToString(element));
}

// Actually rendering suspended components

if (Object.keys(suspended).length) {
  res.write(`
    <script>
      // ...
    </script>
  `);

  await Promise.all(
    Object.entries(suspended).map(async ([key, content]: [any, any]) => {
      const children = Array.isArray(content) ? content : [content];
      const contents = await Promise.all(
        children.map(async (child: any) => {
          const element = await child.type(child.props);
          return renderToString(element);
        })
      );
      res.write(`
        <template>${contents.join("")}</template>
        <suspense-content target-id="${key}"></suspense-content>
      `);
      delete suspended[key];
    })
  );
}

```

YOU CAN FIND ALL CODE EXAMPLES AT
[HTTPS://GITHUB.COM/JULIANBURR/
 TALK-OUT-OF-ORDER-STREAMING](https://github.com/julianburr/talk-out-of-order-streaming)





// Insert the "magic" bit of Javascript that makes out-of-order-streaming work 🚀

```
res.write(`
  <script>
    window.customElements.define('suspense-content',
      class SuspenseContent extends HTMLElement {
        connectedCallback () {
          const content = this.previousElementSibling.content;
          const id = this.getAttribute('target-id');
          const target = document.querySelector('[data-suspense-id="' + id + '"]');

          target.innerHTML = '';
          while (content.firstChild) {
            target.appendChild(content.firstChild);
          }
        }
      }
    );
  </script>
`);
```

YOU CAN FIND ALL CODE EXAMPLES AT
[HTTPS://GITHUB.COM/JULIANBURR/
TALK-OUT-OF-ORDER-STREAMING](https://github.com/julianburr/talk-out-of-order-streaming)



GOOD RESOURCES TO GET STARTED

[HTTPS://REACT.DEV/REFERENCE/REACT/SUSPENSE](https://react.dev/reference/react/suspense)

OFFICIAL DOCS — SUSPENSE (ON THE CLIENT)

**[HTTPS://GITHUB.COM/REACTJS/RFCS/BLOB/MAIN/
TEXT/0213-SUSPENSE-IN-REACT-18.MD](https://github.com/reactjs/rfcs/blob/main/text/0213-suspense-in-react-18.md)**

RFC DOCUMENT FOR SUSPENSE ON THE SERVER

[HTTPS://WWW.YOUTUBE.COM/WATCH?V=MAEBEQHZR84](https://www.youtube.com/watch?v=MAEBEQHZR84)

BEN HOLMES — REACT SERVER COMPONENTS FROM SCRATCH

**[HTTPS://GAL.HAGEVER.COM/POSTS/
OUT-OF-ORDER-STREAMING-FROM-SCRATCH](https://gal.hagever.com/posts/out-of-order-streaming-from-scratch)**

GAL SCHLEZINGER — OUT OF ORDER STREAMING FROM SCRATCH

THANK YOU!



[HTTPS://GITHUB.COM/JULIANBURR/
TALK-OUT-OF-ORDER-STREAMING](https://github.com/julianburr/talk-out-of-order-streaming)

[HTTPS://WWW.JULIANBURR.DE/
UXDX-2025-SLIDES.PDF](https://www.julianburr.de/uxdx-2025-slides.pdf)

[HTTPS://BSKY.APP/PROFILE/JULIANBURR.DE](https://bsky.app/profile/julianburr.de)

[HTTPS://LINKEDIN.COM/IN/JULIANBURR](https://linkedin.com/in/julianburr)

[HTTPS://GITHUB.COM/JULIANBURR](https://github.com/julianburr)

