HTTP & Axios

CS568 – Web Application Development I

Assistant Professor Umur Inan
Computer Science Department
Maharishi International University

Maharishi International University - Fairfield, Iowa

Content

- HTTP & HTTP Verbs
- AJAX & JSON
- Axios
  - Config
  - Interceptors
- Axios Example
- React Axios

Hypertext Transport Protocol (HTTP)

HTTP is the underlying protocol used by the World Wide Web and this protocol defines how messages are formatted, and what actions Web servers and browsers should take in response to various commands.

HTTP Verbs

- GET: Retrieves data from the server.
- HEAD: Same as GET, but response comes without the body.
- POST: Submits data to the server.
- PUT: Replace data on the server.
- PATCH: Partially update a certain data on the server.
- DELETE: Delete data from the server.
- OPTIONS: Handshaking and retrieves the capabilities of the server.

AJAX

Asynchronous JavaScript and XML

- Not a programming language, but another asynchronous JavaScript API
- Downloads data from a server in the background
- Allows dynamically updating a page without making the user wait
- Avoids the "click-wait-refresh" pattern

XMLHttpRequest

XMLHttpRequest (XHR) objects are used to interact with servers. You can retrieve data from a URL without having to do a full page refresh. This enables a Web page to update just part of a page without disrupting what the user is doing. XMLHttpRequest is used heavily in AJAX programming.

JSON

JavaScript Object Notation (JSON): Data format that represents data as a set of JavaScript objects

- Natively supported by all modern browsers
- Replaced XML (Extensible Markup Language)

Axios

Axios is a promise-based HTTP Client for node.js and the browser.

- Make **XMLHttpRequests** from the browser
- Make http requests from **node.js**
- By default, axios **serializes** JavaScript objects to **JSON**
- Supports the Promise API **Intercept** request and response
- Transform request and response data
- Client side support for protecting against XSRF

https://axios-http.com/docs/intro

npm install axios -save

Sending Get Request

```
//Blog.js


componentDidMount(){


axios.get('https://jsonplaceholder.typicode.com/posts')
        .then((response)=>{
            console.log(response);
        });
    }
```

Response Schema

```
{
  data: {},
  status: 200,
  statusText: 'OK',
  // headers from the server
  headers: {},
  // config that was provided to `axios` for the request
  config: {},
  // request that generated this response
  request: {}
}
```

Config

You can specify config defaults that will be applied to every request.

Learn more about Axios request config.

```
axios.defaults.baseURL = 'https://api.example.com';
axios.defaults.headers.common['Authorization'] = AUTH_TOKEN;
axios.defaults.headers.post['Content-Type'] =
'application/x-www-form-urlencoded';
```

Interceptors

You can intercept requests or responses before they are handled by then or catch.

Interceptors are common in programing. It obstructs requests or responses to prevent them from continuing to a destination. For example, you can implement an interceptor that stops malicious requests.

In Axios, the interceptors can be used to add common headers like authorization header or to log the requests etc.

Logging requests with Interceptors

```
//index.js
axios.interceptors.request.use(request => {
   console.log(request);
   return request;
}, error => {
   console.log(error);
   return Promise.reject(error);
});
```

Rendering Data to the Screen

```
class Blog extends Component {
    state = {
        posts: []
    }
    componentDidMount() {

axios.get('https://jsonplaceholder.typicode.com/posts')
            .then((response) => {
                this.setState({ posts: response.data });
            });
    }
    render() {
        const posts = this.state.posts.map(item => {
            return <Post
                    key={item.id}
                    title={item.title}>
                    </Post>
        });
```

```
return (
            <div>
                <section className="Posts">
                    {posts}
                </section>
                <section>
                    <FullPost />
                </section>
                <section>
                    <NewPost />
                </section>
            </div>
        );
    }
}
```

Making a Post Selectable

```
//post.js
const post = (props) => (
    <article className="Post" onClick={props.postClicked}>
        <h1>{props.title}</h1>
        <div className="Info">
            <div className="Author">{props.author}</div>
        </div>
    </article>
);
```

```
//Blog.js
state = {
        posts: [],
        clickedPostId:0
    }
    componentDidMount() {
        axios.get('https://jsonplaceholder.typicode.com/posts')
            .then((response) => {
                const posts = response.data.slice(0, 5);
                const updatedPosts = posts.map(item => {
                    return {
                        ...item,
                        author: 'Umur'
                    }
                });
                this.setState({ posts: updatedPosts });
            });
    }
```

Making a Post Selectable

```
//Blog.js
 postClickedHandler = (id)=>{
       this.setState({clickedPostId:id});
   }

   render() {
       const posts = this.state.posts.map(item => {
           return <Post
               key={item.id}
               title={item.title}
               author={item.author}
               postClicked={()=>{this.postClickedHandler
(item.id)}}
           >
           </Post>
       });
```

```
return (
          <div>
              <section className="Posts">
                  {posts}
              </section>
              <section>
                  <FullPost id={this.state.clickedPostId} />
              </section>
              <section>
                  <NewPost />
              </section>
          </div>
      );
  }
```

Making a Post Selectable

```
//Full Post.js
class FullPost extends Component {
    render() {
        let post = <p>Please select a Post!</p>;
        if (this.props.id != 0) {
            post = (
                <div className="FullPost">
                    <h1>Title</h1>
                    <p>Content</p>
                    <div className="Edit">
                        <button
className="Delete">Delete</button>
                    </div>
                </div>
            );
        }
        return post;
    }
}
```

Fetching Data on Update

```
//Full Post.js
class FullPost extends Component {
  state = {
      post: null
   }

    componentDidUpdate() {
      if (this.props.id != 0) {
          if(!this.state.post || (this.state.post &&
this.state.post.id !== this.props.id)){

axios.get('https://jsonplaceholder.typicode.com/posts/' +
this.props.id)
              .then((response) => {
                  this.setState({ post: response.data });
              });
          }
      }
}
```

```
render() {
      let post = <p>Please select a Post!</p>;
      if (this.props.id != 0) {
          post = <p>Fetching!</p>;
      }
      if (this.state.post) {
          post = (
              <div className="FullPost">
                  <h1>{this.state.post.title}</h1>
                  <p>{this.state.post.content}</p>
                  <div className="Edit">
                      <button
className="Delete">Delete</button>
                  </div>
              </div>
          );
      }
      return post;
  }
```

Posting Data to Server

```
//NewPost.js
class NewPost extends Component {
    state = {
        title: '',
        content: '',
        author: 'Umur'
    }
    postDataHandler =()=>{
        const post = {
            title:this.state.title,
            body:this.state.content,
            author: this.state.author
        }
axios.post('https://jsonplaceholder.typicode.com/posts',po
st)
        .then(response =>{
            console.log(response)
        });
    }
render () {
        return (
```

```
div className="NewPost">
                <h1>Add a Post</h1>
                <label>Title</label>
                <input type="text" value={this.state.title}
onChange={(event) => this.setState({title: event.target.value})}
/>
                <label>Content</label>
                <textarea rows="4" value={this.state.content}
onChange={(event) => this.setState({content:
event.target.value})} />
                <label>Author</label>
                <select value={this.state.author}
onChange={(event) => this.setState({author:
event.target.value})}>
                    <option value="Umur">Umur</option>
                    <option value="Aynur">Aynur</option>
                </select>
                <button onClick={this.postDataHandler}>Add
Post</button>
            </div>
        );
    }
```