Client-side Fetching

Client-side data fetching is useful when your page doesn't require SEO indexing, when you don't need to pre-render your data, or when the content of your pages needs to update frequently. Unlike the server-side rendering APIs, you can use client-side data fetching at the component level.

If done at the page level, the data is fetched at runtime, and the content of the page is updated as the data changes. When used at the component level, the data is fetched at the time of the component mount, and the content of the component is updated as the data changes.

It's important to note that using client-side data fetching can affect the performance of your application and the load speed of your pages. This is because the data fetching is done at the time of the component or pages mount, and the data is not cached.

Client-side data fetching with useEffect

The following example shows how you can fetch data on the client side using the useEffect hook.

```
import { useState, useEffect } from 'react';

function Profile() {
  const [data, setData] = useState(null);
  const [isLoading, setLoading] = useState(false);

  useEffect(() => {
    setLoading(true);
    fetch('/api/profile-data')
      .then((res) => res.json())
      .then((data) => {
        setData(data);
        setLoading(false);
      });
  }, []);

  if (isLoading) return <p>Loading...</p>;
```

```
  if (!data) return <p>No profile data</p>;

  return (
    <div>
      <h1>{data.name}</h1>
      <p>{data.bio}</p>
    </div>
  );
}
```

Client-side data fetching with SWR

The team behind Next.js has created a React hook library for data fetching called **SWR**. It is **highly recommended** if you are fetching data on the client-side. It handles caching, revalidation, focus tracking, refetching on intervals, and more.

Using the same example as above, we can now use SWR to fetch the profile data. SWR will automatically cache the data for us and will revalidate the data if it becomes stale.

For more information on using SWR, check out the SWR docs.

```
import useSWR from 'swr';

const fetcher = (...args) => fetch(...args).then((res) => res.json());

function Profile() {
  const { data, error } = useSWR('/api/profile-data', fetcher);

  if (error) return <div>Failed to load</div>;
  if (!data) return <div>Loading...</div>;

  return (
    <div>
      <h1>{data.name}</h1>
      <p>{data.bio}</p>
    </div>
  );
}
```