Next.js - Pre-Rendering

In Next.js, we know it generates HTML for a page called pre-rendering. Next.JS supports two types of pre-rendering.

- **Static Generation** − This method generates the HTML page at build time. This pre-rendered HTML is sent on each request. This method is useful for marketing websites, blogs, e-commerce products listing wesites, helps, documentation websites.
- **Server Side Generation** − This method generates the HTML page on each request. This method is suitable when an html page contents can vary with each request.

Per Page Pre-rendering

Next.JS allows to set pre-rendering method for each page where most of pages follow static generation and other pages will use server side rendering.

Static Generation Without Data

Static generation can be done without data in which case, HTML pages will be ready without need to prefetch the data and then start rendering. Data can be fetched later or on request. This technique helps in showing user an User Interface without any data in case data takes time to come.

Static Generation With Data

Static generation can be done with data in which case, HTML pages will not be ready until data is fetched, as HTML may be dependent on data. Each component has a special method **getStaticProps** which can be used to fetch data and pass data as props of the page so that page can render accordings to passed props.

getStaticProps() function runs at build time in production and runs for every request in dev mode.

Let's create an example to demonstrate the same.

In this example, we'll create a update index.js and first.js page to make a server hit to get data.

Let's update the nextjs project used in Global CSS Support chapter.

Update index.js file in pages directory to use getServerSideProps() method. This method will be called per request.

```
import Link from 'next/link'
import Head from 'next/head'
```

```
function HomePage(props) {
   return (
      <>
         <Head>
            <title>Welcome to Next.js!</title>
         </Head>
         <div>Welcome to Next.js!</div>
         <Link href="/posts/first"><a>First Post</a></Link>
         <br/>
         <div>Next stars: {props.stars}</div>
         <img src="/logo.png" alt="TutorialsPoint Logo" />
      </>
   )
}

export async function getServerSideProps(context) {
   const res = await fetch('https://api.github.com/repos/vercel/next.js')
   const json = await res.json()
   return {
      props: { stars: json.stargazers_count }
   }
}

export default HomePage
```

Update first.js file in pages directory to use getStaticProps() method. This method will be called once.

```
import Link from 'next/link'
import Head from 'next/head'
import Container from '../../components/container'

export default function FirstPost(props) {
   return (
      <>
         <Container>
            <Head>
               <title>My First Post</title>
            </Head>
            <h1>My First Post</h1>
            <h2>
               <Link href="/">
                  <a>Home</a>
               </Link>
               <div>Next stars: {props.stars}</div>
            </h2>
```

```
        </Container>
      </>
    )
}

export async function getStaticProps() {
    const res = await fetch('https://api.github.com/repos/vercel/next.js')
    const json = await res.json()
    return {
      props: { stars: json.stargazers_count }
    }
}
```

Start Next.js Server

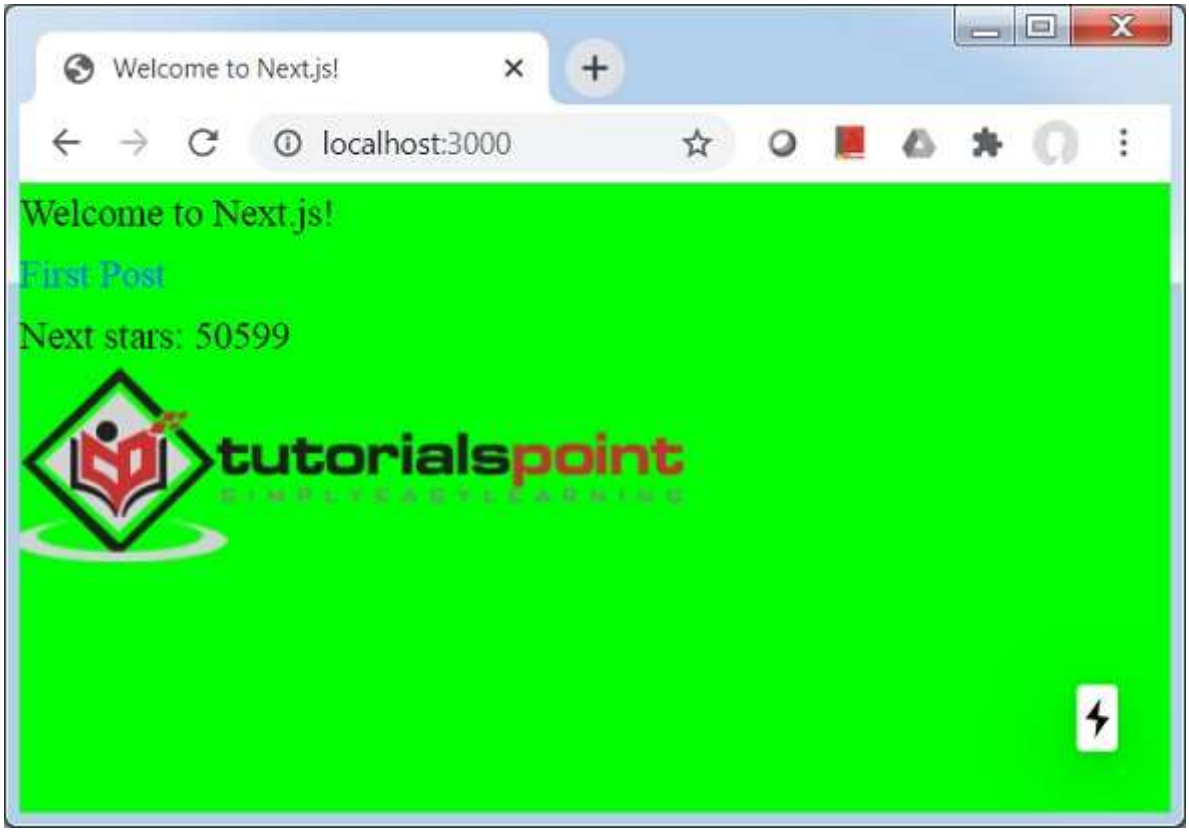
Run the following command to start the server −.

```
npm run dev
> nextjs@1.0.0 dev \Node\nextjs
> next

ready - started server on http://localhost:3000
event - compiled successfully
event - build page: /
wait  - compiling...
event - compiled successfully
event - build page: /next/dist/pages/_error
wait  - compiling...
event - compiled successfully
```

Verify Output

Open localhost:3000 in a browser and you will see the following output.



Click on First post link.