

Лекция 1

Введение в Web

О чем курс?

- Фундаментальный web
- Advanced javascript
- Клиентская разработка (vanilla js)
- Серверная разработка (node.js)
- Real-time messages (polling, websocket)

Лекции

1. Введение в Web
2. Основы JavaScript
3. Модель клиент-сервер
4. Запросы AJAX
5. Асинхронный JavaScript
6. Бекенд на JavaScript
7. Web реального времени

Контрольные мероприятия

- Модуль 1
 - 1 РК
 - 3 ЛР
 - 1 ДЗ
- Модуль 2
 - 1 РК
 - 3 ЛР
 - Доп задание

Web

Подробнее тут <https://vc.ru/selectel/76371-cto-proishodit-kogda-polzovatel-nabiraet-v-brauzere-adres-sayta>

Базовый сценарий работы web-приложения

- Пользователь вводит URL
- Браузер загружает страницу - HTML документ
- Браузер анализирует (парсит) HTML и загружает доп. ресурсы
- Браузер отображает (рендерит) HTML-страницу

URL - unified resource locator

<http://mi-ami.ru:8080/profile/account.html?gender=male&age=13#comments>

- **http** - протокол
- **mi-ami.ru** - доменное имя (DNS имя сервера)
- **8080** - TCP порт
- **/profile/account.html** - путь до документа
- **?gender=male&age=13** - query-параметры (параметры запроса)
- **#comments** - якорь

Документы

Документ - это тело ответа HTTP-запроса. Он может иметь несколько типов (MIME-типы):

- text/html
- text/css
- text/javascript
- image/png
- video/mp4
- и так далее...

Документы

По смыслу документы можно разделить на статические и динамические.

Статические:

- Файлы на дисках сервера, зачастую с постоянным адресом

Динамические:

- Создаются на каждый запрос
- Содержимое зависит от внешних факторов (пользователя, времени и тд)
- Адрес может меняться (может быть и постоянным)

Ресурсы

```
1 <link rel="stylesheet" href="/css/index.css">
2 <script src="http://code.jquery.com/jquery-2.1.4.js">
3 </script>
4 
```

Ресурсы

```
1 .slide {  
2   background-image: url(../pictures/network.png)  
3 }  
4  
5 @font-face {  
6   font-family: Terminus;  
7   src: url(fonts/terminus.ttf);  
8 }
```

HTTP

Протоколы

Существует большое множество различных сетевых протоколов связи.
Самые распространенные:

- TCP
- UDP
- **HTTP (работает поверх TCP)**
- FTP
- SSH

HTTP - HyperText Transfer Protocol

Основой HTTP является технология «клиент-сервер»: всегда есть клиент, который посылает запрос, и сервер который получает запрос и отдает нужный ответ.

Изначально использовался для передачи исключительно HTML, но вскоре был расширен при помощи MIME-типов.

Отсюда делаем вывод, что каждый запрос браузера за ресурсами - это HTTP-запрос.

Структура HTTP-запроса

Каждое HTTP-сообщение состоит:

- метод (GET, POST, PUT, DELETE и тд);
- URL запроса (адрес ресурса);
- заголовки - характеризуют тело сообщения, параметры передачи и прочие сведения;
- тело - может отсутствовать.

Структура HTTP-ответа

Ответ как правило состоит также из тела и заголовков, а также из статуса ответа. Различают 5 видов статусов:

- **1xx** - информативный статус
- **2xx** - успешный статус
- **3xx** - перенаправление
- **4xx** - клиентская ошибка
- **5xx** - ошибка сервера

Пример HTTP-запроса

```
1 GET http://www.ru/robots.txt HTTP/1.0
2 Accept: text/html, text/plain
3 User-Agent: curl/7.64.1
4 If-Modified-Since: Fri, 24 Jul 2015 22:53:05
  GMT
```

Пример HTTP-ответа

```
1 HTTP/1.1 404 Not Found
2 Server: nginx/1.5.7
3 Date: Sat, 25 Jul 2015 09:58:17 GMT
4 Content-Type: text/html; charset=iso-8859-1
5 Connection: close
6
7 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
8 <HTML><HEAD>...
```

HTML

Подробнее тут <http://htmlbook.ru/html>

Как выглядит HTML

```
1 <!DOCTYPE html>
2 <html>
3     <head>
4         <title>Страница</title>
5         <meta http-equiv="Content-Type"
6             content="text/html; charset=utf-8">
7         <meta name="description" content="Сайт">
8         <link rel="stylesheet" href="./style.css">
9     </head>
10    <body id="the_body">
11        <p class="article">...</p>
12        <script src="./script.js"></script>
13    </body>
14 </html>
```

DOCTYPE

DOCTYPE - указание типа содержимого.

- HTML 5

```
1 <!DOCTYPE html>
```

- HTML 4 (Строгий синтаксис)

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
2 "http://www.w3.org/TR/html4/strict.dtd">
```

HTML теги (верхний уровень)

- html - обертка
- head - заголовок (не отображается на странице)
- body - тело (то, что видит пользователь)

HTML теги (внутри head)

- title - название страницы (отображается в заголовке браузера)

```
1 <title>Страница</title>
```

- meta - дополнительная информация для браузера

```
1 <meta name="description" content="Для друзей">
```

- link - подключение ресурсов (например, CSS)

```
1 <link rel="stylesheet" href="/style.css">
```

- script - загрузка JavaScript

```
1 <script src="./jquery.js"></script>
```

HTML теги (внутри body)

- h1 - h6 - заголовки
- p - параграфы
- div - абстрактный блочный контейнер
- span - абстрактный строчный контейнер
- a - гиперссылки
- img - изображение
- ul, ol, li - маркированные списки

Гиперссылки

- href - URL гиперссылки
- target - в каком окне открывать
- name - название якоря (вместо href)

```
1 <a href="http://iu5.bmstu.ru" target="_blank"></a>
```

Формы

- action - URL, куда будет отправлена форма
- method - GET или POST
- enctype - способ кодирования

```
1 <form method="POST">
2   <input name="image" type="file">
3   <input name="id" type="hidden" value="3">
4   <input name="nick" type="text">
5   <input type="submit" value="Отправить">
6 </form>
```

CSS

Подробнее тут <http://htmlbook.ru/css>

Как выглядит CSS

```
1 .mid-play {  
2     padding:13px 0px 0px 13px;  
3 }  
4 p.inner-play a {  
5     color:#3c3c3c;  
6     text-decoration: underline;  
7 }  
8 .big-top {  
9     background-image: url(/img/pc/220_130_top.gif);  
10 }
```

Как задать стили?

- Встроены в браузер (у каждого тега)
- Внешний файл

```
1 <link rel="stylesheet" href="style.css">
```

- В HTML коде

```
1 <style>...</style>
```

- В HTML теге

```
1 
```

Какие бывают стили?

- width, height - размер элемента
- margin, padding - границы и отступы
- display - отображение
- color - управление цветом
- background - фон элемента
- font - управление шрифтом
- text-align - выравнивание текста

Классы и идентификаторы

- id - идентификатор элемента (уникален на странице)

```
1 <div id="userpic"></div>
```

- class - список классов элемента (может повторяться на странице)

```
1 <button class="btn btn-main">Одобрить</div>
```

CSS селекторы (базовые)

- Универсальный селектор

```
1 * { margin: 0px; padding: 0px; border: 0px; }
```

- Имена тегов

```
1 p { margin-top: 10px; }
```

- Имена классов (с точки)

```
1 .btn { border: solid 1px gray; }
```

- Идентификатор тегов (с решетки)

```
1 #userpic { padding: 10px }
```


CSS селекторы (сложные)

- Контекстные (вложенные)

```
1 div.article a { text-decoration: underline }
```

- Дочерние (вложенность = 1 уровень)

```
1 a > img { border: 2px }
```

- Соседние

```
1 h2.sic + p { margin-left: 30px }
```

- Группировка

```
1 h1, h2 { color: red }
```

Наследование стилей

```
1 <head>
2   <style>
3     body { color: darkgray; font-family: Arial; }
4     p { font-size: 110% }
5   </style>
6 </head>
7 <body>
8   <p> Привет, <a href="/">Мир</a> </p>
9 </body>
```

Приоритеты стилей

В случае, если два разных стиля конфликтуют между собой, применяется тот, что обладает большей **специфичностью**.

Если специфичность двух стилей совпадает, применяется тот, что расположен **ниже** в HTML/CSS коде.

Указание в значении стиля флага **!important** позволяет перекрыть проверку специфичности.

```
1 <head>
2   <style>
3     p { color: red; }
4     .class { color: black; }
5     #id { color: blue; }
6   </style>
7 </head>
8 <body>
9   <p id="id" class="class">
10     Hello World!
11   </p>
12 </body>
```

JavaScript

Подробнее тут <https://learn.javascript.ru>

Как выглядит JavaScript

```
1 <!DOCTYPE HTML>
2 <html>
3 <body>
4   <p>Перед скриптом...</p>
5   <script>
6     alert( 'Привет, мир!' );
7   </script>
8   <p>...После скрипта.</p>
9 </body>
10 </html>
```

Как загрузить JavaScript?

- Внешний файл

```
1 <script src="./jquery.js"></script>
```

- В HTML коде

```
1 <script>...</script>
```