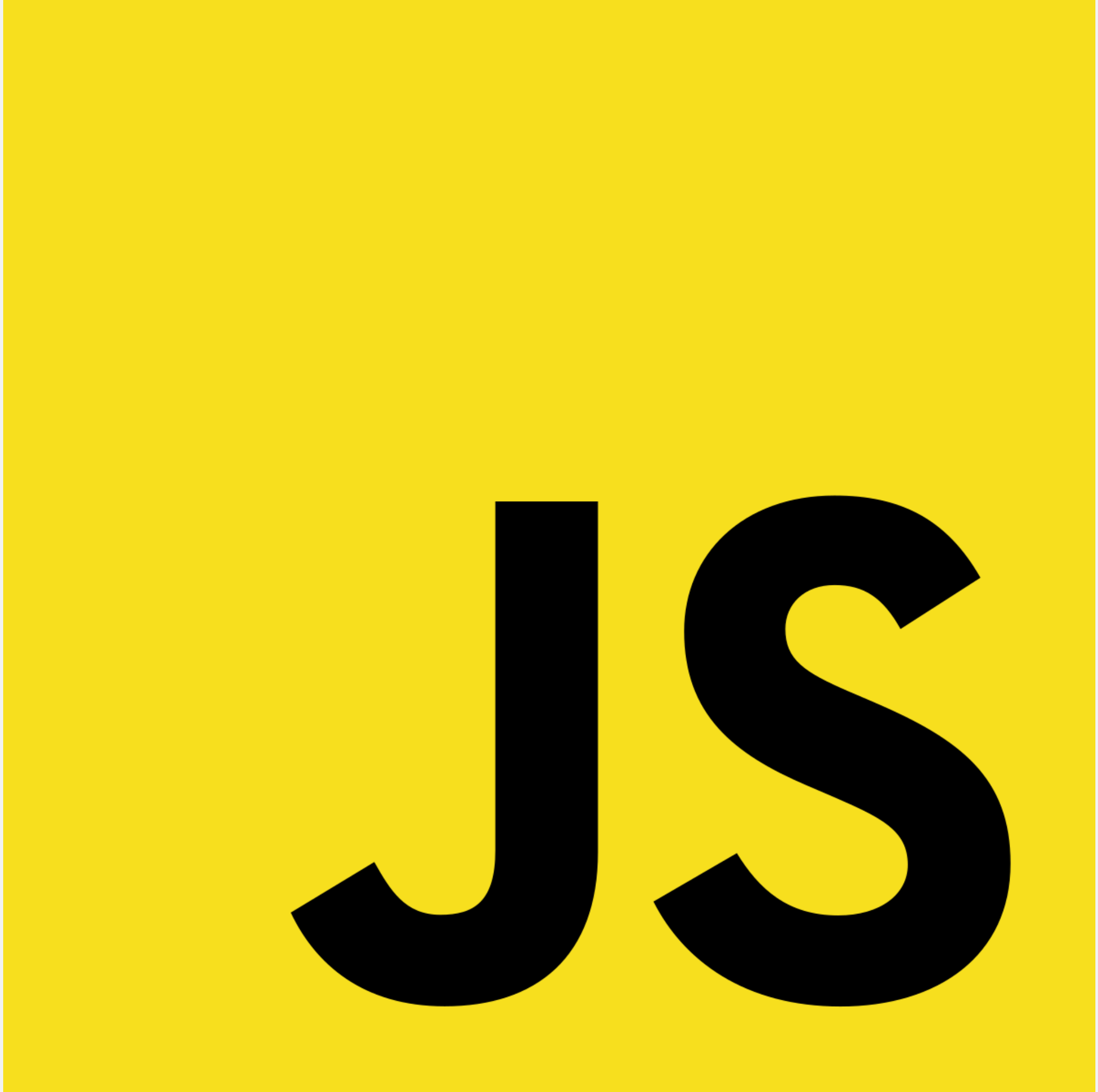
JavaScript

```
curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
sudo apt-get install nodejs
```

```
brew install node
```

- Среда выполнения JavaScript, основанная на движке V8 из браузера Chrome
- Использует неблокирующую модель ввода-вывода
- Пакетная экосистема Node.js и npm является самой большой экосистемой библиотек в мире


JS

JavaScript

- Dynamically typed
- Weakly typed

```
// Statically typed
int a = 10;
string b = 'asdf';

// Dynamically typed
let a = 10;
a = 'asdf';
```

JS

```
// Weakly typed
let a = 1 + '2'; // '12'
```

JS

Comparisons

```
const num = 1;
const strNum = '1';

const sum = num + strNum; // 11

if (num == strNum) {...} // true

if (num === strNum) {...} // false
```

JS

Primitive values (immutable)

- Undefined (`undefined`), used for unintentionally missing values.
- Null (`null`), used for intentionally missing values.
- Booleans (`true` and `false`), used for logical operations.
- Numbers (`-100`, `3.14`, and others), used for math calculations.
- Strings (`"hello"`, `"abracadabra"`, and others), used for text.
- Symbols (uncommon), used to hide implementation details.
- BigInts (uncommon and new), used for math on big numbers.

Objects and Functions (mutable)

- Objects (`{}` and others), used to group related data together.

- Functions (`x => x * 2` and others), used to refer to code.

```
const var1 = 1234;
let var2 = {
  key1: 'value1',
  key2: ['massive', 4321]
};
```

JS

```
function foo(arg1, arg2) {
  const result = arg1 + arg2;

  return result;
}
```

JS

```
const num = 10;

function double(n) {
  return n * 2;
}

double(n);
console.log(n);
```

JS

```js
const obj = { name: 'Valera' };

function getWork(item) {
  item.work = 'Student';
}

getWork(obj);
console.log(obj);
```

JS

```
const arr = ['Tom', 'Jerry'];

arr[0] = 'Tomas';

console.log(arr);
```

JS

```
const str = 'Yikes';

str[0] = 'L';

console.log(str);
```

JS

```
const original = {
  name: 'doc',
  meta: {
    title: 'Hello world'
  }
};

function makeCopy(orig) {
  const copy = {
    name: orig.name,
    meta: orig.meta
  };

  copy.meta.title = 'New World';

  return copy;
}

const copied = makeCopy(original);
```

JS

```
for (let i = 0; i < 10; ++i) {
  console.log('Hello world!');
  if ((i > 5 && true) || false) {
    break;
  }
}
```

JS

```
const arrowFunction = (arg1, arg2) => {
  const result = arg1 + arg2;

  return result;
}
```

JS

```
const arrowFunction = arg1 => {
  const result = arg1 * 2;

  return result;
}
```

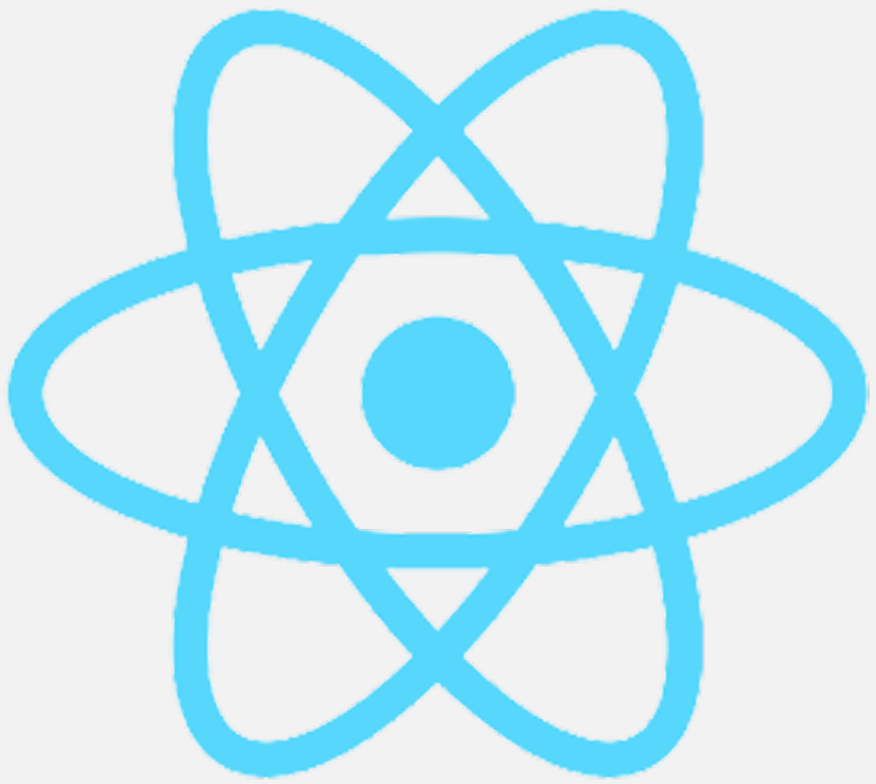
JS

```
const arrowFunction = arg1 => arg1 * 2;
```

JS

```
const even = [1, 2, 3, 4].filter(item => item % 2 === 0);

const squared = [1, 2, 3, 4].map(item => item * item);

const sum = [1, 2, 3, 4].reduce((accumulator, item) =>
accumulator + item, 0);
```

JS



React

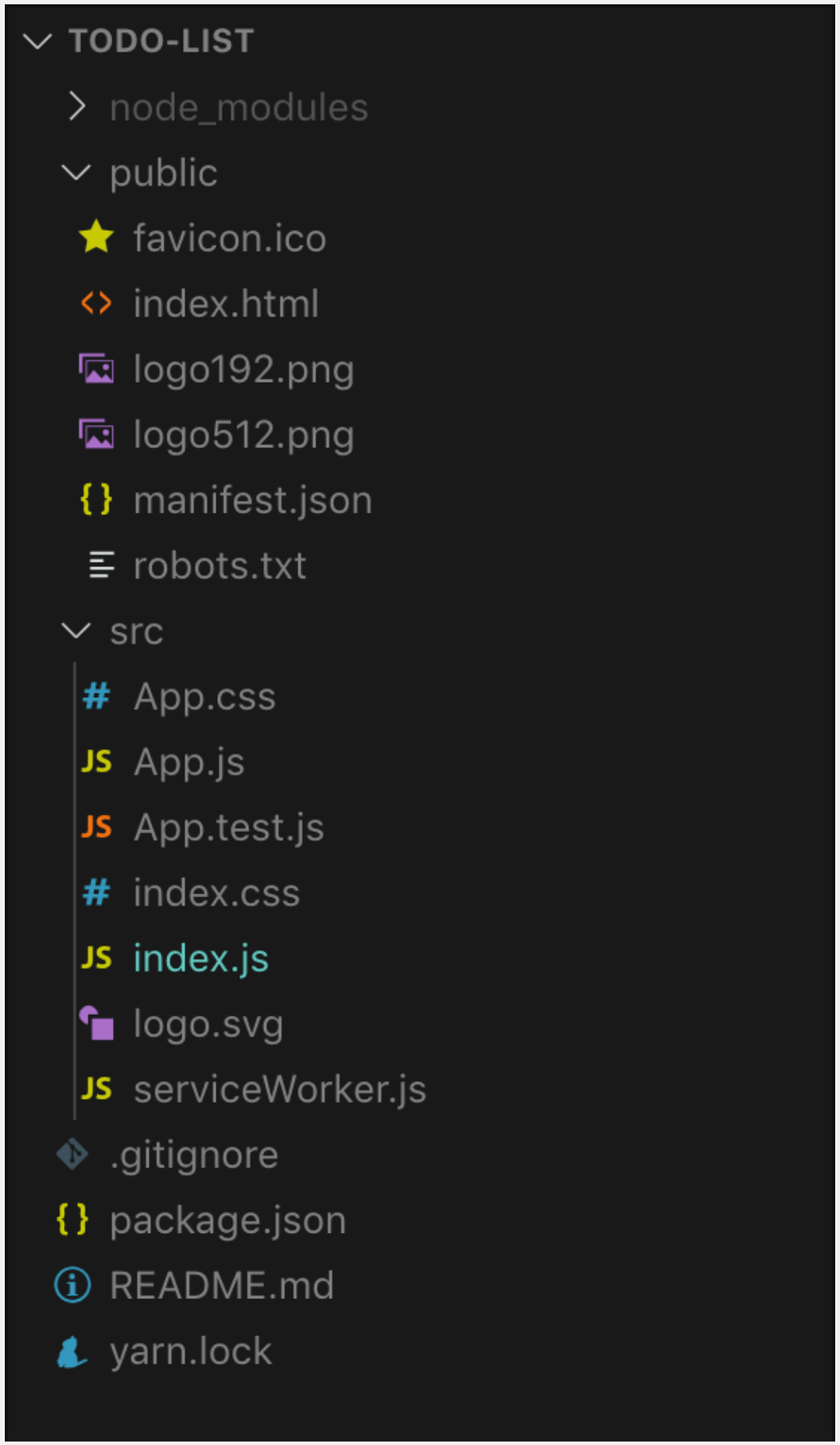
Hello world

```
npx create-react-app my-app
cd my-app
npm run start
```

React

- A JavaScript library for building user interfaces
- Declarative
- Component-based
- Technology stack agnostic

- Designed by Jordan Walke
- First deployed on Facebook’s newsfeed 2011
- Open sourced in May 2013
- Designed for speed, simplicity and scalability

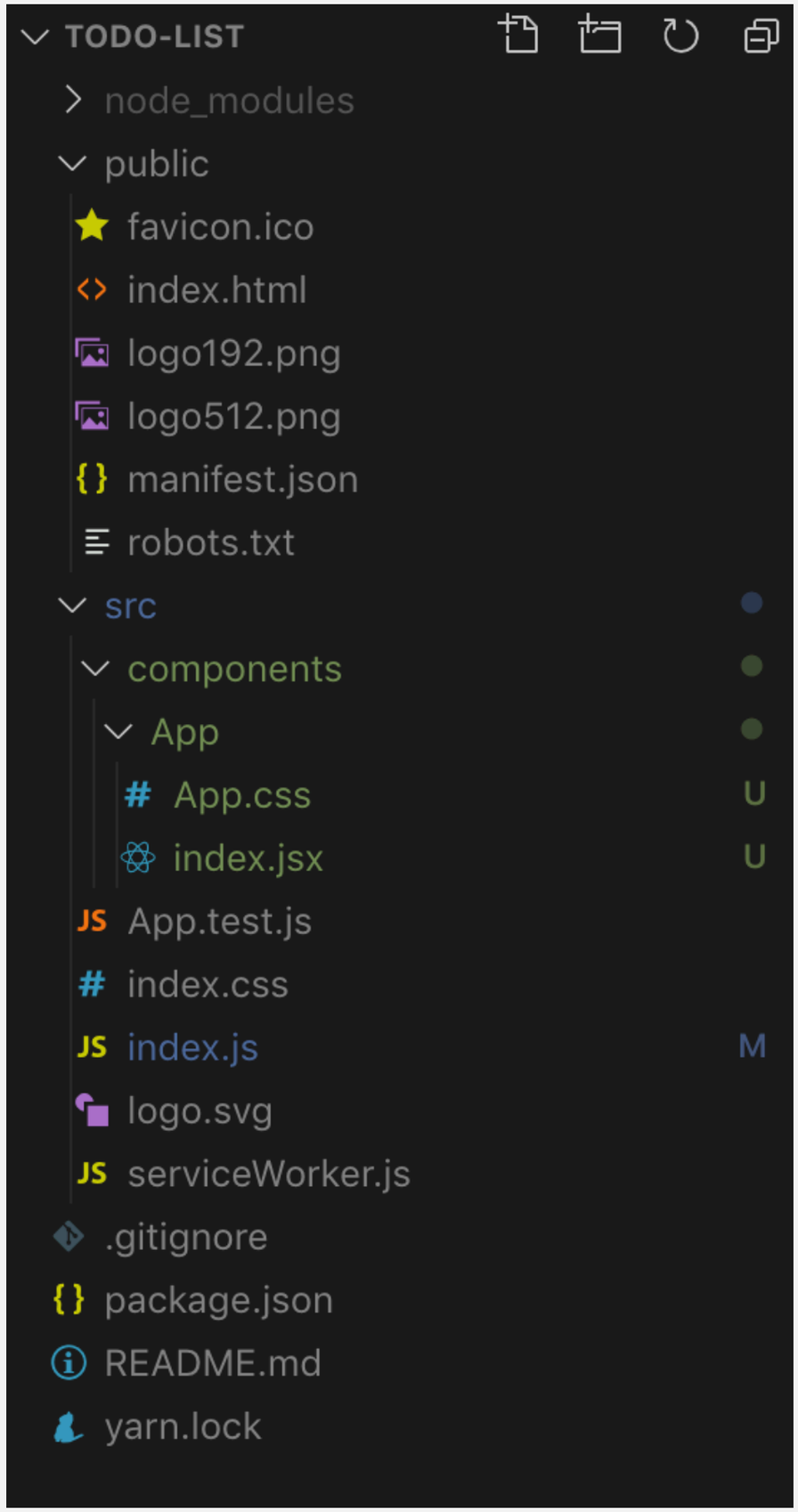

TODO-LIST
node_modules
public
favicon.ico
index.html
logo192.png
logo512.png
manifest.json
robots.txt
src
App.css
App.js
App.test.js
index.css
index.js
logo.svg
serviceWorker.js
.gitignore
package.json
README.md
yarn.lock

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />
    <link rel="apple-touch-icon" href="logo192.png" />
    <!--
      manifest.json provides metadata used when your web app is installed on a
      user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
    -->
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <!--
      Notice the use of %PUBLIC_URL% in the tags above.
      It will be replaced with the URL of the `public` folder during the build.
      Only files inside the `public` folder can be referenced from the HTML.

      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
      work correctly both with client-side routing and a non-root public URL.
      Learn how to configure a non-root public URL by running `npm run build`.
    -->
    <title>React App</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
    <!--
      This HTML file is a template.
      If you open it directly in the browser, you will see an empty page.

      You can add webfonts, meta tags, or analytics to this file.
      The build step will place the bundled scripts into the <body> tag.

      To begin the development, run `npm start` or `yarn start`.
      To create a production bundle, use `npm run build` or `yarn build`.
    -->
  </body>
</html>
```

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import * as serviceWorker from './serviceWorker';

ReactDOM.render(<App />, document.getElementById('root'));

// If you want your app to work offline and load faster, you can change
// unregister() to register() below. Note this comes with some pitfalls.
// Learn more about service workers: https://bit.ly/CRA-PWA
serviceWorker.unregister();
```

```jsx
import React from 'react';
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default App;
```

```
{
  "name": "todo-list",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "react": "^16.9.0",
    "react-dom": "^16.9.0",
    "react-scripts": "3.1.1"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": "react-app"
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```


TODO-LIST
node_modules
public
favicon.ico
index.html
logo192.png
logo512.png
manifest.json
robots.txt
src
components
App
App.css
U
index.jsx
U
App.test.js
index.css
index.js
M
logo.svg
serviceWorker.js
.gitignore
package.json
README.md
yarn.lock

```jsx
const element = <div className="css-class">Hello world</div>;
```

```jsx
const element = <div className="css-class">Hello world</div>;

const element = React.createElement(
  'div',
  {
    className: 'css-class',
  },
  'Hello world'
);

const element = {
  type: 'div',
  props: {
    className: 'css-class',
    children: 'Hello world',
  },
};
```

```jsx
const task = { id: 1, name: 'name', descr: 'descr' };

const Task = () => (
  <div className="container">
    <div className="name">{task.name}</div>
    <div className="descr">{task.descr}</div>
  </div>
);
```