**Содержание**

|  |  |
| --- | --- |
| Введение...…………………………………………………………………….. | 3 |
| 1. JavaScript, особенности и сферы применения…………………………… | 7 |
| 2. Обзор фреймворка jQuery. Достоинства и недостатки………………….. | 11 |
| 3. Bootstrap фреймворк второго уровня…………………………………….. | 14 |
| 4. Обзор фреймворка Ext JS. Достоинства и недостатки………………….. | 16 |
| Заключение…………………………………………………………………… | 18 |
| Список использованной литературы……………………………………….. | 19 |

# Введение

С появления первых сетей для обмена информации между американскими университетами началось совершенствование методов доступа и распространения информации. Постепенно происходил переход от простого набора статических документов, оформленных с помощью специальных языков разметки (*HTML*, *XML*), произошёл переход к сложным веб приложениям. Кроме того изменились сами цели использования веб-технологий. В настоящее время больше не достаточно предоставлять информацию с простейшим графическим оформлением. Современные реалии требуют сложную и эффективную обработку данных.

С начала развития сетей *internet* постоянно увеличивалось их использование во всех сферах жизнедеятельности человека. Однако темп совершенствования оборудования не позволяет обеспечивать все увеличивающиеся потребности. В какой-то момент стало очевидным необходимость создание мощного средства программирования на стороне клиента, позволяющего не только переложить часть нагрузки на клиента, но и создавать более эффективные веб-приложения. Наиболее удачным решением стало создание и внедрение скриптового языка программирования *JavaScript*. В настоящее время данный язык используется в большинстве разработанных и разрабатываемых веб-приложений.

Не смотря на постоянное развитие и совершенствование, *JavaScript* столкнулся с одной сложностью. Разработчики браузеров на протяжении длительного времени не могли договориться об единообразии работы с веб-документами, что требовало разработки отдельного варианта скрипта для каждого браузера. Для решения этой задачи появляются *JS­­*-фреймворки. С течением времени фреймворки становились более совершенными, получили свою направленность для решения определенного круга задач.

В настоящее время *JavaScript* и его фреймворки широко используются в следующих областях:

1. Электронный бизнес. По результатам исследований, большинство ресурсов Интернета так или иначе связаны с коммерческой деятельностью. Интернет используется для рекламы и непосредственной продажи товаров и услуг, для маркетинговых исследований, электронных платежей и управления банковскими счетами.
2. Средства массовой информации. По жанрам интернет-издания не отличаются от офлайновых – есть новостные сайты, литературные, научно-популярные, детские, женские и т. п. Однако, если офлайновые издания выпускаются периодически (раз в день, неделю, месяц), то интернет-издания обновляются по мере появления нового материала. Также существует интернет-радио и интернет-телевидение.
3. Литература, музыка, кино. Электронные библиотеки, доступные через Интернет, содержат огромное число произведений. При этом многие книги, доступные в сети, давно стали библиографической редкостью, а некоторые и вообще не издавались.
4. Связь. Электронная почта в настоящее время является одним из наиболее используемых средств связи. Также популярны IP-телефония и использование таких программ, как Skype (бесплатное проприетарное программное обеспечение с закрытым кодом, обеспечивающее шифрованную голосовую связь и видеосвязь через Интернет между компьютерами (VoIP), используя технологии пиринговых сетей, а также платные услуги для звонков на мобильные и стационарные телефоны). В последние годы популярность обрели мессенджеры, передающие сообщения через Интернет, они стали вытеснять из повседневной жизни людей сотовую связь, которая в сравнении с ними чаще всего уступает в функциональности, скорости, а также стоимости.
5. Общение. Интернет является способом массового общения людей, объединённых различными интересами. Для этого используются интернет-форумы, блоги и социальные сети. Социальные сети стали своего рода Интернет-пристанищем, где каждый может найти техническую и социальную базу для создания своего виртуального «Я». При этом каждый пользователь получил возможность не просто общаться и творить, но и делиться плодами своего творчества с многомиллионной аудиторией той или иной социальной сети.
6. Краудсорсинг. Интернет оказался хорошим средством для решения общественно значимых задач силами множества добровольцев, координирующих при этом свою деятельность.

Таким образом, актуальность исследования обусловлена широким использованием веб-технологий в различных сферах деятельности, а так же высокими темпами развития веб-сервисов.

Так же совместно с научным руководителем были определены объект, предмет, цель и задачи исследования:

**Объектом исследования** является процесс разработки клиентской части веб-приложений и сайтов различной сложности с использованием *JS*-фреймворков.

**Предметом исследования** является подходы и методы оптимизации разработки сайтов и веб-приложений реализованные в конкретном *JS*-фреймворке.

**Цель НИР**: анализ и выявление тенденций развития и оптимизации процесса разработки современных веб-приложений с использованием *JS*-фреймворков.

**Задачи НИР**. Для достижение поставленной цели необходимо решить следующие задачи:

* проанализировать подходы к разработке клиентской части веб-приложения на основе *jQuery* и *Ext* *JS*;
* проанализировать возможностей использования *JS*-фреймворков, а так же их эффективности при решении различных задач;
* выявить положительные и отрицательные стороны использования конкретного фреймворка, а так же определить спектр эффективно решаемых задач.

**Теоретическая основа исследования.** В качестве теоретической основы исследования послужили труды зарубежных специалистов в области веб-разработки C. Lindley, M. Haverbeke, A. Osmani, E. Hoigaard, A. MacCaw, N Zakas, P. Koch, N. Zakas, Д. Крейн, Э. Паскарелло, Д. Джеймс, T. Holowaychuk, A. Young, R. Murphey, Б. Бибо, И. Кац, J. Garcia и др.

**Методологическая основа исследования.** В исследовании широко использовались методы сравнения, обобщения, классификации.

**Информационная база исследования.** В качестве информационной базы исследования использованы инструкции разработчиков.

**Результаты НИР:** выделенные тенденции развития и оптимизации разработки веб-приложений и интерфейсов; выявленные перспективы использования и развития *JS*-фреймворков, их достоинства и недостатки.

# 1. JavaScript, особенности и сферы применения

Не смотря на существование большого количества технологий направленных на разработку веб-приложений, несомненным лидером в настоящее время является *JavaScript(JS)*. Данный язык программирования представляет собой мультипарадигменный язык программирования [1], направленный на создание клиентской части веб-приложения с использованием наиболее эффективных методик программирования нашего времени. В настоящее время *JS* позволяет использовать все преимущества объектно-ориентированного, императивного и функционального стилей программирования [1]. В своих возможностях язык направлен на реализацию стандарта *ECMA*-262.

*JavaScript* предоставляет возможности в создании объектно-ориентированных клиентских приложений, однако в связи с реализацией других парадигм программирования, объектная модель имеет целый ряд особенностей по сравнению с другими языками программирования. Разнообразие предоставляемых средств предоставляемых средств делает язык гибким и позволяет осуществить быстрый переход с других языков программирования. Так же следует отметить что стандарт постоянно расширяют добавлением новых возможностей. Можно сказать что данный язык постоянно совершенствуется и впитывает наиболее удачные идеи [2 – 6].

По своему синтаксису язык схож с Си:

* все идентификаторы регистрозависимы,
* в названиях переменных можно использовать буквы, подчёркивание, символ доллара, арабские цифры,
* названия переменных не могут начинаться с цифры,
* для оформления однострочных комментариев используются //, многострочные и внутристрочные комментарии начинаются с /\* и заканчиваются \*/.

Однако имеет ряд особенностей [1]:

* объекты с возможностью интроспекции;
* функции как объекты первого класса;
* автоматическое приведение типов;
* автоматическая сборка мусора;
* анонимные функции.

Структуру языка можно представить в виде трех частей [2, 7]:

1. Ядро (*ECMAScript*);
2. Объектная модель браузера (*Browser Object Model* или *BOM*);
3. Объектная модель документа (*Document Object Model* или *DOM*).

Ядро представляет собой основу для построения скриптового языка (*ECMAScript*). Описывает типы данных, инструкции, ключевые слова, операторы, объекты, регулярные выражения, не ограничивая возможности расширения функционала.

Объектная модель браузера ­представляет собой прослойку между ядром и объектной моделью документа [8]. При этом основная задача заключается в управлении окнами браузера и обеспечении их взаимодействия. Каждое из окон браузера представляется объектом *window*, центральным объектом *DOM* [9].

Помимо управления окнами *BOM* обычно обеспечивает следующие сущности [8]:

* управление фреймами,
* поддержка задержки в исполнении кода и зацикливания с задержкой,
* системные диалоги,
* управление адресом открытой страницы,
* управление информацией о браузере,
* управление информацией о параметрах монитора,
* ограниченное управление историей просмотра страниц,
* поддержка работы с HTTP cookie.

Объектная модель документа представляет собой интерфейс программирования для работы с XML и HTML документами [9]. В общем виде:

* генерация и добавление узлов,
* получение узлов,
* изменение узлов,
* изменение связей между узлами,
* удаление узлов.

В настоящее время *JavaScript* широко применяется при реализации:

* веб-приложений (клиентской части) – клиент-серверных программ, в котором клиентом является браузер, а сервером – веб-сервер, имеющих распределённую между сервером и клиентом логику [10];
* *AJAX* – популярном подходе к построению интерактивных пользовательских интерфейсов веб-приложений, заключающемся в «фоновом» асинхронном обмене данными браузера с веб-сервером [11, 12];
* *сomet* – широкое понятие, описывающее механизм работы веб-приложений, использующих постоянные *HTTP*-соединения, что позволяет веб-серверу отправлять данные браузеру без дополнительного запроса со стороны браузера [13];
* браузерные операционные системы – исходный код некоторых операционных систем более чем на половину состоит из *JS* [14];
* бурмарклеты – используется для создания небольших программ, размещаемых в закладки браузера [15];
* пользовательские скрипты в браузере – это программы, написанные на JavaScript, выполняемые в браузере пользователя при загрузке страницы и позволяющие автоматизировать часть работ связанных с управлением пользовательским интерфейсом [16];
* серверные приложения – *JavaScript* может исполняться на серверах, использующих *Java* с 6 версии [17]. Наиболее популярные платформы: *Jaxer*, per*severe-framework*, *Helma*, *v8cgi*, *node.js* [18 – 20];
* виджеты – вспомогательная мини-программа для добавления дополнительных функций к пользовательскому интерфейсу;
* и др.

Однако существенным недостатком *JavaScript* являются существенные отличия в обработке скриптов различными браузерами. Скрипт разработанный без учета особенностей обработки различных интерпретаторов с большой вероятностью не будет корректно работать на иных браузерах или более старых версиях. Существует два распространённых способа решения этой проблемы:

1. Первый путь заключается в использование полифиллов. Полифилл (англ. *polyfill*) – это библиотека, которая добавляет в старые браузеры поддержку возможностей, которые в современных браузерах являются встроенными [21, 22]. Например библиотека *ES5 shim* позволяет использовать функции добавленные стандартом *ES5* в *Internet Explorer8*.
2. Использование JS-фреймворков – позволяет достичь высокого уровня абстракции и обеспечить кросс-браузерность приложения за счет использования готовых решений. Среди наиболее известных можно отметить *Ember.js*, *Adobe Spry*, *AngularJS*, *Dojo Toolkit*, *Extjs*, *jQuery*, и др. [1, 19, 22].

Преимущества использования JavaScript фреймворков [2 – 4, 22]:

1. Эффективность – проекты, которые раньше заняли бы месяцы и сотни строчек кода, сейчас могут быть реализованы намного быстрее с хорошо структурированными готовыми шаблонами и функциями.
2. Безопасность – лучшие JavaScript фреймворки имеют фирменную систему безопасности и поддерживаются крупным сообществом, члены которого и просто пользователи выступают в роли тестировщиков.
3. Расходы – большинство фреймворков с открытым кодом и бесплатны. Поскольку они помогают программистам быстрее разрабатывать пользовательские решения, итоговая цена веб приложения будет ниже.

# 2. Обзор фреймворка jQuery. Достоинства и недостатки

Одним из широко используемых *JS* фрейморков является *jQuery*. Распространение данного фреймворка обусловлено широким применением для создания простых интерфейсов пользователя в современных веб приложениях. Не секрет что для создания типовых сайтов (блоги, интернет магазины, новостные порталы) используются различные *CMS*, при этом некоторые по умолчанию используют *jQuery* (*Drupal*), а некоторые позволяют установить этот фреймворк в качестве расширения (*Joomla* 3). Наличие большого числа сторонних библиотек позволяют в короткие сроки создать полнофункциональный сайт средней сложности [22].

*jQuery* представляет собой *JavaScript* библиотеку фокусирующуюся на работе с *DOM* моделью, то есть обеспечивает взаимодействие *JavaScript* и *HTML*. Библиотека *jQuery* помогает легко получать доступ к любому элементу *DOM*, осуществлять работу с атрибутами и содержимым элементов. В библиотеке используется подход при котором работа функции зависит от передаваемых аргументов. Таким образом одна функция *attr* позволяет осуществлять всё необходимые действия связанные с модификацией атрибутов тегов. На рисунке 1 приведены примеры использования функции *attr* [23 – 26].

|  |  |
| --- | --- |
| $("div").attr("class") | вернет имя класса у первого div-элемента на странице |
| $("div").attr("class", "divEl") | класс всех div-элементов на странице станет равным divEl |
| $("div").attr({"class":"divEl", "title":"Див"}) | класс div-элементов станет равен divEl, а title — "Див" |
| $(".rool").attr("title") | вернет подсказку элемента с классом rool. Если таких элементов на странице несколько — вернет первого из них. |

Рисунок 2.1 – примеры использования функции *attr* библиотеки *jQuery*

Также библиотека *jQuery* предоставляет удобный *API* для работы с *AJAX* запросами [11, 24]. Использование обёртки предоставляет возможность значительно сократить и упростить создание *AJAX* запроса (рис. 2.2).

|  |  |
| --- | --- |
| $.ajax({  type: "POST",  url: "some.php",  data: "name=John&location=Boston",  success: **function**(msg){  alert( "Прибыли данные: " + msg );  }  }); | var xhr = new XMLHttpRequest();  xhr.open('GET', '/my/url', true);  xhr.send();  xhr.onreadystatechange = function() {  if (this.readyState != 4) return;  if (this.status != 200) {  alert( 'ошибка: ' + (this.status ? this.statusText : 'запрос не удался') );  return;  }  } |

Рисунок 2.2 – Примеры AJAX для *jQuery* и *JavaScritp*

Таким образом основные возможности [24, 25]:

* Движок кросс-браузерных *CSS*-селекторов *Sizzle*, выделившийся в отдельный проект;
* Переход по дереву *DOM*, включая поддержку *XPath* как плагина;
* События;
* Визуальные эффекты;
* *AJAX*-дополнения;
* *JavaScript*-плагины.

Точно так же, как *CSS* отделяет визуализацию от структуры *HTML*, *JQuery* отделяет поведение. Например, вместо прямого указания на обработчик события нажатия кнопки управление передаётся *JQuery*, которая идентифицирует кнопки и затем преобразует его в обработчик события клика. Такое разделение поведения и структуры также называется принципом ненавязчивого *JavaScript*.

Библиотека *jQuery* содержит функциональность, полезную для максимально широкого круга задач. Тем не менее, разработчиками библиотеки не ставилась задача совмещения функций, которые подошли бы всюду, поскольку это привело бы к большому коду, большая часть которого не востребована. Поэтому была реализована архитектура компактного универсального ядра библиотеки и плагинов. Это позволяет собрать для ресурса именно ту *JavaScript*-функциональность, которая на нём была бы востребована [26].

Таким образом достоинства использования *jQuery*:

1. Облегчает манипулирование моделью документа DOM;
2. Множество различных эфектов;
3. Выполнение Ajax-запросов;
4. Кросс-браузерная совместимость;
5. Модульность jQuery, Вы можете нарастить функционал, подключая модули;
6. Простота использования.

Недостатки:

1. Скорость выполнения. Да, чистый JavaScript работает быстрее (в умлых руках), но говорить о быстродействии не совсем правильно (растет производительность пользовательских компьютеров, да и сами разработчики постоянно наращивают быстродействие самого фреймвёрка)

Это размер библиотеки. Размер jQuery библиотеки порядка 19 кБ. Не так уж и много, но все же. Опять же это спорный минус при современных скоростях интернета или при использовании CDN (об этом мы поговорим в следующей статье, когда остановимся на вариантах подключения jQuery библиотеки).

# 3. *Bootstrap* фреймворк второго уровня

Одной из немало важных особенностей *jQuery* является появление фреймворков основанных на нем. Одним из наиболее распространённых является *Bootstrap*.

*Bootstrap* – свободный набор инструментов для создания сайтов и веб-приложений. Включает в себя *HTML*- и *CSS*-шаблоны оформления для типографики, веб-форм, кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса, включая *JavaScript*-расширения [27].

Этот фреймворк начал разрабатываться как внутренняя библиотека компании *Twitter* под названием *Twitter Blueprint*. После нескольких месяцев разработки он был открыт под названием *Bootstrap* 19 августа 2011 года [28].

Основными нововведениями второй версии, появившейся 31 января 2012 года, стали 12-колоночная сетка и поддержка адаптивности [27].

Третья версия выпущена 19 августа 2013 года. В ней адаптивность получила дальнейшее развитие, был осуществлён переход к концепции *mobile first*, оптимизации прежде всего под мобильные устройства. Дизайн по умолчанию стал плоским [29].

Работа над четвёртой версией начата 29 октября 2014 года. Альфа версия вышла 19 августа 2015 года [30].

К недостаткам можно отнести бедную цветовую гамму стандартного набора иконок. К преимуществам — хорошую реализацию *grid*-сетки для масштабирования веб-страницы, создания адаптивного дизайна.

Основные инструменты *Bootstrap* [27]:

Сетки – заранее заданные размеры колонок, которые можно сразу же использовать, например ширина колонки 140 px относится к классу .span2 (.col-md-2 в третьей версии фреймворка), который можно использовать в CSS-описании документа.

Шаблоны – фиксированный или резиновый шаблон документа.

Типографика – описания шрифтов, определение некоторых классов для шрифтов, таких как код, цитаты и т. п.

Медиа – представляет некоторое управление изображениями и видео.

Таблицы – средства оформления таблиц, вплоть до добавления функциональности сортировки.

Формы – классы для оформления форм и некоторых событий, происходящих с ними.

Навигация – классы оформления для табов, вкладок, страничности, меню и панели инструментов.

Алерты – оформление диалоговых окон, подсказок и всплывающих окон.

# 4. Обзор фреймворка *Ext JS*. Достоинства и недостатки

*Ext JS* представляет собой *JS*-фреймворк для построения интерактивных веб-приложений с использованием *AJAX*, *DHTML* и *DOM*. Библиотека включает в себя широкий набор элементов управления для использования в веб-приложениях: текстовые поля, дата и время, числовые поля, списки, элементы выбора, редактор *HTML,* деревья и прочее. Многие из этих элементов позволяют легко настроить взаимодействие с сервером через *AJAX* запросы.

После выхода библиотеки версии 1.1 обновление происходило в следующей последовательности:

* *Ext JS* 2.0. Библиотека была выпущена 4 декабря 2007 года. В данной версии стандартный пользовательский интерфейс был переделан для создания веб-приложений, максимально приближенных к настольным. Также обновилась документация для библиотеки [31].
* *Ext JS* 3.0. Данная версия библиотеки увидела свет 6 июля 2009 года. Новым стало добавление элементов обеспечивающих коммуникацию пользователей, всплывающие боковые панели, элементы управления флэш-графикой, элементы работы со списками [31].
* *Ext JS* 4.0. Данная версия выпущена 26 апреля 2011 года. Она включает в себя полностью пересмотренную структуру классов, пакетов данных, анимации и рисования [32]. Кроме того переработана типизация, а дополнительные архитектурные возможности позволяют обеспечить проектирование приложений в рамках MVC (Model-View-Controller).
* *Ext JS* 5.0. Данная версия выпущена 2 июня 2014 года [32]. В ней были расширены возможности для создания приложений ориентированных на использование сенсорных экранов [33], расширены возможности обработки больших объемов данных [34], добавлены новые компоненты и проведена оптимизация графического пакета.
* *Ext JS* 6.0. Выпущена 29 марта 2016 года. В данная версия включает фреймворк для мобильных приложений.

Достоинства [35]:

1. большой набор виджетов позволяет в короткие темпы разработать высоко функциональный пользовательский интерфейс;
2. приближенность пользовательского интерфейса к оконному приложению;
3. оптимизация интерфейса для мобильных устройств;
4. организация эффективной работы с большими объемами данных;
5. наличие проработанной документации в том числе и на русском языке.

Недостатки библиотеки заключены в ограниченных возможностях стилевого оформления приложения, а так же в больших размерах самой библиотеки, что делает её не эффективной при создании малых проектов. Так же не мало важным недостатком является лицензия, она позволяет использовать библиотеку только на не коммерческих проектах [35].

# Заключение

В заключении следует отметить, что по результатам проведенных исследований можно сделать вывод о широком распространении *JavaScript* в разработке современных веб-приложений. Постоянное развитие и совершенствование языка позволяет не только решить проблемы вызванные несогласованностью разработчиков браузеров, но и в значительной степени расширить его возможности. Реализации нескольких тенденций в развитии современного программирования позволяет не только использовать преимущества нескольких подходов и опустить недостатки, но и обеспечивает легкий переход программистов с других языков.

Развитие всевозможных фреймворков избавляет программистов от необходимости решения типовых однообразных задач, а так же позволяет использовать многочисленные наработки, не заботясь об совместимости на различных браузерах.

Однако следует внимательно относиться к выборы фреймворка, дабы максимально использовать его возможности. На примере рассмотренных *jQuery*, *Bootstrap* и *Ext JS* можно сделать вывод, что каждый фреймворк не является универсальным. Задачи эффективно решаемые одной библиотекой не будут столь эффективны при решении с испльзованием иной. Таким образом достоинством любого *JS*-фреймворка является наличие механизмов для решения конкретного спектра задач. Так *Bootstrap* и *jQuery* позволяют в короткие сроки разрабатывать такие сайты-визитки, информационные и новостные ресурсы. При этом нет необходимости заботиться о крос-браузерности полученных сайтов. Использование *Ext JS* будет более эффективным на крупных проектах связанных с обработкой и анализом больших объемов данных.

# Список использованной литературы

1. Википедия / JavaScript [Электронный ресурс]. – 2017. – Режим доступа: <https://ru.wikipedia.org/wiki/JavaScript> (дата обращения: 16.05.2017).
2. Lindley, C. JavaScript Enlightenment // Cody Lindley – O’Reilly Media, 2012. ­– 166 p. − ISBN 978-1-4493-4288-3
3. Haverbeke, M. Eloquent JavaScript // Marijn Haverbeke – No Starch Press, 2011. ­– 224 p. − ISBN 978-1-59327-282-1
4. Osmani, A. Developing Backbone.js Application // Addy Osmani –O’Reilly Media, 2013. ­– 373 p. − ISBN 978-1-449-32825-2
5. Hoigaard, E. Smooth CoffeeScript / E. Hoigaard // Smooth CoffeeScript [Электронный ресурс]. − 2017. − Режим доступа: <http://autotelicum.github.io/Smooth-CoffeeScript/> (дата обращения: 30.05.2017).
6. MacCaw, A. The Little Book on CoffeeScript / Alex MacCaw // The Little Book on CoffeeScript[Электронный ресурс]. − 2017. − Режим доступа: <http://arcturo.github.io/library/coffeescript/01_introduction.html> (дата обращения: 16.05.2017).
7. Zakas N. 1. What is JavaScript? // Professional JavaScript for Web Developers. — 2nd ed. — USA, Canada: Wiley Publishing, Inc., 2009. — P. 3. — ISBN 978-0-470-22780-0.
8. Koch P.-P. Chapter 6. BOM // ppk on JavaScript. — 1st ed. — New Riders Press, 2006. — 528 p. — ISBN 0321423305.
9. MDN / Mozila Developer Network [Электронный ресурс]. – 2017. – Режим доступа: <https://developer.mozilla.org/ru/> (дата обращения: 30.05.2017).
10. Zakas N. 10. The Document Object Model // Professional JavaScript for Web Developers. — 2nd ed. — USA, Canada: Wiley Publishing, Inc., 2009. — P. 261 — 317. — ISBN 978-0-470-22780-0.
11. Википедия / Веб-приложение [Электронный ресурс]. – 2017. – Режим доступа: <https://ru.wikipedia.org/wiki/Веб-приложение> (дата обращения: 16.05.2017).
12. Крейн, Д. Ajax в действии // Крейн Д., Паскарелло Э., Джеймс Д. – Вильямс, 2006. ­– 640 с. − ISBN 5-8459-1034-X
13. Википедия / AJAX [Электронный ресурс]. – 2017. – Режим доступа: <https://ru.wikipedia.org/wiki/AJAX> (дата обращения: 16.05.2017).
14. Википедия / Comet [Электронный ресурс]. – 2017. – Режим доступа: <https://ru.wikipedia.org/wiki/Comet_(программирование)> (дата обращения: 16.05.2017).
15. Википедия / WebOS [Электронный ресурс]. – 2017. – Режим доступа: <https://ru.wikipedia.org/wiki/WebOS> (дата обращения: 16.05.2017).
16. Oracle / webnotes [Электронный ресурс]. – 2017. – Режим доступа: <http://www.oracle.com/technetwork/java/javase/webnotes-136672.html> (дата обращения: 16.05.2017).
17. Kiessling, M. Node Beginner / Manuel Kiessling // The Node Beginner Book[Электронный ресурс]. − 2017. − Режим доступа: <http://www.nodebeginner.org/> (дата обращения: 30.05.2017).
18. Holowaychuk, T. Mastering NodeJS / TJ Holowaychuk // Mastering NodeJS Open Source Node eBook[Электронный ресурс]. − 2017. − Режим доступа: <http://visionmedia.github.io/masteringnode/> (дата обращения: 30.05.2017).
19. Young, A. Building A JavaScript Framework / Alex Young // DailyJS - JavaScript Blog [Электронный ресурс]. − 2017. − Режим доступа: <http://dailyjs.com> (дата обращения: 16.05.2017).
20. JavaScritp.ru / Полифилы [Электронный ресурс]. – 2017. – Режим доступа: <https://learn.javascript.ru/dom-polyfill> (дата обращения: 16.05.2017).
21. Murphey, R. jQuery Fundamentals / Rebecca Murphey // DailyJS - jQuery Fundamentals [Электронный ресурс]. − 2017. − Режим доступа: <http://jqfundamentals.com> (дата обращения: 30.05.2017).
22. jquery.page2page.ru / Работа с атрибутами [Электронный ресурс]. – 2016. – Режим доступа: <http://jquery.page2page.ru/index.php5/Работа_с_атрибутами> (дата обращения: 21.05.2017).
23. Бибо, Б. jQuery. Подробное руководство по продвинутому JavaScript, 2-е издание // Бибо Б., Кац И. – Спб.: Символ-плюс, 2011. ­– 623 с. − ISBN 978-5-93286-201-8, 978-1-935182-32-0
24. Бенедетти, Р. Изучаем работу с jQuery // Райан Бенедетти, Ронан Крэнли. – Спб.: Питер , 2012. ­– 508 с. − ISBN 978-5-459-00896-8
25. Bibeault, В. jQuery in Action, Third Edition // Bear Bibeault, Yehuda Katz, and Aurelio De Rosa – Manning, 2015. ­– 504 с. − ISBN 9781617292071
26. Бибо, Б. jQuery. Подробное руководство по продвинутому JavaScript // Бер Бибо, Иегуда Кац — Спб.: Символ-Плюс, 2009. — 384 с. — ISBN 978-5-93286-135-6, 5-93286-135-5, 1-933988-35-5.
27. Википедия / Bootstrap (фреймворк) [Электронный ресурс]. – 2016. – Режим доступа: <https://ru.wikipedia.org/wiki/Bootstrap_(фреймворк)> (дата обращения: 21.05.2017).
28. blog.twitter.com / Bootstrap from Twitter [Электронный ресурс]. – 2017. – Режим доступа: <https://blog.twitter.com/2011/bootstrap-from-twitter> (дата обращения: 21.05.2017.
29. The Bootstrap Blog / Bootstrap 3 released [Электронный ресурс]. – 2017. – Режим доступа: <http://blog.getbootstrap.com/2013/08/19/bootstrap-3-released/> (дата обращения: 21.05.2017).
30. The Bootstrap Blog / Bootstrap 4 alpha [Электронный ресурс]. – 2017. – Режим доступа: <http://blog.getbootstrap.com/2015/08/19/bootstrap-4-alpha/> (дата обращения: 21.05.2017).
31. Ext JS Documentation / History [Электронный ресурс]. – 2017. – Режим доступа: <http://docs.sencha.com> (дата обращения: 21.05.2017).
32. Ext JS Documentation / Blog [Электронный ресурс]. – 2017. – Режим доступа: <http://docs.sencha.com/blog> (дата обращения: 21.05.2017).
33. eWeek / Sencha Ext JS 5 Unifies Mobile, Desktop App Dev [Электронный ресурс]. – 2017. – Режим доступа: <http://www.eweek.com/blogs/first-read/sencha-ext-js-5-unifies-mobile-desktop-app-dev> (дата обращения: 21.05.2017).
34. Dr.Doob’s / Sencha Ext JS 5 Streamlines and Unifies [Электронный ресурс]. – 2017. – Режим доступа: <http://www.drdobbs.com/web-development/sencha-ext-js-5-streamlines-and-unifies/240168397> (дата обращения: 21.05.2017).
35. Garcia, J. Ext JS in Action // Jesus Garcia. – Manning Publication , 2011. ­– 495 p. − ISBN 9781935182115