NEXT.JS

tutorialspoint

SIMPLY EASY LEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia https://twitter.com/tutorialspoint

About the Tutorial

The Next.js is React Based framework with server side rendering capability. It is very fast and Search engine optimisation (SEO) friendly. By using Next.js, you can create robust react based application quite easily and test them on the computer.

Audience

This tutorial is designed for software programmers, who want to learn the basics of Next.js and its concepts in a simple and easy manner. This tutorial will give you enough understanding on the various functionalities of Next.js with suitable examples.

Prerequisites

Before proceeding with this tutorial, you should have a basic understanding of the programming language JavaScript and web framework React.

Copyright & Disclaimer

Table of Contents

1. Next.js — Overview

The Next.js is a React Based framework with server side rendering capability. It is very fast and search engine optimisation (SEO) friendly.

By using Next.js, you can create robust react based application quite easily and test them.

KeyFeatures

Following are the key features of Next.js:

- **Hot Code Reload**: Next.js server detects modified files and reloads them automatically.
- **Automatic Routing**: No need to configure any Uniform Resource Locator (URL) for routing and files are to be placed in the pages folder. All the urls will be mapped to the file system. Customisation can be done.
- **Component specific styles**: styled-jsx provides support for global as well as component specific styles.
- **Server side rendering**: React components are prerendered on server. Hence, loads faster on client.
- **Node Ecosystem**: Next.js being react based, gels well with the Node ecosystem.
- **Automatic code split**: Next.js renders pages with libraries they need. Next.js instead of creating a single large javascript file, creates multiples resources. When a page is loaded, only required javascript page is loaded with it.
- **Prefetch**: Next.js provides link component, which is used to link multiple components which supports a prefetch property to prefetch page resources in the background.
- **Dynamic Components**: Next.js allows to import JavaScript modules and React components dynamically.
- **Export Static Site**: Next.js allows to export full static site from your web application.
- **Built-in Typescript Support**: Next.js is written in Typescripts and provides excellent Typescript support.

2. Next.js — Environment Setup

As Next.js is a react based framework, we are using Node environment. Now, make sure you have **Node.js** and **npm** installed on your system.

You can use the following command to install Next.js:

```
npm install next react react-dom
```

You will get the following output, once Next.js is successfully installed:

```
+ react@16.13.1
+ react-dom@16.13.1
+ next@9.4.4
added 831 packages from 323 contributors and audited 834 packages in 172.989s
```

Now, let's create a node package.json, by using the below mentioned command:

```
npm init
```

Select default values while creating a package.json as shown below:

```
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (nextjs)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
```

```
author:
license: (ISC)
About to write to \Node\nextjs\package.json:
{
   "name": "nextjs",
   "version": "1.0.0",
   "description": "",
   "main": "index.js",
   "dependencies": {
      "next": "^9.4.4",
      "react": "^16.13.1",
      "react-dom": "^16.13.1"
   },
   "devDependencies": {},
   "scripts": {
      "test": "echo \"Error: no test specified\" && exit 1"
   },
   "author": "",
   "license": "ISC"
}
Is this OK? (yes)
```

Now, update the scripts section of package.json to include Next.js commands as stated below:

```
{
   "name": "nextjs",
   "version": "1.0.0",
   "description": "",
   "main": "index.js",
   "dependencies": {
      "next": "^9.4.4",
      "react": "^16.13.1",
      "react-dom": "^16.13.1"
   },
   "devDependencies": {},
   "scripts": {
```

```
      "test": "echo \"Error: no test specified\" && exit 1",
      "dev": "next",
      "build": "next build",
      "start": "next start"
   },
   "author": "",
   "license": "ISC"
}
```

Create pages directory

Create a pages folder within next.js folder and create an index.js file with the following contents:

```
function HomePage() {
   return <div>Welcome to Next.js!</div>
}

export default HomePage
```

Start Next.js Server

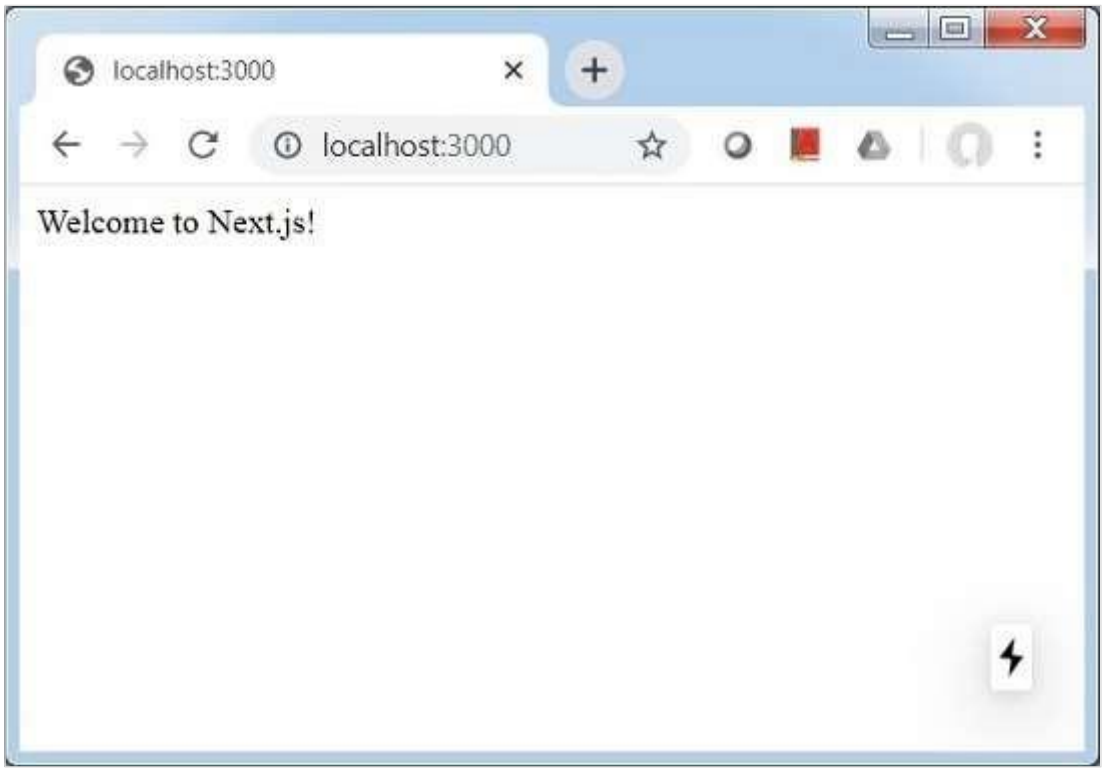
Run the following command to start the server:

```
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:



3. Next.js — Pages

In Next.js, we can create pages and navigate between them by using a file system routing feature. We will use **Link** component to have a client side navigation between pages.

Here, a page is a React Component and are exported from pages directory. Each page is associated with a route based on its file name.

For example:

- **pages/index.js** is linked with '/' route.
- **pages/posts/first.js** is linked with '/posts/first' route and so on.

Let's update the next.js project created in **Environment Setup** chapter, which is available at https://www.tutorialspoint.com/nextjs/nextjs_environment.htm.

Create post directory and first.js within it with the following contents:

```
export default function FirstPost() {
   return <h1>My First Post</h1>
}
```

Add Link Support to go back to Home page. Update first.js as follows:

```
import Link from 'next/link'

export default function FirstPost() {
   return (
      <>
         <h1>My First Post</h1>
         <h2>
            <Link href="/">
               <a>Home</a>
            </Link>
         </h2>
      </>
   )
}
```

Add Link Support to home page to navigate to first page. Update index.js as follows:

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

```
function HomePage() {
   return (
      <>
         <div>Welcome to Next.js!</div>
         <Link href="/posts/first"><a>First Post</a></Link>
      </>
   )
}

export default HomePage
```

Start Next.js Server

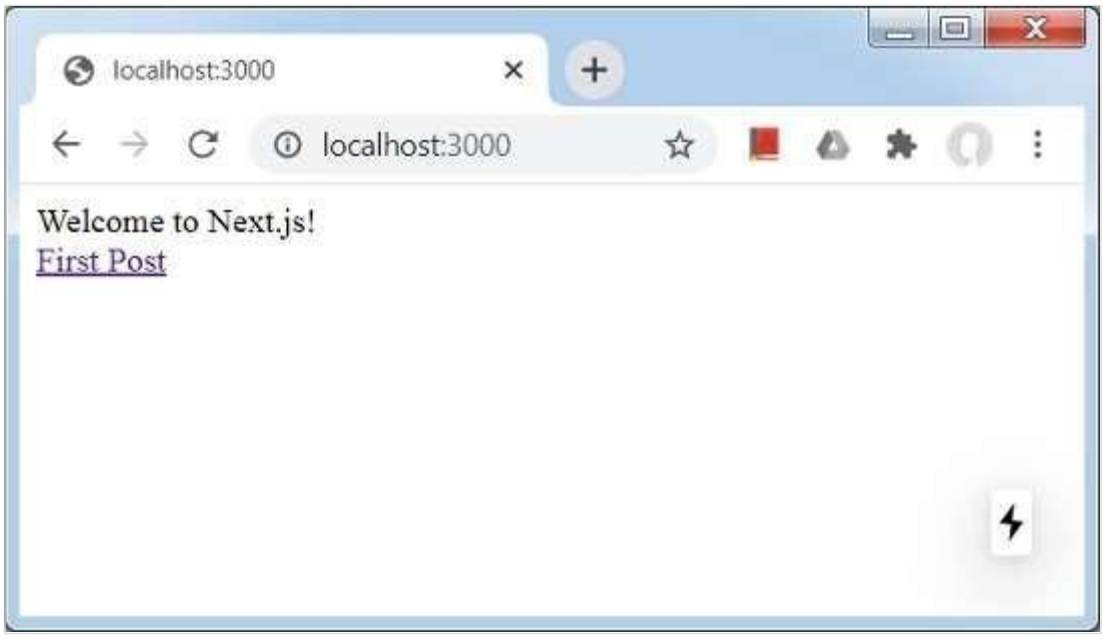
Run the following command to start the server:

```
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 Link and you will see the following output:

4. Next.js — Static File Serving

In Next.js, we can serve static pages like images, very easily by putting them in **public**, a top level directory. We can refer to these files in similar way like pages in **pages** directory.

In Next.js, a page is a React Component and is exported from pages directory. Each page is associated with a route based on its file name.

Let's update the nextjs project used in **Pages** chapter, which is available at https://www.tutorialspoint.com/nextjs/nextjs_pages.htm.

Create public directory and place any images within it. We have taken **logo.png**, **TutorialsPoint Logo** image.

Update first.js as follows. Here, we have added a reference to logo.png in index.js file.

```
import Link from 'next/link'

export default function FirstPost() {
   return (
      <>
         <h1>My First Post</h1>
         <h2>
            <Link href="/">
               <a>Home</a>
            </Link>
         </h2>
         <br/">
         <img src="/logo.png" alt="TutorialsPoint Logo" />
      </>
   )
}
```

Start Next.js Server

Run the following command to start the server:

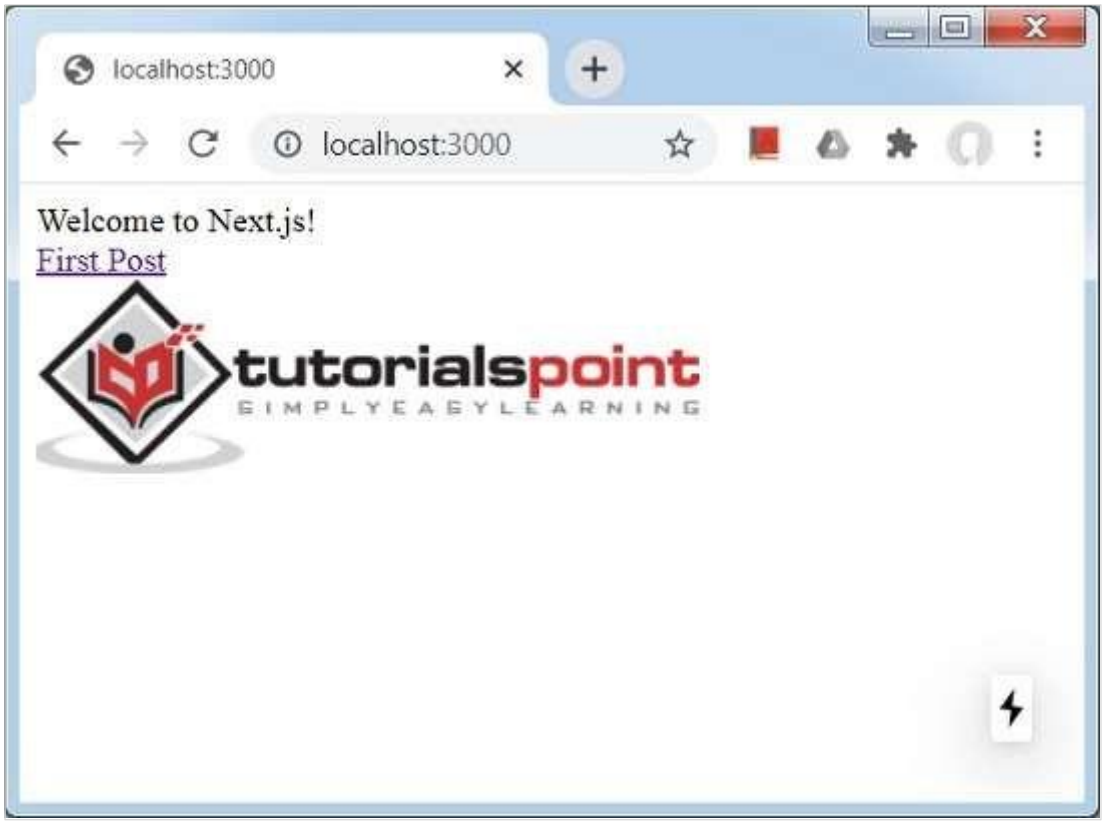
```
npm run dev
> nextjs@1.0.0 dev \Node\nextjs
> next
```

Next.js

```
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:



Public directory is also useful in case of SEO purpose. It can be used for robot.txt, Google Site verification or any other static assets in the web application.