**Лабораторная работа №19-20**

**Хранение данных на клиенте (JSON+COOKIE+WEB STORAGE)**

**JSON**

1 Опишите понятия API, JSON, XML, FRONT-END, BACK-END, AJAX. ЕСТЬ ЛИ МЕЖДУ НИМИ СВЯЗЬ?

Определения и связи

API (Application Programming Interface): Это набор правил и протоколов, которые позволяют различным программным приложениям взаимодействовать друг с другом. API определяет, как запрашивать и передавать данные.

JSON (JavaScript Object Notation): Формат обмена данными, основанный на текстовом представлении объектов JavaScript. Он легкий и удобен для чтения, что делает его популярным для передачи данных через API.

XML (eXtensible Markup Language): Формат разметки, используемый для хранения и передачи структурированных данных. XML более сложен и громоздок по сравнению с JSON.

Front-end: Часть веб-приложения, с которой взаимодействует пользователь. Это интерфейс, написанный с использованием HTML, CSS и JavaScript.

Back-end: Серверная часть приложения, которая обрабатывает запросы, управляет базами данных и выполняет бизнес-логику. Back-end обычно работает с API для взаимодействия с клиентом.

AJAX (Asynchronous JavaScript and XML): Технология, позволяющая веб-приложениям обмениваться данными с сервером асинхронно. AJAX часто использует JSON для передачи данных.

Связь: Все эти понятия тесно связаны. API может возвращать данные в формате JSON или XML, Front-end взаимодействует с API через AJAX для динамического обновления интерфейса, а Back-end обрабатывает запросы и возвращает данные.

2 JSON – что это?  Принцип действия JSON. Зачем он нужен?

JSON — это легковесный формат обмена данными, который легко читается человеком и машиной. Он используется для передачи структурированных данных между клиентом и сервером.

Принцип действия: JSON представляет данные в виде пар "ключ-значение" и может быть использован для сериализации объектов. Он удобен для работы с JavaScript, так как его структура схожа с объектами JavaScript.

Зачем нужен: JSON позволяет эффективно обмениваться данными, особенно в веб-приложениях, где требуется быстрая передача информации между клиентом и сервером.

3 Синтаксис и структура JSON

Синтаксис JSON:

Данные представляются в виде объектов (с фигурными скобками {}) и массивов (с квадратными скобками []).

Ключи (строки) всегда заключены в двойные кавычки.

Пример структуры:

json

Copy

{

"name": "John",

"age": 30,

"isStudent": false,

"courses": ["Math", "Science"],

"address": {

"city": "New York",

"zip": "10001"

}

}

4 Как работать с JSON?

Для работы с JSON в JavaScript используются встроенные методы:

Превращение объекта в JSON: JSON.stringify(object)

Парсинг JSON в объект: JSON.parse(jsonString)

Пример:

javascript

Copy

const obj = { name: "Alice", age: 25 };

const jsonString = JSON.stringify(obj); // Преобразует объект в строку JSON

const parsedObj = JSON.parse(jsonString); // Преобразует строку JSON обратно в объект

5 Преимущества JSON

Легковесность: Меньший размер по сравнению с XML.

Читаемость: Легко читается и пишется человеком.

Совместимость: Прекрасно работает с JavaScript и другими языками программирования.

Производительность: Быстрее парсится по сравнению с XML.

6 Преимущества JSON.

JSON: Лучше подходит для веб-приложений из-за своей легковесности и простоты. Идеален для работы с JavaScript.

XML: Предпочтителен, когда необходима строгая схема и поддержка сложных структур. Часто используется в веб-службах.

**COOKIE и WEB STORAGE**

1 Укажите известные вам способы сохранения данных на клиенте?

Cookies: Небольшие текстовые файлы, сохраняемые на клиенте.

Local Storage: Хранилище для хранения данных без срока действия.

Session Storage: Хранилище для хранения данных, доступное только в рамках одной сессии.

IndexedDB: Базы данных на стороне клиента для хранения значительных объемов структурированных данных.

2 Для чего предназначены cookie и Web storage? Какие преимущества и недостатки имеет каждый способ хранения?

Cookies:

- Предназначение: Хранение небольших объемов данных, таких как идентификаторы сессий и предпочтения пользователя.

- Преимущества: Поддержка на всех браузерах, автоматическая отправка на сервер с каждым запросом.

- Недостатки: Ограниченный объем (обычно 4KB), накладные расходы на отправку с каждым запросом.

Web Storage (Local Storage и Session Storage):

- Предназначение: Хранение больших объемов данных на клиенте без автоматической отправки на сервер.

- Преимущества: Большой объем хранимых данных (до 5-10MB), простота использования.

- Недостатки: Данные не отправляются на сервер, ограниченная поддержка в старых браузерах.

3 Как установить и как получить значения куки? Приведите примеры Установка:

javascript

Copy

document.cookie = "username=John; expires=Fri, 31 Dec 2024 23:59:59 GMT; path=/";

Получение:

javascript

Copy

const cookies = document.cookie; // Получает все куки в виде строки

const username = cookies.split('; ').find(row => row.startsWith('username=')).split('=')[1]; // Извлечение значения

4 Опишите назначение объектов Session storage и Local storage.

Session Storage: Используется для хранения данных, которые должны быть доступны только в рамках текущей вкладки или окна. Данные удаляются, когда вкладка закрывается.

Local Storage: Хранит данные, которые могут быть доступны между сессиями. Данные сохраняются, даже если браузер закрыт и открыт снова.

5 Как установить и как получить значения из хранилища Web storage? Приведите примеры.

Local Storage:

Установка:

javascript

Copy

localStorage.setItem('key', 'value');

Получение:

javascript

Copy

const value = localStorage.getItem('key');

Session Storage:

Установка:

javascript

Copy

sessionStorage.setItem('sessionKey', 'sessionValue');

Получение:

javascript

Copy

const sessionValue = sessionStorage.getItem('sessionKey');

Эти методы позволяют легко сохранять и извлекать данные в браузере, обеспечивая удобный интерфейс для работы с данными на стороне клиента.