ИНФОРМАТИКА

COMPUTER SCIENCE

МРНТИ 50.05.19
УДК 004.416.6

10.51889/2959-5894.2023.82.2.019

Г.О. Исакова[1], Р.З. Сулейменова[1], Б.Е. Таныкпаева[1], Г.Р. Есенбаева[1], А.Ж. Тулкибаев[1]*

[1]С. Сейфуллин атындағы Қазақ агротехникалық университеті, Астана қ., Қазақстан
[*]e-mail: issakova.g@rambler.ru

БАЗАЛЫҚ ИНТЕРАКТИВТІ ОБЪЕКТІЛЕРІ БАР ДИНАМИКАЛЫҚ САЙТТАРДЫ БАҒДАРЛАМАЛАУ ЕРЕКШЕЛІКТЕРІ

Аңдатпа

Интерактивті элементтері -Web-беттің жеке құрылымдық блоктары арасындағы әртүрлі түймелер-ауысулар, бейнеклиптер мен аудио фрагменттерді іске қосу түймелері, құрылым элементтері бойынша өтуге мүмкіндік беретін сілтемелер, бір ресурстың шекарасын кеңейтетін пішін болып табылатын динамикалық сайттарсыз заманауи Интернетті елестету қиын Осы секілді сипатталғанға жақын сайттарды CSS стильдері арқылы немесе CMS жүйесі бойынша жасауға болады. Алайда, жоспарланғандай сайтты құруға бағдарламалауды қажет етеді. Мақалада негізгі интерактивті элементтері бар динамикалық сайттарды бағдарламалау ерекшелігі ашылады, ал бұл өз кезегінде жаңа динамикалық сайтты жазуға ғана емес, сонымен бірге HTML көмегімен жүзеге асырылған ескі сайтты жаңартуға мүмкіндік береді. Негізгі идея - сайт беттерін пішімдеу және оның дизайны, электрондық құрылғының экранына байланысты сайттың көрінісін өзгерту үшін CSS элементтерімен бірге JavaScript және HTML бағдарламалау тілдерінің негізгі мүмкіндіктерін пайдалана отырып, объектіге бағытталған бағдарламалауға жүйелік тәсілдеу болып табылады. Нәтижесінде, веб-беттерді бағдарламалаудың ұсынылған ерекшеліктері ұсынылған CMS жүйелеріне тәуелсіз құрылған заманауи динамикалық сайттарды құрудың негізгі тәсілін өзгертуге және бұрыннан бар HTML сайттарының мүмкіндіктерін едәуір кеңейтуге мүмкіндік береді.

Түйін сөздер: динамикалық сайт, элемент, шаблон, объект, контейнер, блок, браузер, бағдарламалық код, атрибут.

Аннотация

Г.О. Исакова[1], Р.З. Сулейменова[1], Б.Е. Таныкпаева[1], Г.Р. Есенбаева[1], А.Ж. Тулкибаев[1]
[1]Казахский агротехнический университет имени С. Сейфуллина, г.Астана, Казахстан

СПЕЦИФИКА ПРОГРАММИРОВАНИЯ ДИНАМИЧЕСКИХ САЙТОВ С БАЗОВЫМИ ИНТЕРАКТИВНЫМИ ОБЪЕКТАМИ

Современный Интернет сложно представить без динамических сайтов с интерактивными элементами – различными кнопками-переходами между отдельными структурными блоками Web-страницы, кнопками запуска видеоклипов и аудиофрагментов, ссылок, позволяющих переходить по элементам структуры, элементов форм, которые расширяют границы одного ресурса. Сайты, близкие к описанным, можно разработать с помощью стилей CSS или разработать под системой CMS. Однако получить сайт в соответствии с поставленными задачами позволяет именно программирование. В работе раскрывается специфика программирования динамических сайтов с базовыми интерактивными элементами, позволяющая не только написать новый динамический сайт, но и произвести модернизацию старого сайта, реализованного с помощью HTML без полного переписывания последнего. Основная идея заключается в системном подходе к объектно-ориентированному программированию с использованием фундаментальных возможностей языков программирования JavaScript и HTML наряду с элементами CSS для форматирования страниц сайта, его оформления, изменения представления сайта в зависимости от экрана электронного устройства. В итоге, предложенные особенности программирования Web-страниц дают возможность изменить базовый подход к созданию современных динамических сайтов, созданных независимо от предлагаемых CMS-систем и значительно расширить возможности уже существующих HTML-сайтов.

Ключевые слова: динамический сайт, элемент, шаблон, объект, контейнер, блок, браузер, программный код, атрибут.

Abstract

SPECIFICS OF PROGRAMMING DYNAMIC SITES WITH BASIC INTERACTIVE OBJECTS

Issakova G.O.[1], Suleimenova R.Z.[1], TanykpayevaB.Ye.[1], Yessenbayev G.R.[1], Tulkibayev A.Zh.[1]

[1]Kazakh Agrotechnical University named after S. Seifullin, Astana, Kazakhstan

It is difficult to imagine the modern Internet without dynamic websites with interactive elements. These elements are various buttons for transitions between individual structural blocks of a Web page, buttons for launching video clips and audio fragments, links that allow you to navigate through the elements of the structure, form elements that expand the boundaries of a single resource. Sites close to those described can be developed using CSS styles or developed under a CMS system. However, it is programming that allows to get a website for a reason. The paper reveals the specifics of programming dynamic sites with basic interactive elements, which allows not only to write a new dynamic site, but also to modernize the old site implemented using HTML without completely rewriting the latter. The main idea is a systematic approach to object-oriented programming using the fundamental capabilities of the JavaScript and HTML programming languages along with CSS elements for formatting site pages, its design, changing the presentation of the site depending on the screen of an electronic device. As a result, the proposed programming features of Web pages make it possible to change the basic approach to creating modern dynamic sites created independently of the proposed CMS systems and significantly expand the capabilities of existing HTML sites.

Keywords: dynamic web page, element, pattern, object, container, block, browser, program code, attribute.

Введение

Динамический сайт сегодня является основой Интернета [1]. В настоящее время большинство сайтов имеют множество движущихся элементов или фрагментов анимации. Не все из обозначенного является декоративным оформлением. Следует отметить, что при программировании динамических сайтов активно используются интерактивные элементы [2], то есть, такие элементы, которые реагируют на действие пользователя (кнопки, ссылки, элементы форм и другое). Особую актуальность сайты с таким наполнение получили в период локдауна, связанного с эпидемией COVID-19 и перехода систем и служб в дистанционный режим работы. Динамический сайт с базовыми интерактивными объектами стал востребован в системе образования и науки. Всевозможные тесты, контрольные и лабораторные работы, семинары и конференции, мастер-классы и даже обыкновенные уроки в дистанционном режиме требуют активного динамического представления [3]. При этом подобные сайты должны легко разворачиваться даже при слабом сигнале Wi-Fi, давать полноценный доступ к ресурсу на устройствах с разным разрешением экрана, платформах, ориентациях экрана [1]. Не меньше требований выдвигается к сайтам предоставления административных и медицинских услуг в режиме он-лайн, интернет-площадок торговли, различных развлекательных и игровых порталах [4].

Но все же, в рамках данного исследования, наибольшую актуальность представляют именно динамические сайты с интерактивными объектами в сфере образования, поскольку такие сайты должны полноценно выполнять образовательную функцию (информационное содержание), при этом легко открываться на электронных устройствах с различными характеристиками. Цель данной статьи – раскрыть специфику программирования динамических сайтов с базовыми интерактивными элементами, необходимыми для расширения возможностей изменения дизайна в зависимости от поведения пользователя, размера экрана, платформы, ориентации экрана электронного устройства. В целом это позволит вывести дистанционное и смешанное образование на качественно новый уровень, позволяющий формировать гармонично развитую личность не зависимо от возможности очного посещения образовательного учреждения.

Гипотеза исследования состоит в том, что качественно созданный динамический сайт с интерактивными объектами позволяет представить больше структурированной информации по сравнению с обычными гипертекстовыми сайтами.

Методология исследования

Исследование в рамках данной работы базировалось на системном подходе с использованием теоретических и эмпирических методов исследования, в частности – наблюдения, эксперимента и сравнения. Наблюдение базировалось на следующих фактах: можно разделить существующие электронные устройства на ряд категорий и программировать сайты для отдельных категорий устройств. Но это займет много времени и, кроме того, с каждым годом появляется все большее количество разнообразных устройств, подключаемых к сети Интернет. Даже в настоящее время имеется огромный спектр устройств, включая бытовые приборы, которые взаимодействуют с Интернетом в рамках «Интернета вещей» [5].

Эксперимент состоял в поиске подходов к объектно-ориентированному программированию с исследованием фундаментальных возможностей языков программирования JavaScript и HTML наряду с использованием элементов CSS для форматирования страниц сайта, его оформления, изменения представления сайта в зависимости от экрана электронного устройства. Сравнение основано на проведении детального анализа в отношении полноты и объема предоставляемой информации на сайте с обычными гипертекстом и сайте, разработанном с использованием предлагаемой технологии. Здесь следует отметить, что динамические сайты под CMS (Система управления содержимым) в данной работе не рассматривались по причине того, что данные системы позволяют создавать сайт и управлять его содержимым, однако представлены готовыми шаблонами. Обычно содержимое рассматривается как неструктурированные данные в рамках поставленной задачи, в отличии от систем с базами данных. Шаблоны CMS могут иметь возможность редактирования под некоторые требования пользователя (бесплатные версии), однако более функциональные шаблоны, как правило, имеют закрытый код и используются на коммерческой основе. Любой шаблон является готовым блоковым решением и сложно добиться полного соответствия поставленным задачам от разработчика сайта.

Проверка гипотезы исследования строилась на положении, что при открытии Web-страницы в браузере, браузер строит определенную модель документа в виде контейнеров [6]. Эти контейнеры вложены один в другой, содержат теги или текст. Каждый контейнер представлен своим объектом – в зависимости от того, какой тег или текст он представляет. То есть – объектную модель документа (DocumentObjectModel). И к каждому объекту можно получить доступ через переменную document, благодаря свойству «documentElement» ссылаться на свой объект, представленный тегом. Здесь же представлены свойства заголовка и тела Web-страницы, где содержаться объекты этих элементов.

Понимая эту структуру Web-странницы, далее можно рассматривать переходы между контейнерами, как ответвления, а точки ответвления – как базовые узлы страницы. И именно операции перемещения между базовыми узлами следует рассматривать как точку применения интерактивного объекта для перехода между контейнерами. А чтобы при таких операциях не утратить структуру страницы, следует присвоить отдельным контейнерам специальные CSS стили [7], которые помогут адаптировать сайт под возможности техники и браузера.

Результаты исследования

Результаты теоретического этапа исследования.

Разработка динамических сайтов представляет интерес для исследователей Web-программирования примерно с 1994 года. Уже тогда был задан вопрос – как донести пользователю больше содержательной и интересной информации, доступной для отображения на различных устройствах [8]. До той поры, пока разработка указанных сайтов была только в поле зрения ученых, подобные до [8] публикации были часты и типичны. С точки зрения современных информационных технологий, подобные материалы были больше философскими, нежели научными и представляли интерес для широкого круга исследователей, интересующихся перспективными направлениями науки. Постепенно все больше пользователей присоединялись к сети. С совершенствованием HTML, сайты начали обретать более современный дизайн и постепенно становиться главными носителями информации, по сравнению с радио и телевидением. Темп программирования динамических сайтов приобрела коммерческую основу [9] и стала менее доступной для широкого круга исследователей, поскольку многие аспекты разработки защищались коммерческой тайной [6].

В настоящее время большинство литературы по Web-разработке динамических сайтов представлено базовыми основами программирования с применением различных языков, использованием стилей CSS [6 – 7], их усовершенствования и непосредственно методике обучения Web-программированию [1]. Непосредственно в Интернете размещено много материалов по обозначенной теме. Возможно, они не представляют научную, однако дают значительную практическую ценность – позволяют отследить над какими задачами трудятся современные Web-разработчики. В частности, в Wix-блоге [10], площадке дискуссий известного конструктора сайтов, постоянно возникает дискуссия о преимуществах динамических сайтов над статическими. Одним из аргументов в пользу динамических сайтов выдвигается их адаптация под разные браузеры и устройства, а вот как негатив – отмечается сложность разработки таких сайтов.

Анализируя источники [2, 6, 7], можно отметить, что просто гибкий сайт не решает всех поставленных вопросов в отношении адаптации под браузер и устройство. И одним из «минусов» гибкости являются некоторые трудности для пользователя при работе с таким сайтом. Например,

работа на одном устройстве, а, затем, переход на другое устройство, может замедлить работу из-за изменения расположения отдельных элементов на экране. Ряд картинок на портативных устройствах могут быть в усеченном варианте или представлены фрагментом. Непосредственно для программиста могут быть трудны и отнимают много времени работы над изменением иллюстраций (их резка, разделение, а затем прописывание отдельных медиатипов в коде). Хотя с усовершенствованием стилей CSS эта задача стала немного проще [7]. Пожалуй, в рамках данного исследования, наиболее близкой к подтверждению установленной гипотезы является работа [11], где начало работы над сайтом приравнивается к чистому холсту, на котором рисуется структура будущей Web-страницы, а затем вводятся ограничения, коими и являются контейнеры. Далее же следует расширить возможности самих контейнеров, дополнить их элементами, облегчающими конструкцию Web-страницы, но расширяющими возможности подачи и объема информации [12]. То есть, разработка динамического сайта – использование разнообразных слоев, картинок, умной разметки. Именно слои и являются основой динамического сайта, а никак не гибкие картинки. Как отмечается далее в указанной работе [11], с возможностями прятать и показывать элементы, изменять размеры картинок, и многое другое, только дизайнерскими решениями, без основ программирования, была найдена возможность адаптировать Web-страницы к различным возможностям устройств и экранов [13]. Но именно программирование динамического сайта, с изменении свойств различных объектов, гибкими контейнерами, которые содержат свои элементы с изменяемыми параметрами, позволяет представить пользователю больше структурированной информации, которую, благодаря интерактивным объектам, он может фильтровать под свои потребности.

Результаты практического этапа исследования.

Выявление специфики программирования динамических сайтов с базовыми интерактивными объектами потребовало проведения ряда экспериментов. В частности – это задачи с использованием адаптивных приемов в программировании Web-сайтов и изменении свойств различных объектов в зависимости от ориентации экрана. Приведем примеры таких задач. Для этого используем макет, заданный нижеследующим кодом, который описывает стили Web-документа. Но при этом представим развертку экрана в пикселях (px) и процентах (рис. 1).

```
.page {
      margin: 36px auto;
      width: 960px;     == > 90% //экспериментально, оптимально для развертки
экрана
}
.blog {
margin: 0 auto 53px;
width: 900px;     ⇒ 93,75%        /* 900px / 960px */
}
.blog.main {
float: left;
width: 566px;     ⇒ 62.8888889% /* 566 / 900*/
}
.blog.other {
float: right;
      width: 331px;     => 36.8888889%          /* 331 / 900 */
}
```

Рисунок 1. Листинг кода с процентным представлением размера контейнера

Таким образом, произошло изменение контейнера, представленного 960 px на 90%. Представленный макет страницы сайта перестает быть фиксированным, поскольку контейнер изменяется в размерах при каком-либо изменении окна браузера. Но класс «.blog» в макете все еще равен 900 px. Для изменения размеров следует определить пропорции изменения с использованием данных о величине контейнера по макету страницы:

$$900/960 = 0{,}9375 * 100 = 93.75\%.$$

Из листинга кода следует, что макет представлен также двумя колонками – левой и правой. Это блоки, которые находятся внутри блока «.blog» и при изменении этого блока должны также изменять

свои размеры пропорционально. Для этого необходимо пересчитать процентное соотношение изменения блоков «main» и «other» по величине блока «.blog», что составит 62,8% и 36,8% соответственно. В итоге проведенных изменений получается гибкий макет, в процентном соотношении полностью повторяющий первоначальный дизайн Web-страницы. Учитывая изложенное, можно вывести три правила:

а) если используются гибкие отступы (margin) для одного элемента, следует за контекст применять ширину контейнера этого элемента;

б) если используются гибкие поля (padding) для всего документа, контекстом для исчислений становится ширина самого элемента;

в) родительский блок (width), в котором размещаем различные элементы и блоки – определяется в процентах от величины контейнера.

Данное правило важно, когда на сайте размещаются базовые интерактивные объекты, которые должны быть видимыми для пользователя при любом изменении размера экрана или окна браузера, поскольку позволяют получить доступ к полному материалу (информации), представленному на Web-странице. По умолчанию ширина/высота блока определяется шириной/высотой контента, значений отступов, полей и границ. Свойство со значением border-box позволяет сделать так, чтобы ширина и высота базового элемента исчислялась с учетом границ и отступов. Это важно для случая, когда Web-страницу с одинаковой вероятностью будут просматривать как со смартфона, так и с других устройств, вплоть до *Smart*-телевизора. Данное свойство позволяет получить идентичное изображение на всех устройствах, а интерактивные элементы сайта будут находиться в зоне просмотра и работы.

Например, рассмотрим блок «width: 200px; padding: 0 20px». Итоговая ширина равна 220px. Но с применением свойства box-sizing: border-box итоговая ширина будет равна 200px. То есть:

width: 160px + padding: 40px (слева и справа по 20px).

Естественно, становится интересно, как при таких манипуляциях буду вести себя встроенные в сайт картинки, небольшие видео, другой мультимедийный контент. Современные браузеры пропорционально изменяют размеры картинок. Если гибкий контейнер будет изменять свои размеры, то изменятся и размеры указанного контента с изменением пропорций сторон. Но следует при программировании динамических сайтов учесть необходимость задавания тегу img свойство max-width: 100%. Именно это свойство запрещает изображению превышать ширину контейнера.

Это свойство можно применять для тегов, приведенных на иллюстрации (рис. 2).

```
img, embed, object, video {
  max-width: 100 %;
}
```

Рисунок 2. Правило «запрета» превышения ширины контейнера для ряда тегов

Следует отметить, что для создания гибкого фонового изображения можно использовать свойство background-size: cover. Однако это свойство следует использовать с осторожностью – некоторые виды браузеров, особенно устаревшие версии, будут немного искажать изображение, слишком его растягивая или сужая.

Ранее указывалось, что можно разработать сайт для конкретного типа устройств. Позже в CSS [7] была предусмотрена возможность использования различных медиатипов, которые позволяют проектировать дизайн для отдельного браузера или устройства. Эту идею можно использовать для расширения возможностей при программировании динамических сайтов с базовыми интерактивными элементами. В частности, использование медиатипов позволит уменьшить количество подаваемого на экран текста и заменить этот текст другими видами представления информации. Например, представить текст Web-страницы на экране смартфона интерактивной инфографикой, видео, аудиофайлом. При этом в браузере, развернутом на обычном настольном компьютере, будет отражаться именно полнотекстовая версия существующего сайта, игнорируя медиатип предоставления видео или отображения текста вслух.

Для выполнения этой задачи следует прописать соответствующий носитель:

– «handheld» смартфоны;

– «projection» проекторы;
– «screen» экран монитора;
– «speech» отображение текста вслух;
– «tty» устройства с ограничениями дисплея;
– «tv» телевизоры.

Следует подчеркнуть, что медиатип «all» используется по умолчанию для отображения на всех видах устройств. Для спецификации вида устройства под отображение определенного контента необходимо произвести изменение атрибута ссылки на это устройство. Пример листинга кода для этой операции представлен на рис. 3.

```
<link rel="stylesheet" href="global.css" media="all" />
<link rel="stylesheet" href="main.css" media="screen" />
<link rel="stylesheet" href="paper.css" media="print" />
```

Рисунок 3. Пример изменения атрибута ссылки на отображение по видам устройств

Однако такие изменения требуют обязательного создания блока @media с привязкой до конкретного медиатипа (рис. 4).

```
@media screen {
  body {
  font-size: 100 %;
}
@media print {
body {
font-size: 15 pt;
```

Рисунок 4. Листинг-код привязки медиатипа

Значение свойств можно изменять постепенно. Это возможно с использованием, опять таки, CSS, или transitions. В отличии, например, от анимации, где можно управлять любым количеством промежуточных состояний, с помощью transitions можно управлять только двумя состояниями – начальным и конечным. Плавный переход в CSS осуществляется с помощью установления свойства длительности перехода transition-duration в секундах, долях секунды или миллисекундах. Но такие переходы в CSS возможны не всегда. В основном их возможно применить при изменении размера, цвета, позиции базовых объектов. Расширить этот перечень можно с помощью создания меню с плавными переходами, фрагмент кода которого представлен на рис. 5.

```
.breadcrumbs a {
      width: 150px;
      line-height: 37px;
      display: block;
      position: relative;
      color: black;
      font-size: 15px;
      padding-left: 10px;
      text-decoration: none;
      -webkit-transition: all 0.5s;
      transition: all 0.5s;
      background: white;
}
```

Рисунок 5. Фрагмент кода меню с плавными переходами

Выполняя описанные манипуляции по программированию, фактически проводится изменение функций document.forms, document.getElementById, document.createElement и некоторых других, встроенных в объект document. Именно здесь имеется возможность изменения в реальном времени не только размеров, цвета, но и добавлять и изменять какие-либо интерактивные объекты.

Например, с помощью простого фрагмента кода JavaScript можно получить эффект изменения изображения базового интерактивного элемента «кнопка» (рис. 6).

```
<img src="button_off.gif"
   onmouseover="this.src='button_over.gif';"
   onmousedown="this.src='button_down.gif';"
   onmouseout ="this.src='button_off.gif';"
   onmouseup ="this.src='button_over.gif';">
```

Рисунок 6. Фрагмент кода изменения изображения

В представленном коде присутствуют четыре события изображения: onmouseover, onmousedown, onmouseout і onmouseup. Каждое событие фрагментом кода JavaScript изменяет атрибут src изображения, приводя его в соответствии с первоначальной идеей разработки динамического сайта с базовыми интерактивными объектами.

Дискуссия

Описанная в работе специфика показывает, что создание динамического сайта с базовыми интерактивными объектами может привести к интересным результатам, когда при уменьшении размеров предоставляемого контента сайта объем информации может увеличиваться на 36 – 90%. При этом, в отличие от указанных в [1] сложностей при разработке или манипуляций с дизайном [11], в предложенном решении используется чистый код JavaScript, включая изменения контейнеров стиля CSS для создания эластичной структуры Web-страницы.

Анализируя сеть Интернет можно отметить, что в настоящее время большинство сайтов являются динамичными под управлением CMS. Хотя CMS не конструктор, а именно система, имеющаяся оболочка управления позволяет получить некоторый установленный функционал с возможностью расширения. Разработка динамических сайтов под CMS требует больше трудозатрат при установке системы и разработке сайта, однако имеет такие преимущества, как большой функционал, не требующий программирования, простота наполнения и управления сайтом. Даже изменение дизайна такого сайта не требует значительных знаний Web-программирования. Структура сайта, добавление и удаление страниц, новый шаблон можно изменить за несколько минут. Однако тот же HTML-сайт при постановке таких задач приходится переписывать с нуля [13].

Но предложенное в работе решение позволяет преобразовать тот же HTML-сайт без потери информации и значительных трудозатрат. Акцентируя внимание на [12], последнее утверждение не является новым, если не учитывать тот факт, что предлагается изменять не только шаблон, делая его гибким, но и коренным образом перестраивать структуру сайта, структурируя информацию и выдавая ее в зависимости от заложенных параметров демонстрации на различных устройствах и экранах.

То есть, создание динамического сайта под CMS – это просто и доступно в большинстве случаев. Однако, при использовании таких сайтов для поддержки учебного процесса [3], особенно проводимого в дистанционном режиме, часто необходимо применение значительного числа интерактивных объектов, позволяющих приблизить виртуальный урок максимально к требованиям очного образования. Здесь важными становятся разнообразные переходы между отдельными контейнерами, в которых содержится материал, с четким ограничением по времени, позволяющим выстроить все объекты в часовом отрезке стандартного урока. Именно поэтому динамические сайты под CMS в дистанционном образовании не получили значительной поддержки.

Создание HTML-сайта с оперирование только стилями CSS [1, 7] также не позволяет полностью решить поставленную задачу. С помощью CSS можно создать гибкий сайт, где все объекты будут изменять свои размеры в зависимости от величины и ориентации экрана электронного устройства [5]. Как утверждает [11], с этим можно справиться с помощью грамотного объединения адаптивных слоев, картинок и разметки. Но при этом автор подчеркивает, что адаптивные слои являются частой практикой, а вот с адаптивными картинками немного сложнее – здесь нужна кропотливая работа дизайнеров, чтобы передать изображение без искажений в измененном формате.

Для этого существует определенная техника, над совершенствованием которой работают программисты, исследователи и дизайнеры.

Особенность предлагаемого в данной работе решения – изменение контейнеров с одновременными изменениями содержимого, то есть, тех же рисунков. Для Web-программиста это несколько добавляет работы: необходимо прописать изменения позиций иллюстрации. Но одновременно это действие значительно «облегчает» сайт, поскольку не нужно сберегать измененные изображения для разных экранов, не требует адаптации остальных элементов дизайна (резка, обработка, разделение или объединение).

В целом, предложенное в работе решение не является конечным и априорным. Данная разработка показывает один из аспектов возможного подхода к программированию динамичных сайтов с гибкой структурой документа на основе HTML, со стилями, позволяющими изменять внешний вид этого документа вместе с изображениями и определёнными объектами, которые могут как давать гипертекстовые переходы, подключать аудио и видеофайлы, так и с помощью скриптов добавлять прочие эффекты. Скрипты могут быть созданы с помощью объектно-ориентированного языка программирования JavaScript, подключаются также, как и стили и могут быть написаны или внутри Web-страницы, или подключены, как внешние файлы. Хотя и в данном случае, часть возможностей JavaScript может быть реализована с помощью стилей CSS.

Заключение

Проведенное исследование позволяет сделать ряд выводов в отношении Web-программирования динамических сайтов с базовыми интерактивными элементами.

Во-первых, при открытии сайта в браузере пользователь видит текст, а для программиста этот текст представлен как модель документа в виде контейнеров. Эти контейнеры и являются основой программирования для адекватного представления сайта в разных браузерах, под разными операционными системами и экранами.

Во-вторых, программирование динамического сайта требует системного подхода, когда производится задание параметров изменений от контейнеров, до полей, отступов и объектов, увязывая все размеры в определенные пропорции изменений.

В-третьих, объект document сайта дополняется скриптами, позволяющими создавать не только гибкость конструкции в целом, но и вносить изменения подачи информации в зависимости от вида электронного устройства, на котором эта информация будет считываться.

В целом необходимо отметить, что представленная специфика программирования динамических сайтов дает возможность изменить базовый подход к созданию современных сайтов, отойти от использования сайтов под CMS для обеспечения образовательной деятельности и значительно расширить возможности существующих HTML-сайтов.

Список использованных источников:

1 Nixon Robin (2018). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5 (Learning PHP, MYSQL, Javascript, CSS & HTML5) 5th Edition. O'Reilly Media. 832. ISBN: 978-1491978917

2 Wiberg Mikael. (2010) Interaction per se: understanding "the ambience of interaction" as manifested and situated in everyday & ubiquitous IT-use // International Journal of Ambient Computing and Intelligence.—Vol. 2, no. 2.

3 Ровина Е. Е. Интерактивные элементы на учебных занятиях: несколько увлекательных идей и примеры применения / Образование и право. – № 1. – 2021. – сс. 224 – 227. DOI: 10.24411/2076-1503-2021-00041

4 Liang H.-N.; Parsons P.; Wu H.-C.; Sedig K. (2010). "An exploratory study of interactivity in visualization tools: 'Flow' of interaction" (PDF). Journal of Interactive Learning Research. 21 (1): 5 – 45. https://www.learntechlib.org/primary/p/32419/

5 Raafat Aburukba, A. R. Al-Ali, Nourhan Kandil, Diala AbuDamis. Configurable ZigBee-based control system for people with multiple disabilities in smart homes // 2016 International Conference on Industrial Informatics and Computer Systems (CIICS). – IEEE, 2016-03. DOI:10.1109/ICCSII.2016.7462435

6 Нильсен Я., Лоранжер Х. Web-дизайн: удобство использования Web-сайтов. – М.: «Вильямс», 2007. – 496 с. – ISBN 0-321-35031-6.

7 Макфарланд Д. - С. Новая большая книга CSS.– Санкт-Петербург: Питер, 2017. – 720 с. – ISBN 978-5-496-02080-0.

8 Schwabe D., Rossi G. The Object-Oriented Hypermedia Design Model. Communications of the ACM, 38(8): 45 – 46. 1995.

9 Nelson A., Nelson W. (2002). Building Electronic Commerce with Web Database Constructions. Addison Wesley. ISBN 9780201741308.

10 Static vs Dynamic Websites: The Differences, Advantages and Which to Use. Nov 29, 2021. https://www.wix.com/blog/2021/11/static-vs-dynamic-website/

11 Маркотт И. Отзывчивый веб-дизайн. – М.: манн, Иванов и Фебер. – 2012. – 176 с. – ISBN 978-5-91657-385-5.

12 Вора Паван. Шаблоны проектирования веб-приложений // [пер. с англ. М.А. Райтмана]. – М.: Эксмо, 2012. – 576 с. https://monster-book.com/shablony-proektirovaniya-veb-prilozheniy

13 Браун Итан. Изучаем JavaScript. Руководство по созданию современных веб-сайтов, 3-е изд.: Пер. с англ. – СПб.: ООО «Альфа-книга», 2017. – 368 с.

References:

1 Nixon R. (2018) Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5 (Learning PHP, MYSQL, Javascript, CSS & HTML5) 5th Edition. O'Reilly Media. ISBN: 978-1491978917

2 Wiberg M. (2010) Interaction per se: understanding "the ambience of interaction" as manifested and situated in everyday & ubiquitous IT-use. International Journal of Ambient Computing and Intelligence, Vol. 2(2), 1-26.

3 Rovina E. E. (2021) Interaktivnye elementy na uchebnyh zanyatiyah: neskolko uvlekatelnyh idej i primery primeneniya [Interactive elements in learning lessons: Some fascinating ideas and examples of application]. Obrazovanie i Pravo, No. 1, 224-227. (in Russian). DOI: 10.24411/2076-1503-2021-00041 (In Russian)

4 Liang H. N., Parsons P. C., Wu H. C., Sedig K. (2010) An exploratory study of interactivity in visualization tools: 'Flow' of interaction. Journal of Interactive Learning Research. Vol. 21 (1), 5 – 45. https://www.learntechlib.org/primary/p/32419/

5 Aburukba R., Al-Ali A. R., Kandil N., AbuDamis D. (2016) Configurable ZigBee-based control system for people with multiple disabilities in smart homes. In 2016 International Conference on Industrial Informatics and Computer Systems (CIICS). IEEE. DOI:10.1109/ICCSII.2016.7462435

6 Nilsen YA., Loranzher H. (2007) Veb-dizajn: udobstvo ispolzovaniya Web-sajtov [Web design. Ease of use of Web -sites]. M.: Vilyams. ISBN 0-321-35031-6. (in Russian)

7 Makfarland D. S. (2017) Novaya bolshaya kniga CSS [The New Big Book of CSS]. St. P.: Piter. ISBN 978-5-496-02080-0. (in Russian)

8 Schwabe D., Rossi G. (1995) The Object-Oriented Hypermedia Design Model. Communications of the ACM, Vol. 38(8), 45 – 46.

9 Nelson A., Nelson W. (2002) Building Electronic Commerce with Web Database Constructions. Addison Wesley. ISBN 9780201741308.

10 Static vs Dynamic Websites: The Differences, Advantages and Which to Use. Nov 29, 2021. https://www.wix.com/blog/2021/11/static-vs-dynamic-website/

11 Markott, I. (2012) Otzyvchivyj Web-dizajn [Responsive Web Design]. M.: Ivanov i Feber. ISBN 978-5-91657-385-5 (in Russian)

12 Vora, P. (2011). Shablony proektirovaniya veb-prilozhenij [Web Application Design Patterns]. M.: Eksmo. https://monster-book.com/shablony-proektirovaniya-veb-prilozheniy (in Russian)

13 Braun I. (2017) Izuchaem JavaScript. Rukovodstvo po sozdaniyu sovremennyh veb-sajtov [Learning JavaScript. A guide to creating modern Web-sites], 3th Edition. St. P.: Alfa-kniga. (in Russian)