COOKIE, LOCAL STORAGE, SESSION STORAGE

- **Цели**

Понять различия между Cookie, Local Storage и Session Storage
Изучить применение каждого из хранилищ в веб-разработке

- **Актуальность**

Значение клиентского хранения данных в современных веб-приложениях

COOKIE, LOCAL STORAGE, SESSION STORAGE

Что такое Cookie

Cookie (произносится как "куки") — это небольшие фрагменты данных, которые веб-сайты сохраняют на компьютере пользователя через веб-браузер. Они предназначены для хранения информации о пользователе и его взаимодействии с сайтом, что позволяет улучшить пользовательский опыт и обеспечить функциональность веб-приложений.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Что хранят в cookie?

COOKIE, LOCAL STORAGE, SESSION STORAGE

Что хранят в cookie:

Аутентификация и управление сессиями: Хранение идентификаторов сессий для поддержания состояния авторизованных пользователей.

Персонализация: Сохранение пользовательских настроек и предпочтений, таких как язык интерфейса или тема оформления.

Отслеживание и аналитика: Сбор данных о поведении пользователей на сайте для анализа и улучшения сервиса.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Основные характеристики Cookie:

1.Имя и значение: Каждый cookie состоит из пары "имя-значение", где имя идентифицирует cookie, а значение хранит соответствующие данные.

2.Домен и путь: Определяют, для какого домена и пути на сервере доступен данный cookie.

3.Срок действия: Устанавливает время, в течение которого cookie будет храниться на устройстве пользователя. Может быть как сессионным (удаляется после закрытия браузера), так и постоянным (хранится до указанной даты).

4.Флаги безопасности:

1. **Secure:** Гарантирует, что cookie будет передаваться только через защищенные соединения (HTTPS).
2. **HttpOnly:** Запрещает доступ к cookie через JavaScript, что помогает защититься от некоторых типов атак, таких как XSS (межсайтовый скриптинг).

COOKIE, LOCAL STORAGE, SESSION STORAGE

История Cookie

Начало концепции (1989):

1. Идея использования небольших фрагментов данных для хранения информации о пользователях впервые появилась в конце 1980-х годов.
2. Тим Бернерс-Ли, создатель World Wide Web, рассматривал механизмы для сохранения информации между запросами.

1.Создание Cookie (1994):

1. **Лоуренс Эйзенберг** из Netscape Communications разработал концепцию cookie как способа хранения информации на клиентской стороне.
2. Целью было улучшение функциональности веб-сайтов, таких как корзины покупок и аутентификация пользователей.

2.Внедрение стандарта (1995):

1. Netscape представила спецификацию HTTP Cookie в своем браузере Netscape Navigator.
2. RFC 2109 и позже RFC 2965 стандартизировали использование cookie в протоколе HTTP, обеспечивая совместимость между различными браузерами и серверами.

COOKIE, LOCAL STORAGE, SESSION STORAGE

4. **Расширение использования (конец 1990-х – начало 2000-х):**
    1. Cookie стали широко использоваться для персонализации контента, отслеживания пользовательской активности и управления сессиями.
    2. Появились первые опасения по поводу приватности, связанные с возможностью отслеживания пользователей без их явного согласия.
5. **Регулирование и приватность (2000-е годы и далее):**
    1. В ответ на беспокойства о приватности были введены различные законы и регуляции, такие как **GDPR** в Европе, требующие прозрачности и согласия пользователей на использование cookie.
    2. Браузеры начали внедрять дополнительные механизмы защиты, такие как блокировка третьесторонних cookie и ограничение их использования.
6. **Современные тенденции:**
    1. С развитием технологий и увеличением внимания к безопасности и приватности, использование cookie становится более контролируемым.
    2. Появляются альтернативы, такие как **Local Storage** и **Session Storage**, которые предлагают другие способы хранения данных на клиентской стороне с различными характеристиками и ограничениями.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Структура Cookie

Каждый Cookie состоит из нескольких компонентов, которые определяют его поведение и доступность:

1.Имя и значение (Name and Value):

1. **Имя:** Уникальный идентификатор Cookie.
2. **Значение:** Данные, связанные с этим именем.
3. **Пример:** sessionId=abc123xyz

2.Домен (Domain):

1. Определяет, для какого домена доступен данный Cookie.
2. **Пример:** Domain=example.com — Cookie будет доступен для всех поддоменов example.com.

3.Путь (Path):

1. Указывает, для какого пути на сервере доступен Cookie.
2. **Пример:** Path=/shop — Cookie будет отправляться только для запросов, начинающихся с /shop.

COOKIE, LOCAL STORAGE, SESSION STORAGE

4. Срок действия (Expires/Max-Age):
• **Expires:** Дата и время, когда Cookie истечет.
• **Max-Age:** Время в секундах, после которого Cookie станет недействительным.
• **Пример:** Expires=Wed, 21 Oct 2025 07:28:00 GMT или Max-Age=3600

5. Флаги безопасности:
• **Secure:** Гарантирует, что Cookie будет передаваться только через защищенные соединения (HTTPS).
• **HttpOnly:** Запрещает доступ к Cookie через JavaScript, что помогает защититься от некоторых типов атак, таких как XSS.
• **SameSite:** Определяет, будет ли Cookie отправляться при запросах из других сайтов. Значения могут быть Strict, Lax или None.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Типы Cookie

1. Сессионные Cookie (Session Cookies):

1. Хранятся только в течение текущей сессии браузера.
2. Удаляются автоматически после закрытия браузера.
3. Используются для временного хранения данных, например, содержимого корзины покупок.

2. Постоянные Cookie (Persistent Cookies):

1. Сохраняются на устройстве пользователя до истечения срока действия.
2. Используются для хранения предпочтений пользователя, таких как язык интерфейса или тема.

3. Третьесторонние Cookie (Third-Party Cookies):

1. Устанавливаются доменом, отличным от посещаемого пользователем сайта.
2. Часто используются для отслеживания пользователей и рекламных целей.
3. Современные браузеры и регуляции всё чаще ограничивают использование третьесторонних Cookie из-за соображений приватности.

COOKIE, LOCAL STORAGE, SESSION STORAGE

4. HttpOnly Cookie:

1. Недоступны для JavaScript через document.cookie.
2. Предназначены для защиты от атак типа XSS.

5. Secure Cookie:

1. Передаются только по защищённому соединению (HTTPS).
2. Повышают безопасность передачи данных.

6. SameSite Cookie:

1. Управляет отправкой Cookie при кросс-доменных запросах.
2. Значения:
    1. **Strict:** Cookie не отправляется при кросс-доменных запросах.
    2. **Lax:** Cookie отправляется при некоторых кросс-доменных запросах, например, при переходе по ссылке.
    3. **None:** Cookie отправляется при всех кросс-доменных запросах (требуется Secure).

COOKIE, LOCAL STORAGE, SESSION STORAGE

4. HttpOnly Cookie:

1. Недоступны для JavaScript через document.cookie.
2. Предназначены для защиты от атак типа XSS.

5. Secure Cookie:

1. Передаются только по защищённому соединению (HTTPS).
2. Повышают безопасность передачи данных.

6. SameSite Cookie:

1. Управляет отправкой Cookie при кросс-доменных запросах.
2. Значения:
    1. **Strict:** Cookie не отправляется при кросс-доменных запросах.
    2. **Lax:** Cookie отправляется при некоторых кросс-доменных запросах, например, при переходе по ссылке.
    3. **None:** Cookie отправляется при всех кросс-доменных запросах (требуется Secure).

COOKIE, LOCAL STORAGE, SESSION STORAGE

Ограничения Cookie

1.Размер:

1. Каждый Cookie ограничен по размеру до **4 КБ**.
2. Это ограничение накладывает ограничения на объем данных, которые можно хранить в одном Cookie.

2.Количество на домен:

1. Большинство браузеров ограничивают количество Cookie на один домен (обычно около 20-50).
2. При превышении этого лимита старые Cookie могут быть удалены для освобождения места.

3.Передача с каждым запросом:

1. Cookie автоматически отправляются на сервер с каждым HTTP-запросом к соответствующему домену и пути.
2. Это может увеличить объем передаваемых данных и повлиять на производительность.

4.Безопасность:

1. Cookie могут быть уязвимы для атак, если не используются соответствующие флаги безопасности.
2. Не рекомендуется хранить в Cookie чувствительную информацию без дополнительной защиты (например, шифрования).

COOKIE, LOCAL STORAGE, SESSION STORAGE

Уязвимости:

1. XSS (Межсайтовый скриптинг):

1. Если Cookie не имеет флага HttpOnly, злоумышленник может получить доступ к его содержимому через внедрённый скрипт.

2. CSRF (Подделка межсайтовых запросов):

1. Злоумышленник может заставить браузер пользователя отправить запрос с поддельными данными, используя существующие Cookie.

3. Перехват Cookie (Man-in-the-Middle):

1. Без использования HTTPS и флага Secure Cookie могут быть перехвачены при передаче по сети.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Методы защиты:

1. **Использование флагов безопасности:**
    1. **HttpOnly:** Предотвращает доступ к Cookie через JavaScript.
    2. **Secure:** Обеспечивает передачу Cookie только по защищённым соединениям.
    3. **SameSite:** Ограничивает отправку Cookie при кросс-доменных запросах, уменьшая риск CSRF.
2. **Минимизация хранения чувствительных данных:**
    Не храните в Cookie информацию, которая может быть использована для компрометации безопасности пользователя (например, пароли).
3. **Шифрование данных в Cookie:**
    Шифруйте содержимое Cookie для дополнительной защиты данных.
4. **Регулярная проверка и обновление Cookie:**
    1. Устанавливайте разумные сроки действия для Cookie и обновляйте их при необходимости.
    2. Удаляйте ненужные или устаревшие Cookie.
5. **Валидация данных на сервере:**
    1. Не доверяйте данным из Cookie без проверки на серверной стороне.
    2. Используйте подписанные или зашифрованные Cookie, чтобы предотвратить подделку.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Local Storage
Локальное хранилище предоставляет веб-приложениям возможность сохранять данные на стороне клиента в виде пар "ключ-значение". В отличие от Cookie, данные в Local Storage не отправляются на сервер с каждым HTTP-запросом, что делает его более эффективным для хранения больших объемов данных.

Основные характеристики:
•**Объем хранилища:** Обычно до **5 МБ** на домен, что значительно больше, чем ограничение в 4 КБ для одного Cookie.

•**Персистентность данных:** Данные сохраняются до явного удаления, независимо от закрытия браузера или перезагрузки устройства.

•**Доступность:** Доступен только на стороне клиента через JavaScript, без автоматической отправки на сервер.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Основные функции Local Storage:

- **Кэширование данных:** Сохранение часто используемых данных для ускорения загрузки страниц.
- **Сохранение состояния приложения:** Хранение информации о текущем состоянии интерфейса, например, выбранные вкладки или фильтры.
- **Персонализация:** Сохранение пользовательских настроек и предпочтений.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Ограничения Local Storage

1. **Объем хранилища:**
    1. Ограничение около **5 МБ** может быть недостаточным для хранения больших объемов данных.
    2. В некоторых браузерах ограничение может варьироваться.
2. **Одно доменное пространство:**
    1. Данные доступны только для того домена, который их установил.
    2. Нельзя разделять данные между разными доменами или поддоменами.
3. **Отсутствие автоматической синхронизации с сервером:**
    Данные не передаются автоматически на сервер, что требует дополнительной логики для синхронизации, если это необходимо.
4. **Безопасность:**
    1. Уязвим для атак типа XSS, поскольку доступен через JavaScript.
    2. Нет встроенных механизмов защиты данных, таких как шифрование.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Уязвимости:

1.XSS (Межсайтовый скриптинг):

Если злоумышленник внедрит скрипт на страницу, он может получить доступ к данным в Local Storage.

2.Отсутствие шифрования:

Данные в Local Storage хранятся в открытом виде и могут быть легко прочитаны, если злоумышленник получит доступ к устройству пользователя.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Практики безопасности:

1.Защита от XSS:

1. Используйте Content Security Policy (CSP) для ограничения источников скриптов.
2. Избегайте вставки пользовательского ввода напрямую в HTML или JavaScript без предварительной валидации и экранирования.
3. Применяйте фреймворки и библиотеки, которые автоматически защищают от XSS.

2.Минимизация хранения чувствительных данных:

1. Не храните в Local Storage конфиденциальную информацию, такую как пароли, токены аутентификации или личные данные.
2. Используйте серверные сессии и другие безопасные механизмы для хранения чувствительной информации.

COOKIE, LOCAL STORAGE, SESSION STORAGE

3. Шифрование данных:
•Если необходимо хранить чувствительные данные, применяйте криптографические методы для их шифрования перед сохранением в Local Storage.

4. Валидация и проверка данных:
•Проверяйте и валидируйте все данные, полученные из Local Storage, перед использованием их в приложении.
•Не доверяйте данным из Local Storage без проверки на серверной стороне.

5. Использование безопасных соединений:
•Обеспечьте использование HTTPS для защиты данных при передаче между клиентом и сервером.
•Это предотвращает перехват данных, если они каким-либо образом передаются из Local Storage на сервер.

6. Регулярное очищение данных:
•Удаляйте ненужные данные из Local Storage, чтобы минимизировать риск утечки информации.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Критерий	Local Storage	Session Storage	Cookie
Объем хранилища	До 5 МБ	До 5 МБ	До 4 КБ на Cookie
Персистентность	Постоянные, до явного удаления	Сессионные, до закрытия вкладки или браузера	Сессионные или до заданного срока
Передача на сервер	Нет	Нет	Автоматически с каждым HTTP-запросом
Доступность	Только клиентская сторона	Только клиентская сторона	Клиентская и серверная стороны
Безопасность	Уязвим для XSS, нет встроенной защиты	Уязвим для XSS, нет встроенной защиты	Уязвим для XSS и CSRF, но может быть защищён флагами
Использование	Кэширование, сохранение состояния, персонализация	Временное хранение данных, формы, сессии	Управление сессиями, аутентификация, отслеживание

COOKIE, LOCAL STORAGE, SESSION STORAGE

Уязвимости клиентских хранилищ:

- **XSS (Межсайтовый скриптинг):** Злоумышленники могут внедрять вредоносные скрипты на веб-страницы, которые получают доступ к данным в Cookie или Local Storage.

- **CSRF (Подделка межсайтовых запросов):** Злоумышленники могут инициировать нежелательные действия от имени пользователя, используя его аутентификационные данные, хранящиеся в Cookie.

- **Man-in-the-Middle (Перехват данных):** Без использования защищённых соединений (HTTPS) данные могут быть перехвачены при передаче между клиентом и сервером.

COOKIE, LOCAL STORAGE, SESSION STORAGE

Для повышения безопасности Cookie рекомендуется использовать специальные флаги **HttpOnly** и **Secure**.

1 Флаг HttpOnly:

•**Описание:** Флаг **HttpOnly** запрещает доступ к Cookie через JavaScript (document.cookie). Это предотвращает доступ к Cookie через вредоносные скрипты, что защищает от атак типа XSS.

•**Установка:** Добавляется к заголовку Set-Cookie при отправке Cookie с сервера.

2 Флаг Secure:

•**Описание:** Флаг **Secure** гарантирует, что Cookie будет передаваться только через защищённые соединения (HTTPS). Это предотвращает перехват Cookie злоумышленниками при передаче данных по незащищённым каналам.

•**Установка:** Также добавляется к заголовку Set-Cookie.