Введение в куки

Термин cookies (это множественное число, произносится как «кукис» или «куки») был введён в веб-программирование благодаря компании Netscape. В единственном числе часто говорят «кука».

Кука — это небольшая именованная порция информации, которая хранится в каталоге, связанном с браузером пользователя (а не на сервере), но которую сценарий (как клиентский, так и серверный) волен в любой момент изменить.

Фактически, куки – это расшаренное между браузером и сервером пространство данных.

Когда запрос направляется браузером веб-серверу, то в числе заголовков есть Cookie (если хоть какие-то куки установлены).

Серверный сценарий получает все куки, которые сохранены на клиентском компьютере, при каждом своем запуске, так что он может в любой момент своей работы узнать, что у пользователя установлено. Иными словами, если мы хотим, чтобы в течение некоторого времени серверу отправлялась какая-то информация при каждом запросе (с этого браузера) – надо установить куку.

Самым удобным в куках является то, что они могут храниться недели и годы до тех пор, пока их не обновит сервер или же пока не истечет срок их жизни (который тоже назначается сценарием при создании куки). Таким образом, мы можем иметь куки, срок жизни которых как всего несколько минут (или до того момента, пока не закроют браузер), так и несколько лет.

Кука связывается с доменом и может быть установлена и прочитана с помощью JavaScript, в виде строки, доступной через document.cookie.

Эта строка состоит из пар ключ=значение, которые перечисляются через точку с запятой с пробелом "; ".

Следовательно, чтобы прочитать куку, достаточно разбить строку по "; ", и затем найти нужный ключ. Это можно делать либо через split и работу с массивом, либо через регулярное выражение.

- установка: https://kodaktor.ru/set_cookie
- чтение: https://kodaktor.ru/get_cookie

Поскольку нет прямых методов для установки конкретного значения в конкретный ключ, это по современным меркам неудобный вариант обращения, если нужно хранить клиентские данные между уходами со страницы. Более современный вариант – localStorage.

https://kodaktor.ru/vue_simple_tasker

Это локальная БД типа «ключ:значение»

Каждый браузер хранит свои куки отдельно. То есть куки, установленные при пользовании Chrome, не будут видны при работе в FireFox, и наоборот.

Каждой куке сопоставлено время его жизни, которое хранится вместе с ним. Кроме этого, имеется также информация об имени сервера, установившего эту куку, и URL каталога, в котором находился сценарий-хозяин в момент инициализации (за некоторыми исключениями).

Если при установке куки дату не указать, то она будет считаться «сессионной» и удаляется при закрытии браузера. Если дата в прошлом, то кука будет удалена.

localStorage хранит данные без срока, очищается только с помощью JavaScript, или очисткой кэша браузера / Locally Stored Data.

sessionStorage удаляет данные при закрытии браузера (не вкладки).

Зачем нужны имя сервера и каталог? Сценарию передаются только те cookies, у которых параметры с именем сервера и каталога совпадают с хостом и каталогом сценария соответственно (на самом деле каталог не должен совпадать полностью, он может являться подкаталогом того, который создан для хранения кук). Так что совершенно невозможно получить доступ к «чужим» кукам — браузер просто не будет посылать их серверу.

В принципе, любой ресурс, который загружает браузер, может поставить куку.

Например:

- Если на странице есть <img src="http://mail.ru/counter.gif">, то вместе с картинкой в ответ сервер может прислать заголовки, устанавливающие куки.
- Если на странице есть <iframe src="http://facebook.com/button.php">, то во-первых сервер может вместе с button.php прислать куки, а во-вторых JS-код внутри ифрейма может записать в document.cookie

При этом куки будут принадлежать тому домену, который их поставил. То есть, на mail.ru для первого случая, и на facebook.com во втором.

Такие куки, которые не принадлежат основной странице, называются «сторонними» (3rd party) куками. Не все браузеры их разрешают.

Серверная сторона может приказывать браузеру сохранить куки. Мы можем послать заголовок или сгенерировать метатэг.

<meta http-equiv="Set-Cookie" content="name=value; expires=дата; domain=имя_хоста; path=путь; secure" >

Заголовок выглядит аналогично:

Set-Cookie: name=value; expires=дата; domain=имя_хоста; path=путь; secure

Получить куки для серверного сценария легко: все они хранятся в переменной окружения HTTP_COOKIE в таком же формате, как и QUERY_STRING, только вместо символа амперсанда & используется точка с запятой ;. Например, если мы установили два cookies: cookie1=value1 и cookie2=value2, то в переменной окружения HTTP_COOKIE будет следующее:

cookie1=value1;cookie2=value2

Сценарий должен распарсить эту строку, распаковать её и затем работать по своему усмотрению.

Зайдите на https://fork.kodaktor.ru/evil и выполните сценарий:

```
document.cookie = 'first=1';

(async()=>{
 const r = await fetch('/getheaders').then(x=>x.text());
 alert(r);
})();
```

вы увидите продублированные сервером заголовки, которые пришли с этим запросом. Там будет и заголовок Cookie, отправленный браузером:

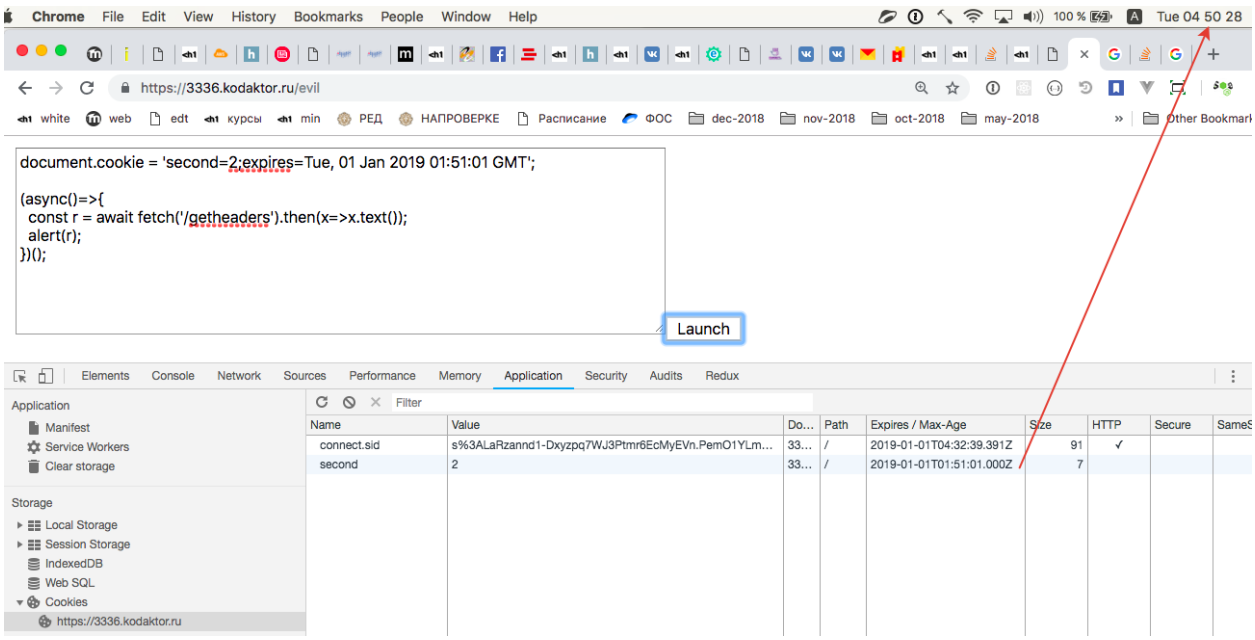
Затем поменяйте первую строку сценария

```
document.cookie = 'first=1;expires=Thu, 01 Jan 1970 00:00:01 GMT';

(async()=>{
 const r = await fetch('/getheaders').then(x=>x.text());
 alert(r);
})();
```

first=1 исчезнет
Дело в том, что мы написали, что кука должна была исчезнуть ещё в прошлом веке.

Можно провести эксперимент с временем жизни
допустим, сейчас Вторник, 1 января 2019 года, 04:50:00

```
document.cookie = 'second=2;expires=Tue, 01 Jan 2019 01:51:01 GMT';

(async()=>{
  const r = await fetch('/getheaders').then(x=>x.text());
  alert(r);
})();
```



а через 1 минуту

```
(async()=>{
  const r = await fetch('/getheaders').then(x=>x.text());
  alert(r);
})();
```

и куки "second" уже не будет ни в заголовках, посланных серверу, ни в разделе Application

Есть ещё варианты указания максимального срока жизни в секундах:

```
document.cookie = 'second=2; max-age=30';
```

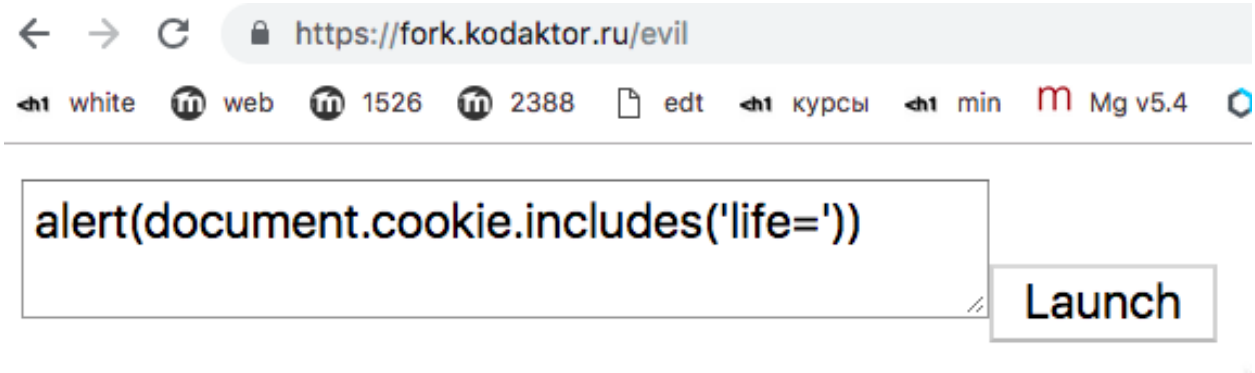
или

```
const date = new Date();
date.setTime(date.getTime()+(30*1000));
document.cookie = 'second=2; expires=' + date.toGMTString();
```

и проверяем document.cookie.includes('second=')
или req.headers.cookie.includes('second=')
через 30 секунд после установки это значение станет false

Установим с помощью Node.js куку

- зайдите на https://fork.kodaktor.ru/evil
- введите alert(document.cookie.includes('life='))
- нажмите кнопку и убедитесь что получаете false
- перейдите в соседнем окне по ссылке https://fork.kodaktor.ru/setheadercookie
- снова нажмите кнопку (сразу) и убедитесь, что получаете true
- выясните, через какое время кука уничтожается




```
const ck = {'Set-Cookie': 'life=on; expires='+new Date(new
Date().getTime()+...).toUTCString()};
rs.writeHead(200, { ...ck });
```