Interaktive Multi-Page-Applications mit Astro

inovex Meetup
Karlsruhe
2024-07-25
Julian Beck

Team inovex
Karlsruhe · Köln · München · Hamburg
Berlin · Stuttgart · Pforzheim · Erlangen

inovex

Julian Beck

Julian Beck

@julianfbeck

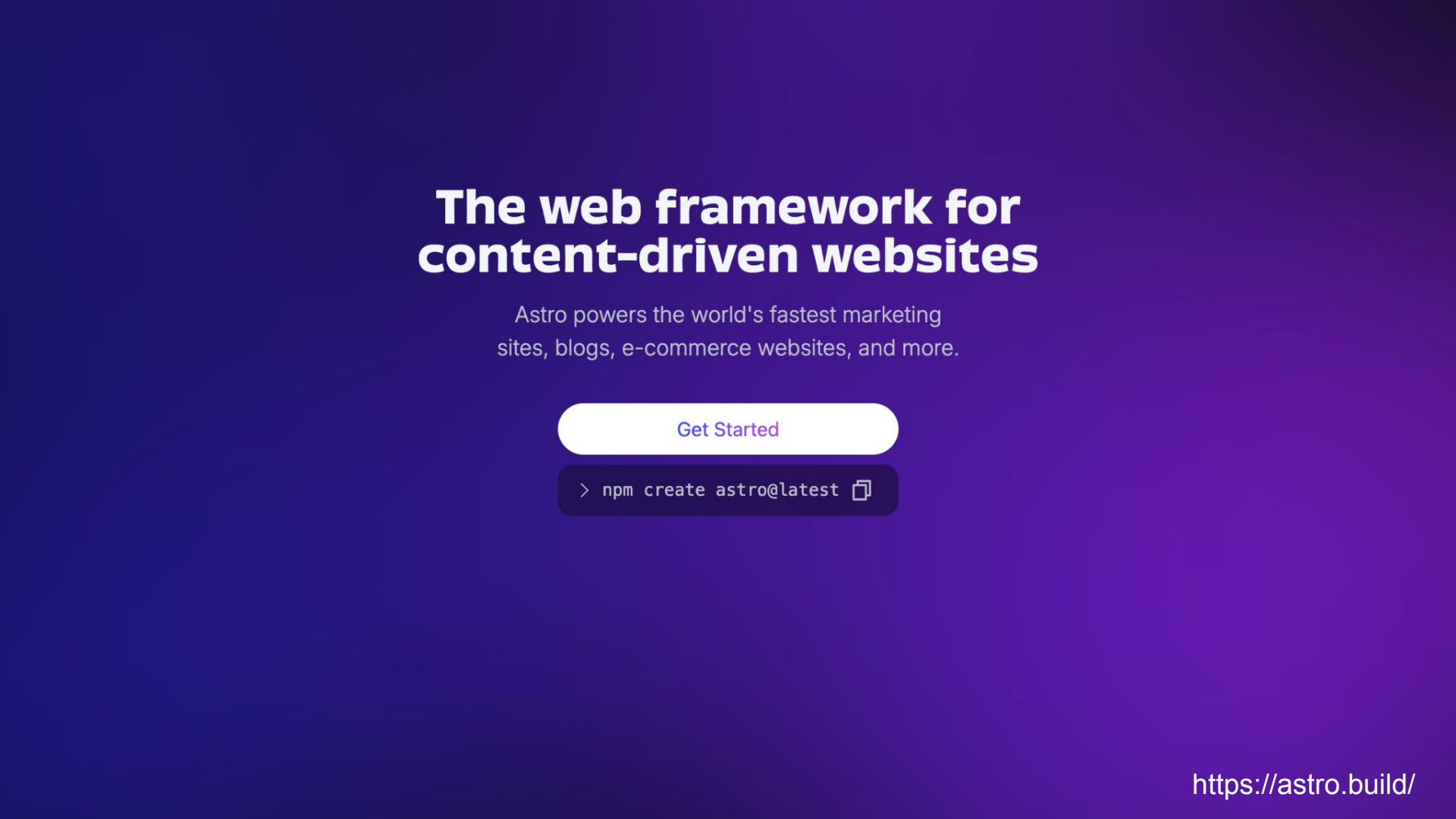
@julianfbeck

juli.sh

Cloud Platform Engineer bei inovex

- Sehr viel Yaml
- Sehr viel Cloud
- Sehr viel Kubernetes

inovex


The web framework for content-driven websites
Astro powers the world's fastest marketing sites, blogs, e-commerce websites, and more.
Get Started
> npm create astro@latest
https://astro.build/

Server-First
Astro improves website performance by rendering components on the server, sending lightweight HTML to the browser with zero unneccessary JavaScript overhead.

Content-Driven
Astro was designed to work with your content, no matter where it lives. Load data from your file system, external API, or your favorite CMS.

Customizable
Extend Astro with your favorite tools. Bring your own JavaScript UI components, CSS libraries, themes, integrations, and more.

https://astro.build/

Kurzer Einblick in eine Astro Code Base

Create and share polls. Welcome to OpenPoll

OpenPoll is a free and open-source platform to create and share polls. Use moderation features to share your polls during a live event or meeting.

Github Repo | Create Poll

https://openpoll.app/

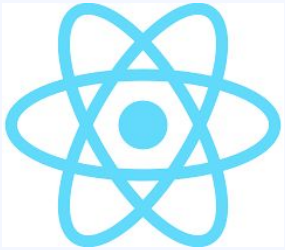
inovex

Was brauchen wir?

- Interaktive Client Komponenten
- APIs
- Datenbank ORM
- Auth
- Live Updates

inovex

Wir machen Astro Interaktiv!

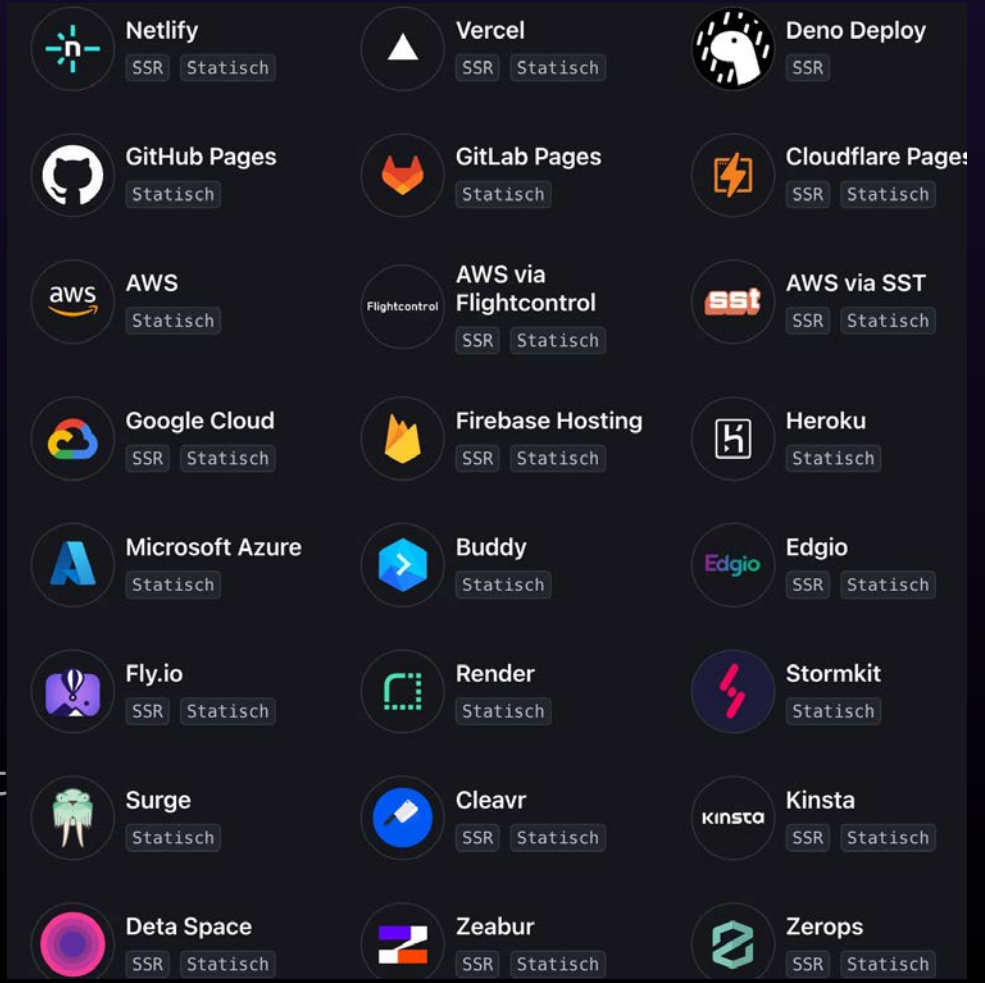


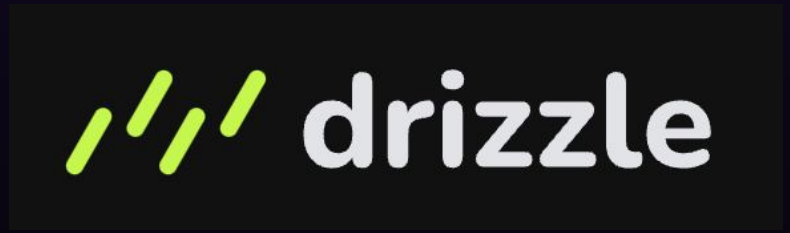
tRPC

inovex

SSR aktivieren

```
// astro.config.mjs
export default defineConfig({
  site: 'https://openpoll.app',
  output: 'server',
  integrations: [
    tailwind({
      applyBaseStyles: false
    }),
    auth(),
    react(),
  ],
  adapter: node({
    mode: 'standalone'
  }),                          // Stat
```



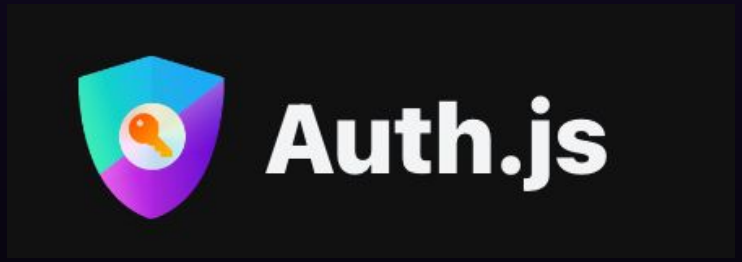


inovex

```ts
// models/schema.ts
export const polls = sqliteTable(
  'polls',
  {
    id: integer('id').primaryKey(),
    event: text('event').notNull(),
    question: text('question').notNull(),
    timestamp: text('timestamp')
      .default(sql`CURRENT_TIMESTAMP`)
      .notNull(),
    shortId: text('shortId')
      .notNull()
      .notNull()
      .$defaultFn(() => nanoid(10)),
    creatorId: text('creatorId')
      .notNull()
      .references(() => users.id, { onDelete: 'cascade' }),
  },
);

export type Poll = InferSelectModel<typeof polls> & {
  options: PollOption[];
};
```

```ts
const poll = await db.query.polls.findFirst({
  where: eq(polls.shortId, input),
  with: { options: true }
});
return poll;
```

inovex



inovex

GitHub OAuth

```
import { DrizzleAdapter } from '@auth/drizzle-adapter';

export default {
  adapter: DrizzleAdapter(db),
  providers: [
    GitHub({
      clientId: env.GITHUB_CLIENT_ID,
      clientSecret: env.GITHUB_CLIENT_SECRET
    })
  ],
  trustHost: true,
  secret: env.AUTH_SECRET,
  callbacks: {
    session({ session, user }) {
      if (session.user) {
        session.user.id = user.id;
      }
      return session;
    }
  }
} satisfies AuthConfig;
```

inovex

tRPC

End-to-end typesafe APIs made easy

inovex

Server

```ts
// src/pages/api/[trpc].ts

import { createContext } from '@/lib/trpc/context';
import { appRouter } from '@/lib/trpc/routers/_app';
import { fetchRequestHandler } from '@trpc/server/adapters/fetch';
import type { APIRoute } from 'astro';

export const ALL: APIRoute = ({ request }) => {
  return fetchRequestHandler({
    endpoint: '/api/trpc',
    req: request,
    router: appRouter,
    createContext
  });
};
```

```ts
// src/lib/trcp/routers/_app.ts
export const appRouter = router({
  poll: pollRouter,
  view: viewRouter,
  api: apiKeyRouter
});

export type AppRouter = typeof appRouter;
export const createCaller = t.createCallerFactory(appRouter);
```

Server

```
// src/lib/trpc/routers/poll.ts

export const pollRouter = router({

  /*
  create: rateLimitedAuthenticatedProcedure
  ...
  */
  get: publicProcedure.input(z.string()).query(async ({ input }) => {
    const poll = await db.query.polls.findFirst({
      where: eq(polls.shortId, input),
      with: { options: true }
    });
    return poll;
  }),

  ... more

})
```

inovex

Client

```tsx
// src/components/PollFormWrapper.tsx
const PollFormWrapper = () => {
  const [queryClient] = useState(() => new QueryClient());
  const [trpcClient] = useState(() =>
    trpcReact.createClient({
      links: [
        httpBatchLink({
          url: '/api/trpc'
        })
      ]
    })
  );

  return (
    <trpcReact.Provider client={trpcClient}
queryClient={queryClient}>
        <PollForm />
    </trpcReact.Provider>
  );
};
export default PollFormWrapper;
```

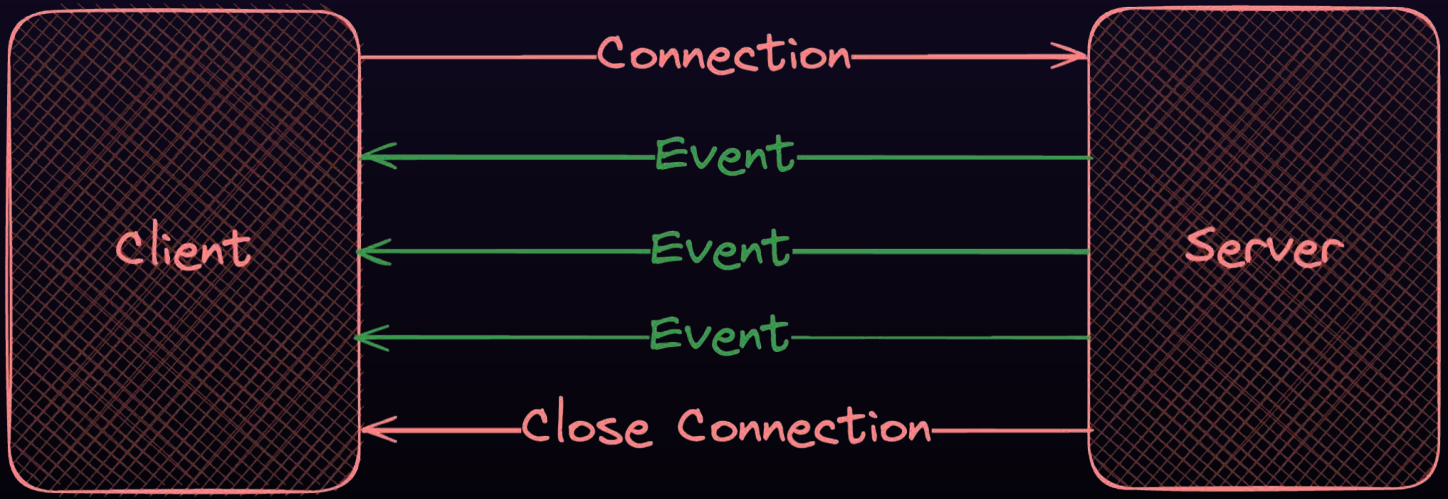
inovex

Client

```
export function PollForm({
  poll: poll,
}: {
  poll: Poll;
}) {
  const { data, refetch } = trpcReact.poll.get.useQuery(poll.shortId, {
    initialData: poll,
  });

  return(
    <>
      /*
        ... more
      */
    </>
  )
})
```

inovex

Server-Sent Events (SSE)



inovex

SSE Server

```
// pages/api/live/[id].ts
export const GET: APIRoute = async ({ request, params }) => {
  //...
  const customReadable = new ReadableStream({
    start(controller) {
      redisSubscriber.subscribe(`update:${id}`, (err) => {
        if (err) console.log(err);
      });
      redisSubscriber.on('message', (channel, message) => {
        if (channel === `update:${id}` && !isControllerClosed) {
          controller.enqueue(encoder.encode(`data: ${message}\n\n`));
        }
      });
    },
    // ...
  });
  // ...
  return new Response(customReadable, {
    headers: {
      Connection: 'keep-alive',
      'Content-Encoding': 'none',
      'Cache-Control': 'no-cache, no-transform',
      'Content-Type': 'text/event-stream; charset=utf-8'
    }
  });
};
```

inovex

SSE Client

```
const source = new EventSource(`/api/live/${poll.shortId}`);

 source.addEventListener('open', () => {
   console.log('SSE opened!');
 });

 source.addEventListener('message', (e) => {
   refetch(); -> TRPC query.poll.get(...)
 });
```

inovex

Litestream is an open-source, real-time streaming replication tool that lets you safely run SQLite applications on a single node.

inovex

Litestream

```
# Restore the database if it does not already exist.
if [ ! -f $DB_PATH ]; then
    echo "No database found, restoring from replica if exists"
    litestream restore -if-replica-exists $DB_PATH
else
    echo "Database already exists, skipping restore"
fi

litestream wal $DB_PATH
litestream generations $DB_PATH
# Run litestream with your app as the subprocess.
exec litestream replicate -exec "node ./dist/server/entry.mjs"
```

inovex

Enlightened library to convert HTML and CSS to SVG

inovex

Satori

```ts
import satori, { type SatoriOptions } from 'satori';
import { Resvg } from '@resvg/resvg-js';
import siteOgImage from './siteOgImage';

const options: SatoriOptions = {
  width: 1200,
  height: 630,
  embedFont: true,
};

function svgBufferToPngBuffer(svg: string) {
  const resvg = new Resvg(svg);
  const pngData = resvg.render();
  return pngData.asPng();
}

export async function generateOgImageForSite(poll: Poll) {
  const svg = await satori(siteOgImage({ data: poll }), options);
  return svgBufferToPngBuffer(svg);
}
```

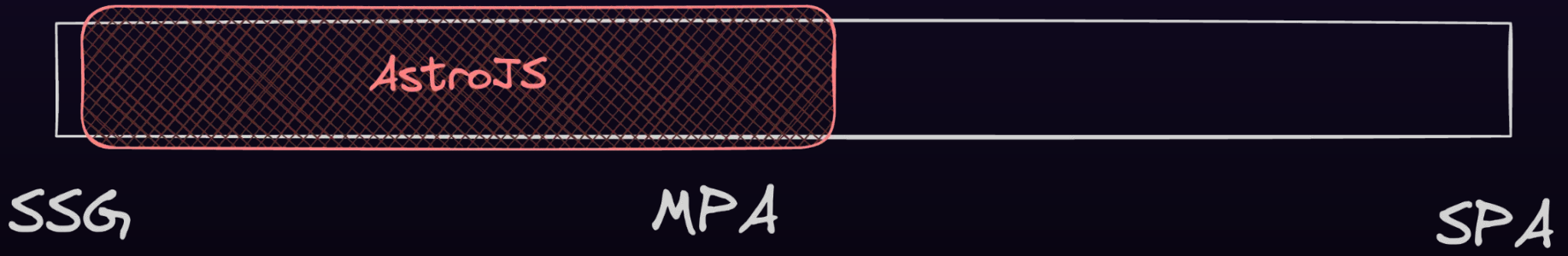
inovex

Satori

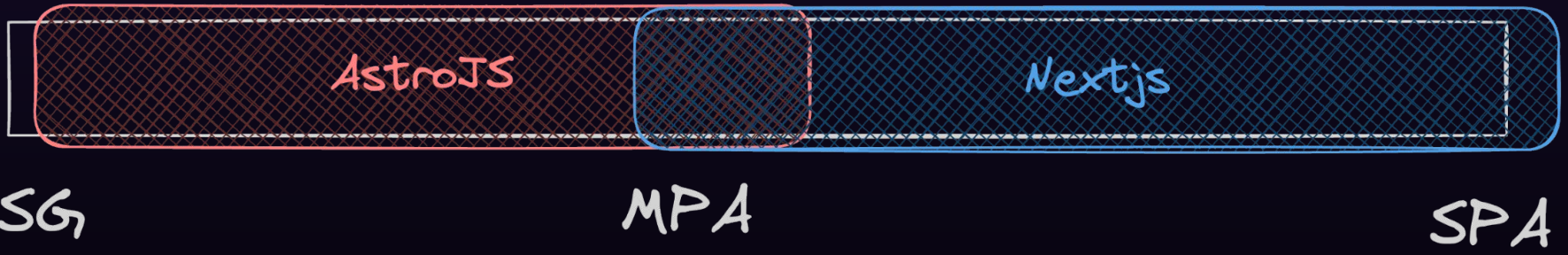
```
// src/pages/api/og/[id].png.ts
export const GET: APIRoute = async ({ params, request }) => {
  const id = params.id;
  if (!id) {
    return new Response('Not found', { status: 404 });
  }
  const poll = await db.query.polls.findFirst({
    where: eq(polls.shortId, id),
    with: { options: true }
  });

  if (!poll) {
    return new Response('Not found', { status: 404 });
  }
  return new Response(await generateOgImageForSite(poll), {
    headers: {
      'Content-Type': 'image/png',
      'Cache-Control': 'public, max-age=200, s-maxage=200'
    }
  });
};
```

inovex

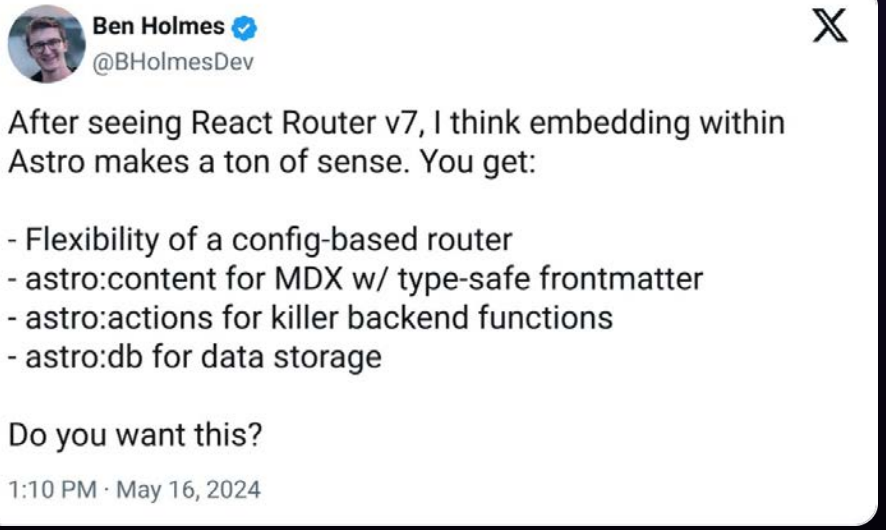
SSG vs SPA



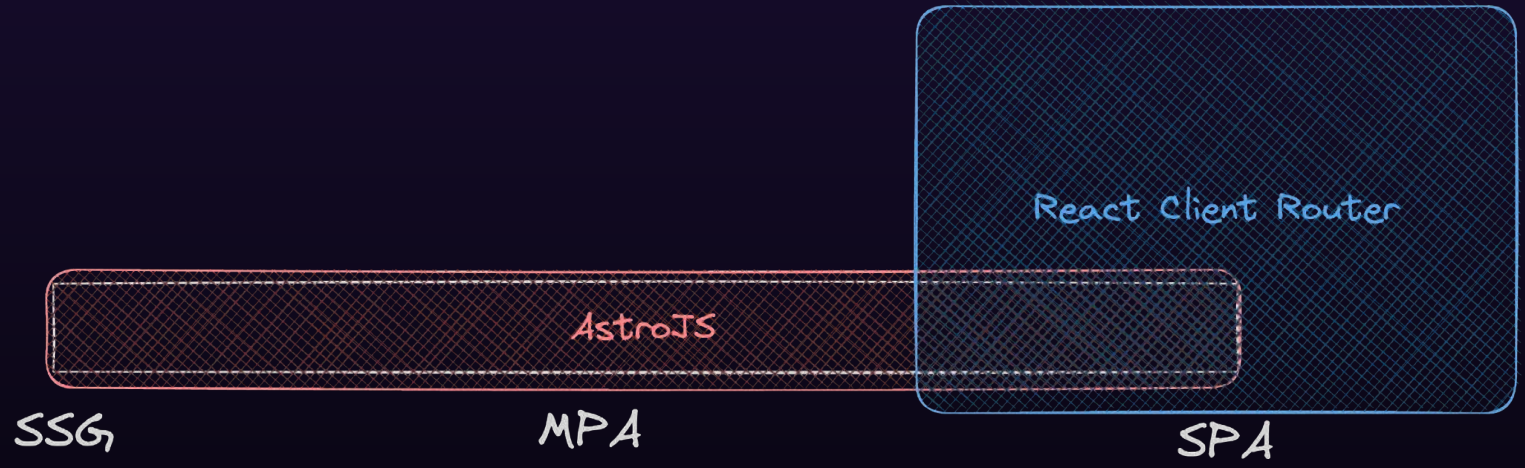
inovex



inovex


Ben Holmes
@BHolmesDev
After seeing React Router v7, I think embedding within Astro makes a ton of sense. You get:
- Flexibility of a config-based router
- astro:content for MDX w/ type-safe frontmatter
- astro:actions for killer backend functions
- astro:db for data storage
Do you want this?
1:10 PM · May 16, 2024

inovex



- /app/[...all] -> React SPA
- /blog/[slug] -> Content Collection für Blogs
- /docs/[slug] -> MDX Content Collection für Dokumentation

inovex

Ausblick

- Astro Actions
- Astro Server Islands

inovex

Vielen Dank!

Julian Beck

@julianfbeck

@julianfbeck

juli.sh

inovex