Browser to Server communication

Hyper Text Transfer Protocol

HTTP -это протокол прикладного уровня для передачи данных от браузера к серверу и обратно. HTTP сообщение обычно передаются между сервером и браузером через порт 80 или 443 при использовании SecureHTTP (HTTPS).

Web server

Веб сервер ответственный за получение и обработку запросов полученных через HTTP. Веб сервер обрабатывает запрос и отправляет ответ обратно веб браузеру. После отправки ответа веб сервер закрывает соединении с браузером и освобождает все ресурсы, которые были задействованы при обработке запроса.

Web client (browser)





Веб браузер -независимое от платформы приложение для запроса и отображения HTML страниц. В обязанности браузера входит отображение информации полученной с сервера и получение информации от пользователя для отправки ее обратно на сервер.

Request

При запросе страницы браузер отправляет текстовую команду на сервер.

GET /default.aspx HTTP/1.1

Host: www.example.com

GET-HTTP глагол (метод или команда) описывающая действие, которое должен выполнить веб сервер.
/default.aspx -запрашиваемая на сервере страница.
HTTP/1.1-версия протокола
Host: www.example.com -заголовок. Доменное имя сайта к которому выполняется запрос. Полезно в том случае если на сервере одновременно работает несколько веб приложений

Response

HTTP/1.1 200 OK

Server: Microsoft-IIS/6.0

Content-Type: text/html

Content-Length: 36

<html><body>Hello world</body></html>

HTTP/1.1-версия протокола
200-status code
OK-описание статуса

Server: Microsoft-IIS/6.0 -заголовок хранящий версию сервера
Content-Type: text/html -заголовок с MIME типом ответа. Данное значение нужно для того, что бы браузер правильно интерпретировал данные полученные от сервера
Content-Length: 36 -размер тела ответа в байтах
<html><body>Hello world</body></html> -тело ответа

HTTP Methods (verbs)

HTTP	**Описание**
OPTIONS	Используется клиентским приложением для получения списка доступных глаголов.
GET	Получение данных с сервера.
HEAD	Получение метаданных (заголовков) ресурса. При данном запросе ресурс не возвращается.
POST	Отправка данных на сервер для обработки. Обычно данные введенные пользователем в форме на странице.
PUT	Позволяет клиенту создать ресурс по указанному URL (создать файл на сервере)
DELETE	Удаление ресурса на сервере
CONNECT	Команда для использования с прокси серверами

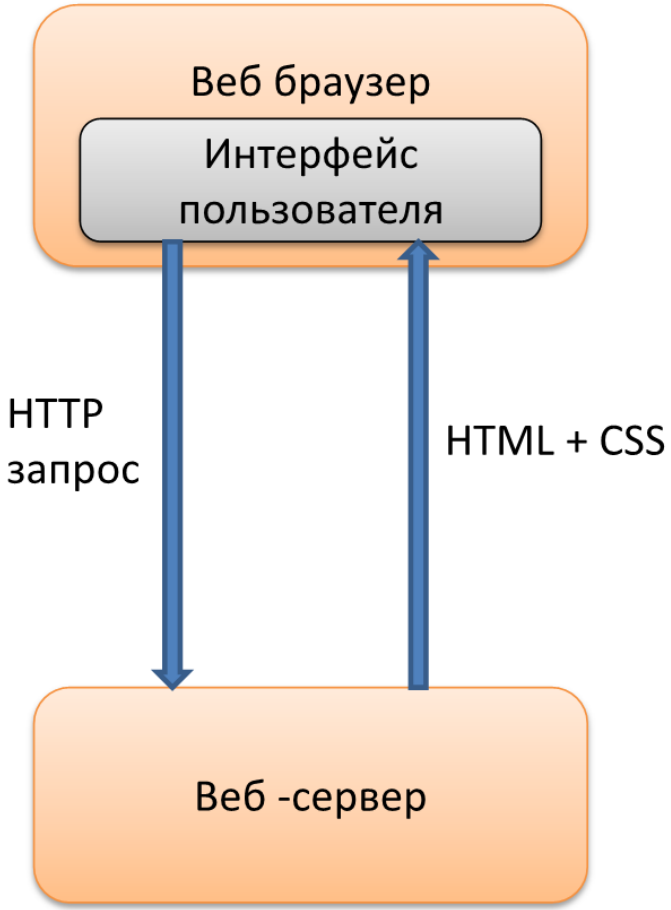
Status code groups

Группа	Описание
1xx	Информационные
2xx	Успешное завершение
3xx	Команды перенаправлений
4xx	Клиентские ошибки
5xx	Серверные ошибки

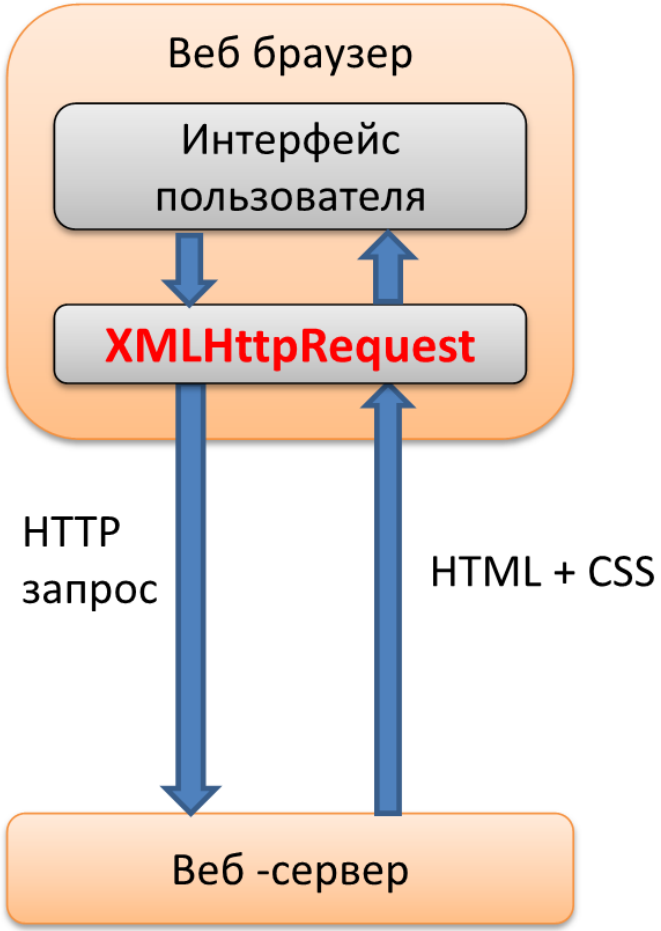
Asynchronous JavaScript and XML (AJAX)

AJAX–Asynchronous JavaScript And XML. Модель для запросов данных от сервера в фоновом режиме, без перезагрузки веб-страницы.

Classic web application model



Web application model with AJAX



XMLHttpRequest

```
var xhr = new XMLHttpRequest();            // Создание объекта для HTTP запроса.
xhr.open("GET", "public/data.json", true);  // Настройка объекта для отправки асинхронного GET запроса

// функция-обработчик срабатывает при изменении свойства readyState
// Значения свойства readyState:
// 0 - Метод open() еще не вызывался
// 1 - Метод open() уже был вызван, но метод send() еще не вызывался.
// 2 - Метод send() был вызван, но ответ от сервера еще не получен
// 3 - Идет прием данных от сервера. Для значения 3 Firefox вызывает обработчик события несколько раз IE только один раз.
// 4 - Ответ от сервера полностью получен (Запрос успешно завершен).

xhr.onreadystatechange = function () {
  if (xhr.readyState == 4) { // если получен ответ
    if (xhr.status == 200) { // и если статус код ответа 200
      document.getElementById("json-output").innerHTML += jsonToHtml(xhr.responseText); // responseText - текст ответа полученного с сервера.
    }
  }
}

xhr.send();                                // Отправка запроса, так как запрос асинхронный сценарий продолжит свое выполнение. Когда с сервера придет ответ сработает событие onreadystatechange
```

Fetch

```
fetch('/public/data.json')      // Вызов ф-и fetch с передачей url
  .then(function (response) {   // Ответ с сервера приходит в параметр response
    return response.json();     // response дает доступ к телу ответа в нужном формате, заголовкам headers, статусу status и некоторым другим полям ответа
  })
  .then(function (data) {
    console.log(data.data);     // В data лежит наш ответ с сервера
  });
```

- response.arrayBuffer()
- response.blob()
- response.formData()
- response.json()
- response.text()

Cross-origin resource sharing (CORS)

Обычно запрос XMLHttpRequest может делать запрос только в рамках текущего сайта. При попытке использовать другой домен/порт/протокол – браузер выдаёт ошибку.

Browser:

GET /your_url/segment

Host:anywhere.com

Origin:http://mysite.com

...

Server:

HTTP/1.1 200 OK
Content-Type:text/html; charset=UTF-8
Access-Control-Allow-Origin: http://mysite.com
...

Q&A