2.  Среда разработки

2.1 HTML

Язык разметки гипертекстовых страниц (HTML – Hypertext Markup Language) представляет собой язык, разработанный специально для создания Web-документов. Он определяет синтаксис и размещение специальных инструкций (тегов), которые не выводятся на экран, но указывают браузеру, как отображать содержимое документа. Он также используется для создания ссылок на другие документы, локальные или сетевые, например, находящиеся в сети Интернет.

Стандарт HTML и другие стандарты для Web разработаны под руководством консорциума W3C (World Wide Web Consortium). В настоящее время действует спецификация HTML 5.0, поддержка которой со стороны всех браузеров уже реализована.

На практике на стандарт HTML большое влияние оказывает наличие тегов, предложенных и поддерживаемых наиболее известными браузерами, такими как Microsoft Internet Explorer и Netscape Navigator.

HTML является сейчас основой любой, даже самой продвинутой, технологии в Интернете.

Годом рождения данного языка принято считать 1986, когда Международная организация по стандартизации (ISO) был принят стандарт, названный SGML (Standard Generalized Markup Language). Данный язык описывали как язык для логического оформления текста, тогда не подразумевали описание внешнего вида документа. SGML даже не представлял собой готовую систему для разметки текста, он не предполагал какого-либо списка структурных элементов, которые применялись бы в определённых ситуациях. Данный язык подразумевал лишь описание синтаксиса написания основных элементов разметки текста (которые сейчас называют тегами). Это основные причины того, что язык SGML не получил значимого распространения.

Интерес к данному языку проявили в 1991 году, когда Европейский институт физики частиц ощутил потребность в передаче гипертекстовой информации по сети Интернет. В том году на основе SGML был создан новый язык — HTML (Hyper Text Markup Language). В HTML версии 1.2 присутствовало около 40 тегов, но, как и в случае с SGML, подразумевалось только логическая разметка текста. Только несколько тегов подразумевали физическую разметку веб-страниц (эти теги не рекомендовались к использованию).

В 1994 году консорциум W3C начал работу над HTML 2.0, окончательный стандарт которой был принят в 1995 году, когда уже вовсю шла разработка HTML 3.0. В HTML 2.0 появились формы — средства отправки информации от пользователя на сервер. Самым большим прорывом явился HTML 3.0. Он включил в себя такие нововведения, как теги для создания таблиц, вставки рисунков, примечаний, разметку математических формул и др. Наиболее важным нововведением явились CSS (таблицы стилей), которые позволяли задавать единый вид всему сайту. Они имели свою структуру, свой синтаксис и применялись вместе с HTML.

В 1995 году появился браузер Netscape Navigator, разработчикам которого была корпорация Netscape Communications. Эта корпорация с целью привлечь пользователей стала вводить новые теги, которые были предназначены на улучшение внешнего вида документов. Тем самым были разрушены изначальные принципы языка.

В 1996 году в свет вышел браузер Internet Explorer 2.0 (разработчик: корпорация Microsoft), но он не имел большой популярности ввиду того, что не поддерживал многих тегов, введённых Netscape. Их поддержка была введена в Internet Explorer 3.0, после выхода которого рынок браузеров был поделён пополам между Microsoft и Netscape.

В скором времени был создан HTML 3.2, который был ориентирован на Internet Explorer. Данный стандарт долгое время оставался единственным для веб-разработчиков. В 2004 году в свет вышел HTML 4.01, обеспечивающий достаточно большую кросс-браузерность. Данный стандарт и CSS сейчас применяются при создании большинства сайтов.

Форматирование документа на языке HTML задается специальными дескрипторами. Дескриптором называется команда форматирования данных и заключена эта команда в угловые скобки «<» и «>». Существуют открывающие и закрывающие дескрипторы, между которыми размещается текст, подлежащий форматированию. Открывающие дескрипторы задают способ форматирования, вторые его отменяют. Разница между такими дескрипторами заключается в том, что в закрывающем дескрипторе перед именем стоит косая черта. Например, дескрипторы <P> и </P>. Существуют также дескрипторы, которые не требуют закрывающего варианта.

Язык HTML не чувствителен к регистру букв, поэтому все дескрипторы можно задавать как прописными, так и строчными буквами. Броузер будет их однозначно интерпретировать в любом написании.

Если просмотреть исходный текст типичной Web–странички, то можно увидеть примерно следующее содержание:

<HTML>

<HEAD>

<TITLE>

заголовок документа

</TITLE>

</HEAD>

<BODY>

текст

текст

текст

<ADDRESS>

текст

</ADDRESS>

</BODY>

</HTML>

Между дескрипторами <HTML> и </HTML> находится непосредственно весь документ. Дескрипторы <HEAD> и </HEAD> определяют область задания заголовка документа. Между парой <TITLE> и </TITLE> помещается заголовок документа. Его содержимое просматривается в названии окна браузера. Информация в заголовке используется некоторыми сетевыми роботами для индексирования данных при сканировании ресурсов WWW. В эту область можно вводить, например, данные об авторе, краткое описание документа. Между дескрипторами <BODY> и </BODY> заключаются данные, которые будут выводиться браузером на экран. Дескрипторы <ADDRESS> и </ADDRESS> содержат данные об авторе, которые будут выводиться на экран, например, имя и фамилию автора, его электронный адрес, дата создания и изменения документа и т.д.

Строго говоря, ни одни из вышеописанных дескрипторов не являются обязательными при создании документов, однако они помогают структурировать документы и их определение является признаком хорошего стиля при составлении Web–страничек.

2.2 CSS

CSS (англ. Cascading Style Sheets — каскадные таблицы стилей) — формальный язык описания внешнего вида документа, написанного с использованием языка разметки.

Преимущественно используется как средство описания, оформления внешнего вида веб-страниц, написанных с помощью языков разметки HTML и XHTML, но может также применяться к любым XML-документам, например, к SVG или XUL.

CSS используется создателями веб-страниц для задания цветов, шрифтов, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS являлось разделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом. Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печатное представление, чтение голосом (специальным голосовым браузером или программой чтения с экрана), или при выводе устройствами, использующими шрифт Брайля.

Как известно, HTML-документы строятся на основании иерархии элементов, которая может быть наглядно представлена в древовидной форме. Элементы HTML друг для друга могут быть родительскими, дочерними, элементами-предками, элементами-потомками, сестринскими.

Элемент является родителем другого элемента, если в иерархической структуре документа он находится сразу, непосредственно над этим элементом. Элемент является предком другого элемента, если в иерархической структуре документа он находится где-то выше этого элемента.

Пускай, например, в документе присутствуют два абзаца p, включающие в себя шрифт с полужирным начертанием b. Тогда элементы b будут дочерними элементами своих родительских элементов p, и потомками своих предков body. В свою очередь, для элементов p элемент body будет являться не только предком, но непосредственно и родителем. И кроме того, эти два элемента p будут являться сестринскими элементами, как имеющими одного и того же родителя — body.

В CSS могут задаваться, при помощи селекторов, не только одиночные элементы, но и элементы, являющиеся потомками, дочерними или сестринскими элементами других элементов (см. подраздел «виды селекторов»).

В первых трёх случаях подключения таблицы CSS к документу (см. выше) каждое правило CSS из таблицы стилей имеет две основные части — селектор и блок объявлений. Селектор, расположенный в левой части правила, определяет, на какие части документа распространяется правило. Блок объявлений располагается в правой части правила. Он помещается в фигурные скобки, и, в свою очередь, состоит из одного или более объявлений, разделённых знаком «;». Каждое объявление представляет собой сочетание свойства CSS и значения, разделённых знаком «:». Селекторы могут группироваться в одной строке через запятую. В таком случае свойство применяется к каждому из них.

селектор, селектор {

свойство: значение;

свойство: значение;

свойство: значение;

}

В четвёртом случае подключения таблицы CSS к документу (см. список) правило CSS (являющееся значением атрибута style тега, на который оно действует) представляет собой перечень объявлений («свойство CSS : значение»), разделённых знаком «;».

Класс или идентификатор может быть присвоен какому-нибудь элементу (тегу) HTML посредством атрибутов class или id этого элемента (тега):

<div id="first"> ... </div>

<p class="big"> ... </p>

Основное отличие между классами элементов и идентификаторами элементов в том, что в документе какой-нибудь класс может быть присвоен сразу нескольким элементам, а идентификатор - только одному. Также отличие в том, что могут существовать множественные классы (когда класс элемента состоит из нескольких слов, разделённых пробелами). Для идентификаторов такое невозможно.

Важно отметить следующее отличие идентификатора от класса: идентификаторы широко используются в JavaScript для нахождения уникального элемента в документе.

Имена классов и идентификаторов, в отличие от названий тегов и их атрибутов, чувствительны к регистру ввода букв.

Свойства классов и идентификаторов задаются с помощью соответствующих селекторов. Причём может быть задано как свойство класса в целом (в таком случае селектор начинается с «.»), или свойство идентификатора самого по себе (в таком случае селектор начинается с «#»), так и свойство какого-нибудь элемента этого класса или с этим идентификатором.

В CSS помимо классов, задаваемых автором страницы, существует также ограниченный набор так называемых псевдоклассов, описывающих вид гиперссылок с определённым состоянием в документе, вид элемента, на котором находится фокус ввода, а также вид элементов, являющихся первыми дочерними элементами других элементов. Также в CSS существует четыре так называемых псевдоэлемента: первая буква, первая строка, применение специальных стилей до и после элемента.

Применение CSS к документам HTML основано на принципах наследования и каскадирования. Принцип наследования заключается в том, что свойства CSS, объявленные для элементов-предков, наследуются элементами потомками. Но, естественно, не все свойства CSS наследуются — например, если для тега параграфа p средствами CSS задана рамка, то она не будет наследоваться ни одним тегом, содержащимся в данном теге p, а вот если для параграфа p средствами CSS задан цвет шрифта (например, color:green;), то это свойство будет унаследовано каждым элементом-тегом, находящимся в параграфе.

Принцип каскадирования применяется в случае, когда какому-то элементу HTML одновременно поставлено в соответствие более одного правила CSS, то есть, когда происходит конфликт значений этих правил. Чтобы разрешить такие конфликты вводятся правила приоритета.

2.3 JavaScript

JavaScript — объектно-ориентированный скриптовый язык программирования. Является диалектом языка ECMAScript.

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса.

На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на Java, но при этом лёгким для использования непрограммистами. Языком JavaScript не владеет какая-либо компания или организация, что отличает его от ряда языков программирования, используемых в веб-разработке.

JavaScript является объектно-ориентированным языком, но используемое в языке прототипирование обуславливает отличия в работе с объектами по сравнению с традиционными класс-ориентированными языками. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам — функции как объекты первого класса, объекты как списки, карринг, анонимные функции, замыкания — что придаёт языку дополнительную гибкость.

Несмотря на схожий с Си синтаксис, JavaScript по сравнению с языком Си имеет коренные отличия:

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

В языке отсутствуют такие полезные вещи, как:

* модульная система: JavaScript не предоставляет возможности управлять зависимостями и изоляцией областей видимости;
* стандартная библиотека: в частности, отсутствует интерфейс программирования приложений по работе с файловой системой, управлению потоками ввода/вывода, базовых типов для бинарных данных;
* стандартные интерфейсы к веб-серверам и базам данных;
* система управления пакетами, которая бы отслеживала зависимости и автоматически устанавливала их.

ядро (ECMAScript),

* объектная модель браузера (Browser Object Model или BOM (de));
* объектная модель документа (Document Object Model или DOM).

Если рассматривать JavaScript в отличных от браузера окружениях, то объектная модель браузера и объектная модель документа могут не поддерживаться.

Объектную модель документа иногда рассматривают как отдельную от JavaScript сущность, что согласуется с определением DOM как независимого от языка интерфейса документа. В противоположность этому ряд авторов находят BOM и DOM тесно взаимосвязанными

ECMAScript не является браузерным языком и в нём не определяются методы ввода и вывода информации. Это скорее основа для построения скриптовых языков. Спецификация ECMAScript описывает типы данных, инструкции, ключевые и зарезервированные слова, операторы, объекты, регулярные выражения, не ограничивая авторов производных языков в расширении их новыми составляющими.

Объектная модель браузера - браузероспецифичная часть языка, являющаяся прослойкой между ядром и объектной моделью документа. Основное предназначение объектной модели браузера - управление окнами браузера и обеспечение их взаимодействия. Каждое из окон браузера представляется объектом window, центральным объектом DOM. Объектная модель браузера на данный момент не стандартизирована, однако спецификация находится в разработке WHATWG и W3C.

Помимо управления окнами, в рамках объектной модели браузера, браузерами обычно обеспечивается поддержка следующих сущностей:

* 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 - интерфейс программирования приложений для HTML и XML-документов. Согласно DOM, документ (например, веб-страница) может быть представлен в виде дерева объектов, обладающих рядом свойств, которые позволяют производить с ним различные манипуляции:

* получение узлов;
* изменение узлов;
* изменение связей между узлами;
* удаление узлов.

JavaScript используется в клиентской части веб-приложений: клиент-серверных программ, в котором клиентом выступает браузер, а сервером - веб-сервер, имеющих распределённую между сервером и клиентом логику. Обмен информацией в веб-приложениях происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются кроссплатформенными сервисами.

JavaScript используется в AJAX, популярном подходе к построению интерактивных пользовательских интерфейсов веб-приложений, заключающемся в «фоновом» асинхронном обмене данными браузера с веб-сервером. В результате, при обновлении данных веб-страница не перезагружается полностью и интерфейс веб-приложения становится быстрее, чем это происходит при традиционном подходе (без применения AJAX).

Comet - широкое понятие, описывающее механизм работы веб-приложений, использующих постоянные HTTP-соединения, что позволяет веб-серверу отправлять данные браузеру без дополнительного запроса со стороны браузера. Для таких приложений используются технологии, непосредственно поддерживаемые браузерами. В частности, в них широко используется JavaScript.