World Journal of Innovation and Modern Technology, Vol. 7, Issue 5, (Oct) 2024
ISSN 2682-5910
DOI: 10.53469/wjimt.2024.07(05).13

The Application of AJAX and JSON in the Case

Xi Li, Zhengde Bao

School of Computer and Software, Jincheng College, Chengdu, Sichuan 611731, Sichuan, China

Abstract: ***In modern Web development, it is very important to master and utilize AJAX and JSON technology to achieve high-efficiency development. This paper gives an overview of some examples to analyze how to use Bootstrap to display data on the front-end page, then send an asynchronous request to the background with Ajax, and after receiving the request, parse the JSON string to perform the corresponding database operation and obtain the required data from the database. the data package to be returned is then packaged as a JSON object array and returned to the front end of the web, the front end of the data package is analyzed to the desired data and the front end of the webpage is not refreshed, And then the whole interaction process is completed.***

Keywords: Ajax; JSON; Bootstrap.

1. Introduction

With the development of the web, efficient web development is what developers pursue, and convenient and interactive web pages are also what users expect. Currently, in terms of user interactive experience, the most user-friendly and relatively simple technology is Ajax technology [1-3]. In terms of data transmission, the introduction of JSON allows for the transfer of data as objects, making it more convenient to transmit data [4-6]. In terms of displaying data on the front-end, we adopt Bootstrap as the web front-end architecture due to its simplicity, intuitiveness, and robustness. This article provides a brief analysis of using the Bootstrap framework to display Ajax and JSON in specific examples. AJAX and JSON, as indispensable technologies in modern web development, play an important role in enhancing user experience, optimizing network performance, and implementing complex interactive functions. With the rapid development of Internet technology, Web applications are gradually changing from static pages to dynamic and interactive ones [7]. AJAX (Asynchronous JavaScript and XML) and JSON (JavaScript Object Notation) are key technologies in this transformation, greatly improving the performance and user experience of web applications by implementing asynchronous data interaction and lightweight data exchange [8]. AJAX is a technology that exchanges data with a server and updates parts of a webpage without the need to reload the entire page. Its core lies in using the XMLHttpRequest object in JavaScript (or the fetch API in modern browsers) to communicate asynchronously with the server. This technology effectively solves the delay problem caused by page refresh in traditional web applications, improving the smoothness and response speed of user operations [9]. JSON is a lightweight data exchange format based on a subset of JavaScript, but language independent. It uses a text format that is easy for humans to read and write to store and represent data, while also having good clarity and compactness, making it easy for machines to parse and generate. Compared to XML, JSON format is more concise and has higher transmission efficiency, so it has been widely used in web development.

2. Key Technologies

2.1 Ajax

Ajax, It can achieve a small amount of data exchange between the front-end page and the back-end server by using the XMLHttpRequest object to make asynchronous requests to the server-side without refreshing the webpage on the front-end, thus automatically updating the page without refreshing the front-end webpage, greatly improving the user experience. But those traditional web pages that do not use Ajax technology must reload the entire web page by refreshing it in order to achieve real-time desired content. This also means that by using Ajax technology, we not only reduce the amount of data transmission, but also alleviate the burden on the network and improve the user experience.

Ajax is not actually a completely new technology, as its technical support for JavaScript and XML has been used for many years [10]. It is a technology cluster that includes the following four powerful technologies, the first of

which is represented using XHTML+CSS on the web; The second method is to use asynchronous communication to interact with the backend server; The third approach is to use the DOM tree code framework to display and interact with clients in the frontend; The last method is to use specific JavaScript to bundle the corresponding things that need to be bundled [11].

2.2 JSON

JSON (JavaScript Object Notation) is a lightweight data exchange format. It is developed on the basis of JavaScript objects and arrays, and it adopts a text format to store and display information data. The format it uses is very unique, which makes its structural hierarchy clear and easy to read [12]. It is used as an ideal data exchange format for object transmission because it has the advantages of being easy to write and read, easy to parse and generate. After using JSON to transmit and exchange data, the amount of code is reduced while also utilizing the special effects of JSON to reduce the amount of data that needs to be transmitted. By reducing the amount of data transmitted over the network, the efficiency of network transmission is greatly improved [13].

JSON uses specific functions to convert the data to be represented in JavaScript objects into JSON strings, which can be easily passed between web programs. Once it reaches the destination, it can be restored to the data format required by the original application. In general, there are two different forms of JSON data transmission: the first form is a collection of "property/value" objects, which are unordered from each other, starting with "{" and ending with "}", with property names and values indicated by ":" in the middle and separated by ","; There is also a type of collection where the values are numerical and the arrays are ordered, starting with "[" and ending with "]", with values separated by ",".

2.3 Bootstrap

Bootstrap is a front-end open-source toolkit launched by the famous social networking site Twitter, designed in collaboration with renowned designers Mark Otto and Jacob Thornton. Because it is a front-end development framework based on HTML, CSS, and JavaScript, it has the characteristics of simplicity, intuitiveness, and robustness, which make it more convenient to implement web development. Meanwhile, due to its many convenient features, it immediately gained the love of many front-end developers after its launch. It not only includes powerful plugins and built-in components, but also provides an advanced grid system, which makes it possible to present web information more elegantly and greatly improves usability. Moreover, using it as a front-end development technology to display data information can make development flexible and standardized. When displaying data information on front-end pages, using Bootstrap to build tables can make the front-end more beautiful and elegant. Its unique responsive CSS feature can also automatically adjust and adapt to different screen sizes [14-15].

3. Example Analysis

3.1 Instance Development Environment

The appointment course instance is managed using the HBuilderX application development tool on the Windows 10 Home operating system, Navicat Premium to manage the backend database, and WampServer and C++servers to manage the backend server. The language used for this development is HTML5+CSS3.

3.2 Analysis of Instance Function Framework

The appointment course instance mainly uses the Bootstrap framework to display data in the front-end, and then uses Ajax technology to write interface functions to pass the request data in the front-end page in the form of JSON strings to the back-end server. The back-end server then uses the corresponding interface methods to parse the received data segments, completes the corresponding database statement operations, and finally packages the required return data into JSON format for return. After they are returned to the front-end webpage, the front-end webpage finally converts the JSON format into the corresponding object array, and extracts the desired data from the object array for information data presentation.

3.3 Front End Page

3.3.1 Display Teacher Information

The main function of the web front-end homepage is to display teacher course information and interact with users to complete lesson scheduling. This page mainly has three dynamic tables and one form. The first table dynamically displays all teacher details, in which each row of data corresponds to the corresponding data field in the teacher form found in the background database. This dynamic table displaying teacher details is written by the bootstrap framework. This dynamic display function is implemented by the interface function TutorSchedule(). The interface function TutorSchedule() creates a JSON string, and then calls the interface function sendToServer (str, function (data)) to transfer the string to the back end for processing. Then it returns the object array data, and executes the function (data) function to transfer the required data values Render on the web page.

3.3.2 Booking Courses

The first form has two input boxes for entering the course information and grade information of the teacher you want to query. This query is consistent with the teacher appointment form, which is also written by the bootstrap framework. The user enters the teacher's course and grade information they want to search for in each input box and clicks the "Search" button. The "Search" button is bound to the Search_Tutor() function. The core function of the Search_Tutor() function is to obtain the values in the two input boxes, assign them to the strTemp string corresponding to the database query operation, and then use the JSON. stringify() method to convert it into a JSON string using the interface function sendoServer (str), The function (data) is returned to the backend server for corresponding operations, and the returned object array data is then used to present the detailed information of the queried teachers in the frontend using the function (data). It is displayed in the second table created using the bootstrap framework, which shows all teachers who meet the appointment criteria. Then, based on the teacher who wants to make an appointment, click the "Appointment" button and bind the function GetOrderMessage () in response to the "Appointment" button. The function is to display the current course appointment information (including the current appointment teacher, current appointment subject, and current appointment grade) in the pop-up box. The user can confirm whether the current information is correct in the pop-up box. If it is not correct, click the "Cancel" button to cancel the appointment. If it is correct, the function GetOrderMessage () will be displayed in the pop-up box. Enter the teaching address you want to book in the input box of the box, Click the "OK" button again, and the "OK" button will bind the function OrderTutor(). The main function of the OrderTutor() function is to pop up a prompt box to confirm the appointment event. After confirmation, the appointment information such as student name, teacher name, teaching address, course, grade, etc. will be sent to the server in the form of a JSON string by calling the sendoServer (str, function (data)) interface function to store the detailed appointment information in the corresponding database table.

3.3.3 Display reservation information

The third table dynamically displays the appointment information of all successful appointments. It is written by calling the bootstrap framework and is dynamically loaded using window. onload=ShowStudentOrder(). The function ShowStudentOrder() is used to call sendoServer (str, The function (data) function sends the operation statement of querying the reserved course and the student name in the database as a JSON data string to the backend. After parsing, the backend performs the operation of querying the reserved course information database and returns an array of reserved course information objects. Then, the function (data) function dynamically displays the information in the object array data (ID, teacher name, class address, course, grade) in the third table on the frontend page, and there is a cancel reservation button after each reserved course information. The cancel button is bound to the function DeleteOrder (id). This function will pop up a prompt box after clicking, prompting whether to cancel this course or not. After clicking OK, execute sendoServer (str), The function (data) interface function passes the reserved course ID and corresponding database operation as a JSON string to the backend, and performs the operation of deleting reserved course information in the backend database. After execution, the frontend displays a prompt box "Reservation cancelled successfully".

3.3.4 Ajax Asynchronous Processing of JSON Data

This instance mainly performs asynchronous processing of JSON data by calling the sendoServer (tStr, funcName) function. In the instance, the sendoServer (tStr, funcName) method is called multiple times to transfer the JSON string to the C++server, which is processed by C++code. The corresponding database method in the JSON string is applied to the database to perform database operations. Then, the data to be returned is converted into JSON format and returned to the frontend page. The frontend page parses it into the corresponding object array through the funcName function and displays it on the frontend page without refreshing.

(1) The core code of the sendoServer (tStr, funcName) function is as follows:

```
var url = "http://127.0.0.1/server/doserver.php";
function sendToServer(tStr,funcName){
    var xmlhttp;
    if(window.XMLHttpRequest) {
        xmlhttp = new XMLHttpRequest();
    } else {
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }
    var str = tStr;
    console.log('sendMes:'+str);
    var res = "";
    xmlhttp.onreadystatechange = function() {
        if(xmlhttp.readyState == 4 && xmlhttp.status == 200) {
            res = xmlhttp.responseText;
            var jsonVar = JSON.parse(res);
            var result = jsonVar.result;
            switch(result) {
                case 0:
                    funcName(jsonVar);
                    break;
                case -1:
                    alert("参数错误。代码-1。");
                    break;
                default:
                    alert(result);
                    break;
            }
        }
    }
    xmlhttp.open("POST", url, true);
    xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    xmlhttp.send(str);
}
```

(2) The core code for implementing parsing using the function (data) function is as follows:

```
function(data) {
    for (var i in data.ansData) {
        id = data.ansData[i].TutorScheduleID;
        T_UserName = data.ansData[i].T_UserName;
        course = data.ansData[i].T_Course;
        grade = data.ansData[i].T_Grade;
        output +=
            `
    <tbody>
        <tr>
        <td>${id}</td>
        <td>${T_UserName}</td>
        <td>${course}</td>
        <td>${grade}</td>
        <td><button id="create-StudentOrder"
        class="btn btn-primary"
        data-toggle="modal"
        data-target="#create-StudentOrder-modal"
        style="position:relative;top:-10px;"
        onclick="GetOrderMessage('${T_UserName}','${course}','${grade}')">
        预约
        </td>
        </tr>
    </tbody>
    `;
    }
    output += `
            </table>
`
    document.getElementById('ShowSeachTutor').innerHTML = output;
}
```

(3) Presentation of the effectiveness of lesson scheduling examples

When the front-end page of the appointment instance is initially displayed, the first table dynamically displays all the information of all the teachers in the database. This table facilitates students to view all the teacher information and then perform query operations. The specific effect is shown in Figure 1. After entering "Math" and "Grade 2" in the two input boxes below the teacher search, click the "Search" button directly below to search for teachers who meet the criteria. Finally, the search results will be dynamically displayed in a new display and form, as shown in Figure 2. Then click on "Teacher Peng" to make an appointment, click on the appointment button, and a pop-up box will pop up displaying the current course appointment information (current appointment teacher, current

appointment subject, current appointment grade). Enter "C208" in the teaching address input box directly below the pop-up box, as shown in Figure 3. Then click the "OK" button to submit, and in my appointment, you can view all of my current appointment information (ID, teacher name, class address, course, grade), as shown in Figure 4. Finally, click on cancel the appointment of Teacher Wen with ID 36, and my appointment information is shown in Figure 5.

Figure 1: Initial position

Figure 2: After entering Math Grade 2 and clicking on search

Figure 3: Click to make an appointment with Teacher Peng and enter C208 in the input box

Figure 4: My appointment counter after clicking OK to submit the appointment information

Figure 5: After deleting Teacher Wen's appointment information

4. Conclusion

This article discusses the use of Bootstrap framework to create tables and display data on a front-end web page in a class scheduling instance. The data is then accessed through Ajax technology and transmitted using JSON strings as the transmission format, achieving dynamic display and interaction of data without refreshing the front-end. This article analyzes the implementation process of this pattern through an example of scheduling classes. In this pattern, the front-end page uses the Bootstrap framework to create forms, displaying corresponding data in a concise and intuitive manner. At the same time, many strong soldering functions in the Bootstrap framework also make it faster for developers to design front-end pages. Ajax technology uses JSON strings to conveniently transfer data between the front-end and back-end, and asynchronously updates to the front-end, improving system performance and enhancing user experience. In short, this mode not only improves the performance of web systems, but also brings users a more convenient and efficient interactive experience, so this web development mode is more mature and excellent. AJAX and JSON, as key technologies in modern web development, have significantly improved the performance and user experience of web applications by implementing asynchronous data interaction and lightweight data exchange. They have been widely used in multiple scenarios such as real-time chat rooms, dynamic data loading, form validation and submission, real-time search, and have demonstrated enormous potential. With the continuous advancement and innovation of technology, AJAX and JSON will play a more important role in future web development.

References

[1] Wu, Z. (2024). An Efficient Recommendation Model Based on Knowledge Graph Attention-Assisted Network (KGATAX). arXiv preprint arXiv:2409.15315.

[2] Wang, Z., Zhu, Y., Chen, M., Liu, M., & Qin, W. (2024). Llm connection graphs for global feature extraction in point cloud analysis. Applied Science and Biotechnology Journal for Advanced Research, 3(4), 10-16.

[3] Ji, H., Xu, X., Su, G., Wang, J., & Wang, Y. (2024). Utilizing Machine Learning for Precise Audience Targeting in Data Science and Targeted Advertising. Academic Journal of Science and Technology, 9(2), 215-220.

[4] Xu, Y., Shan, X., Guo, M., Gao, W., & Lin, Y. S. (2024). Design and Application of Experience Management Tools from the Perspective of Customer Perceived Value: A Study on the Electric Vehicle Market. World Electric Vehicle Journal, 15(8), 378.

[5] Chen, G., Liu, M., Zhang, Y., Wang, Z., Hsiang, S. M., & He, C. (2023). Using Images to Detect, Plan, Analyze, and Coordinate a Smart Contract in Construction. Journal of Management in Engineering, 39(2), 1-18. https://doi.org/10.1061/JMENEA.MEENG-5121

[6] Shen, Z. (2023). Algorithm Optimization and Performance Improvement of Data Visualization Analysis Platform based on Artificial Intelligence. Frontiers in Computing and Intelligent Systems, 5(3), 14-17.

[7] Zhu, Z., Wang, Z., Wu, Z., Zhang, Y., & Bo, S. (2024). Adversarial for Sequential Recommendation Walking in the Multi-Latent Space. Applied Science and Biotechnology Journal for Advanced Research, 3(4), 1-9.
[8] Wu, Z. (2024). Deep Learning with Improved Metaheuristic Optimization for Traffic Flow Prediction. Journal of Computer Science and Technology Studies, 6(4), 47-53.
[9] Zheng, H., Wang, B., Xiao, M., Qin, H., Wu, Z., & Tan, L. (2024). Adaptive Friction in Deep Learning: Enhancing Optimizers with Sigmoid and Tanh Function. arXiv preprint arXiv:2408.11839.
[10] Chen, G., He, C., Hsiang, S., Liu, M., & Li, H. (2023). A mechanism for smart contracts to mediate production bottlenecks under constraints. 31st Annual Conference of the International Group for Lean Construction (IGLC), 1232 – 1244. https://doi.org/10.24928/2023/0176
[11] Wang, Z., Yan, H., Wang, Y., Xu, Z., Wang, Z., & Wu, Z. (2024). Research on autonomous robots navigation based on reinforcement learning. arXiv preprint arXiv:2407.02539.
[12] Z. Ren, "A Novel Feature Fusion-Based and Complex Contextual Model for Smoking Detection," 2024 6th International Conference on Communications, Information System and Computer Engineering (CISCE), Guangzhou, China, 2024, pp. 1181-1185, doi: 10.1109/CISCE62493.2024.10653351.
[13] Yang, H., Zi, Y., Qin, H., Zheng, H., & Hu, Y. (2024). Advancing Emotional Analysis with Large Language Models. Journal of Computer Science and Software Applications, 4(3), 8-15.
[14] Wang, Z., Chu, Z. C., Chen, M., Zhang, Y., & Yang, R. (2024). An Asynchronous LLM Architecture for Event Stream Analysis with Cameras. Social Science Journal for Advanced Research, 4(5), 10-17.
[15] Z. Ren, "Enhancing Seq2Seq Models for Role-Oriented Dialogue Summary Generation Through Adaptive Feature Weighting and Dynamic Statistical Conditioninge," 2024 6th International Conference on Communications, Information System and Computer Engineering (CISCE), Guangzhou, China, 2024, pp. 497-501, doi: 10.1109/CISCE62493.2024.10653360.