

Create React App erklärt



Coding-Assistent

Das ist ein hervorragender Schritt! Der Wechsel von **CRA** zu **Vite** ist wie der Umstieg von einem alten, schweren Kombi auf modernen Sportwagen: Alles fühlt sich sofort schneller an (vor allem der Start des Entwicklungsservers).

Hier ist deine Schritt-für-Schritt-Anleitung für die Migration.

1. Vorbereitung: Abhängigkeiten anpassen

Zuerst müssen wir die alten CRA-Skripte entfernen und die Vite-Pakete installieren.

Schritt A: CRA entfernen

Deinstalliere das Paket `react-scripts`:

Bash

```
npm uninstall react-scripts
```

Schritt B: Vite installieren

Installiere Vite und das React-Plugin für Vite:

Bash

```
npm install vite @vitejs/plugin-react --save-dev
```

2. Die `package.json` aktualisieren

Du musst die Skripte anpassen, damit sie nicht mehr auf `react-scripts`, sondern auf `vite` verweisen.

Vorher:

JSON

```
"scripts": {  
  "start": "react-scripts start",  
  "build": "react-scripts build",  
  "test": "react-scripts test"  
}
```

Nachher:

JSON

```
"scripts": {  
  "dev": "vite",  
  "build": "vite build",  
  "preview": "vite preview"  
}
```

3. Die `index.html` verschieben

In CRA lag die `index.html` im Ordner `public/`. Vite erwartet sie standardmäßig im **Hauptverzeichnis (Root)** deines Projekts.

1. Verschiebe die Datei `public/index.html` in den Hauptordner.
2. Entferne alle `%PUBLIC_URL%`-Präfixe aus der Datei (Vite löst Pfade anders auf).
 - Aus `<link rel="icon" href="%PUBLIC_URL%/favicon.ico" />` wird einfach `<link rel="icon" href="/favicon.ico" />`.
3. **Ganz wichtig:** Du musst den Einstiegspunkt für dein JavaScript händisch in der `index.html` verknüpfen. Füge diese kurz vor dem schließenden `</body>` ein:

HTML

```
<script type="module" src="/src/index.js"></script>
```

(Hinweis: Wenn deine Datei `index.tsx` heißt, passe den Pfad entsprechend an.)

4. Konfigurationsdatei erstellen

Erstelle im Hauptverzeichnis eine neue Datei namens `vite.config.js`. Dies ist das Herzstück deiner neuen Konfiguration.

JavaScript

```
import { defineConfig } from 'vite';
import react from '@vitejs/plugin-react';

export default defineConfig({
    // Hier binden wir das React-Plugin ein
    plugins: [react()],
    server: {
        // Optional: Port festlegen (CRA nutzt standardmäßig 3000)
        port: 3000,
        open: true, // Öffnet den Browser automatisch
    },
    build: {
        outDir: 'build', // Damit der Build-Ordner wie bei CRA weiterhin "build" heißt (Vite nutzt sonst "dist")
    }
});
```

5. Besonderheiten bei Umgebungsvariablen (Env)

Wenn du `.env`-Dateien nutzt, gibt es einen kleinen Unterschied:

- **CRA:** Nutzt `REACT_APP_VARIABLE_NAME` und Zugriff über `process.env`.
- **Vite:** Nutzt `VITE_VARIABLE_NAME` und Zugriff über `import.meta.env`.

Tipp: Du müsstest also deine Variablen in der `.env`-Datei umbenennen (z.B. von `REACT_APP_API_URL` zu `VITE_API_URL`).

Warum lohnt sich dieser Aufwand?

- **Geschwindigkeit:** Vite nutzt "Native ESM", was bedeutet, dass der Server fast instant startet, egal wie groß dein Projekt ist.
- **HMR (Hot Module Replacement):** Wenn du Code änderst, wird nur die betroffene Stelle im Browser aktualisiert – extrem schnell und zuverlässig.
- **Sauberkeit:** Keine versteckten Konfigurationen mehr. Alles ist transparent in der `vite.config.js`.

Nächste Schritte

Sobald du diese Änderungen gemacht hast, kannst du dein Projekt mit `npm run dev` starten.

Gibt es in deinem aktuellen Projekt viele Umgebungsvariablen oder nutzt du TypeScript? Dann kann ich dir helfen, die Konfiguration spezifisch darauf abzustimmen.