

LET'S BUILD SUSPENSE

JULIAN BURR @ REACT ADVANCED LONDON 2024



BIGGEST UX REVOLUTION SINCE JAVASCRIPT



suspense

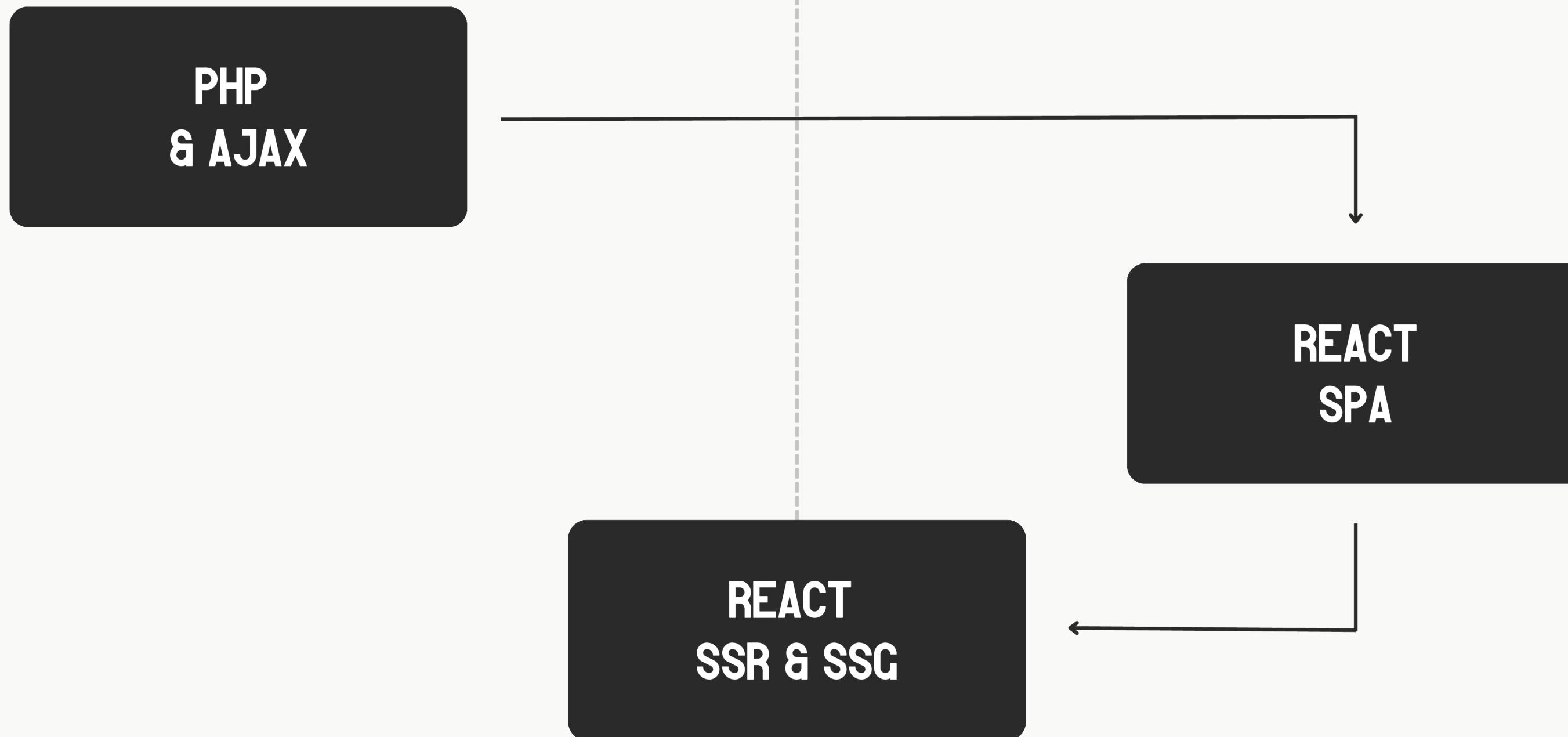
/sə'spɛns/

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

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

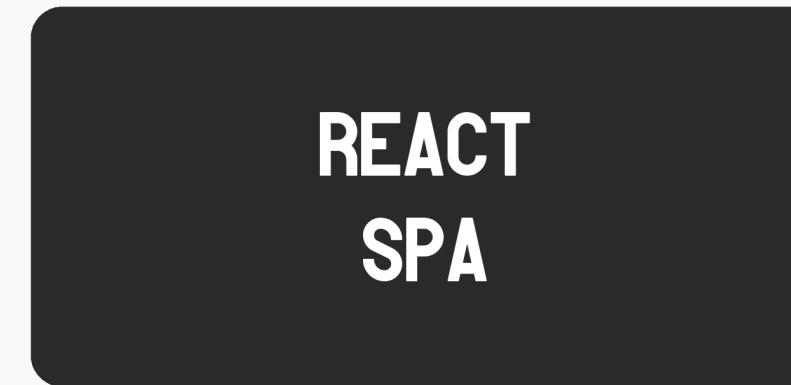
SERVER

CLIENT



SERVER

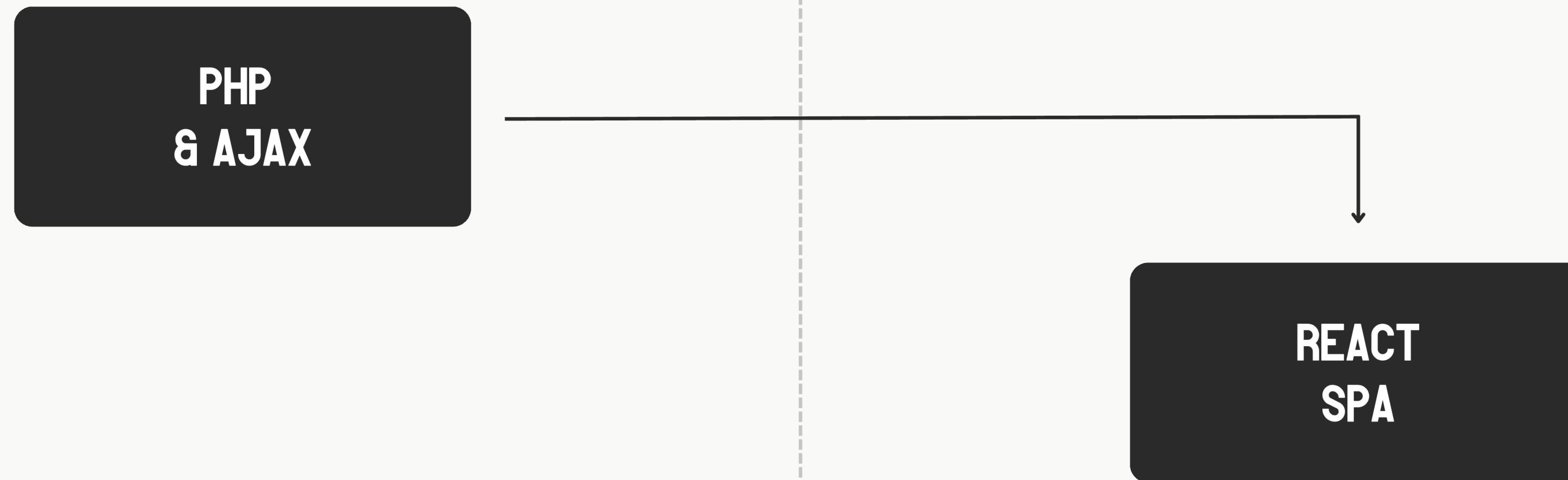
CLIENT



HYDRATION IS
SLOW 🐌

SERVER

CLIENT



REACT SERVER
COMPONENTS

REACT
SSR & SSC

HYDRATION IS
SLOW 🐌

WHY SUSPENSE IS GREAT

WHY SUSPENSE IS GREAT



**SERVER-SIDE
RENDERING**

WHY SUSPENSE IS GREAT



**SERVER-SIDE
RENDERING**



**INTRODUCE
STREAMING**

WHY SUSPENSE IS GREAT



**SERVER-SIDE
RENDERING**



**INTRODUCE
STREAMING**



**OUT-OF-ORDER
STREAMING**

BUILDING SUSPENSE

**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-LETS-BUILD-SUSPENSE](https://github.com/julianburr/talk-lets-build-suspense)



```
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>
            </div>
        ))
      </div>
    </section>
  );
}
```

YOU CAN FIND ALL CODE EXAMPLES AT
[HTTPS://GITHUB.COM/JULIANBURR/TALK-LETS-BUILD-SUSPENSE](https://github.com/julianburr/talk-lets-build-suspense)





```
// 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-LETS-BUILD-SUSPENSE](https://github.com/julianburr/talk-lets-build-suspense)



```
// 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-LETS-BUILD-SUSPENSE](https://github.com/julianburr/talk-lets-build-suspense)



```
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-LETS-BUILD-SUSPENSE](https://github.com/julianburr/talk-lets-build-suspense)





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

res.write(`
`);
```

YOU CAN FIND ALL CODE EXAMPLES AT
[HTTPS://GITHUB.COM/JULIANBURR/TALK-LETS-BUILD-SUSPENSE](https://github.com/julianburr/talk-lets-build-suspense)



GOOD RESOURCES TO GET STARTED

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

RFC DOCUMENT FOR SUSPENSE ON THE SERVER

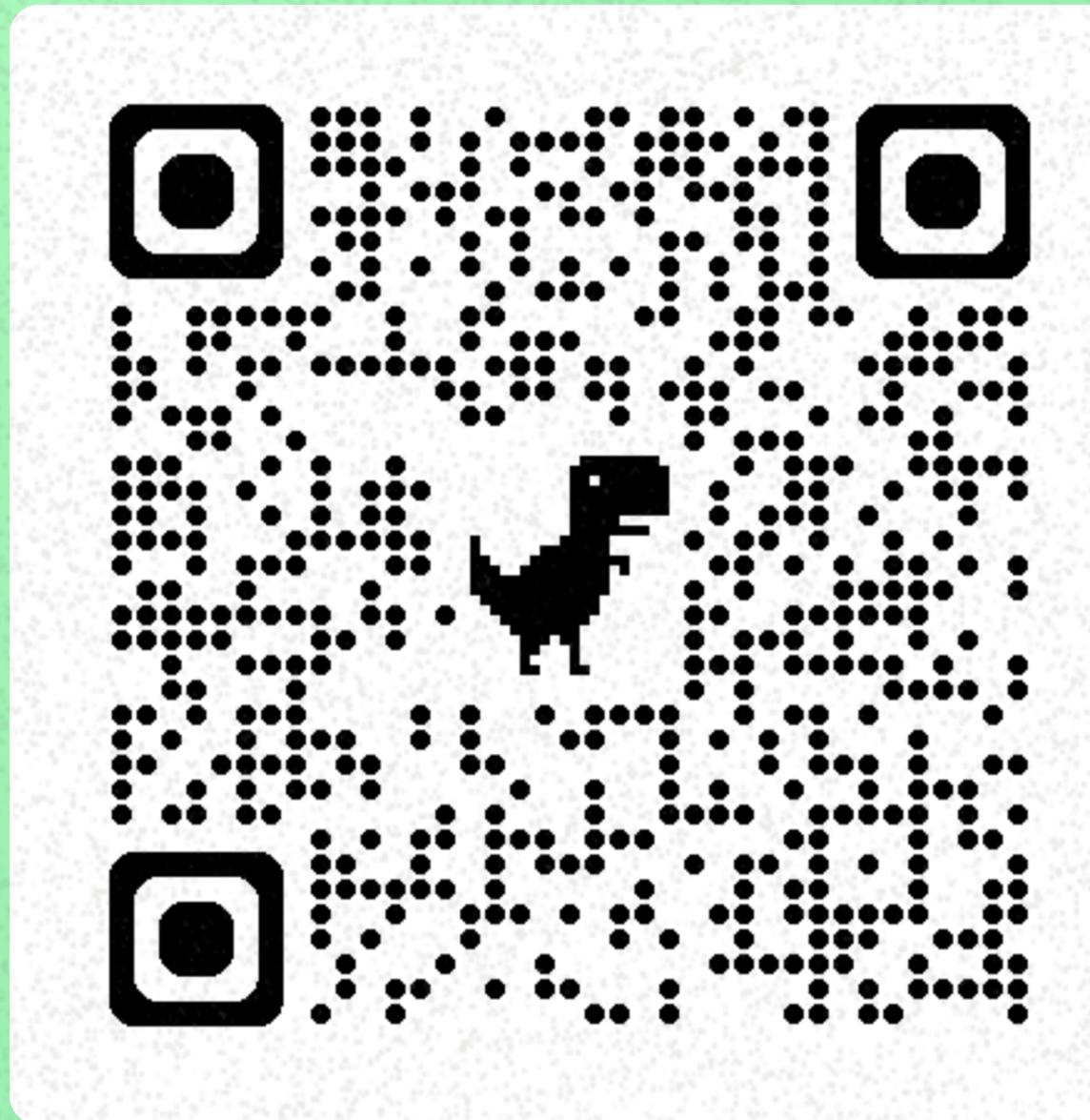
HTTPS://WWW.YOUTUBE.COM/WATCH?V=MAEBEQHZR84

BEN HOLMES — REACT SERVER COMPONENTS FROM SCRATCH

HTTPS://GAL.HAGEVER.COM/POSTS/OUT-OF-ORDER-STREAMING-FROM-SCRATCH

GAL SCHLEZINGER — OUT OF ORDER STREAMING FROM SCRATCH

THANK YOU!



[HTTPS://GITHUB.COM/JULIANBURR/
TALK-LETS-BUILD-SUSPENSE](https://github.com/julianburr/talk-lets-build-suspense)

[HTTPS://WWW.JULIANBURR.DE/
REACT-ADVANCED-LONDON-2024-SLIDES.PDF](https://www.julianburr.de/react-advanced-london-2024-slides.pdf)

[HTTPS://TWITTER.COM/JBURR90](https://twitter.com/jburr90)

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

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



REACT ADVANCED
LONDON