**Глава 2.** Анализ существующих технических решений и выбор варианта, наиболее полно удовлетворяющего требованиям пользователей

2.1 Front-End

2.1.1. Язык разметки

2.1.1.1. XHTML

2.1.1.2. HTML

2.1.2. Каскадные таблицы стилей

2.1.2.1. CSS

2.1.2.2. SASS / SCSS

2.1.2.3. LESS

2.1.3. Язык скриптов

2.1.3.1. JavaScript

2.1.3.2. jQuery

2.1.3.3. Zepto.js

2.1.4. Сборщики проектов

2.1.4.1. Grunt

2.1.4.2. GulpJS

**2.1.1. Язык разметки**

Язык разметки (текста) в компьютерной терминологии — набор символов или последовательностей, вставляемых в текст для передачи информации о его выводе или строении. Принадлежит классу компьютерных языков. Текстовый документ, написанный с использованием языка разметки, содержит не только сам текст (как последовательность слов и знаков препинания), но и дополнительную информацию о различных его участках — например, указание на заголовки, выделения, списки и т. д.

В более сложных случаях язык разметки позволяет вставлять в документ интерактивные элементы и содержание других документов.

**2.1.1.1. XHTML**

XHTML (англ. extensible hypertext markup language — расширяемый язык гипертекстовой разметки) — семейство языков разметки веб-страниц на основе XML, повторяющих и расширяющих возможности HTML 4. Спецификации XHTML 1.0 и XHTML 1.1 являются рекомендациями консорциума Всемирной паутины. Развитие XHTML остановлено; новые версии XHTML не выпускаются; рекомендуется использовать HTML.

XHTML соответствует спецификации SGML, поскольку XML является её подмножеством.

Существует три типа документов XHTML: strict, transitional и frameset. Наиболее употребительной и универсальной из версий XHTML является переходная (англ. transitional), поскольку она позволяет использовать iframe (включение содержимого одной веб-страницы в другую) и атрибут target у ссылок (для указания того, например, что ссылке необходимо открываться в новом окне). Фреймовая версия (англ. frameset) представляет собой расширенный вариант transitional и добавляет к нему, как следует из названия, возможность установки frameset вместо body. DTD ((англ. document type definition) определение типа документа) строгой версии XHTML (англ. strict) не содержит многих тегов и атрибутов, описанных в DTD transitional и признанных устаревшими.

**2.1.1.2. HTML**

HTML (от англ. HyperText Markup Language — «язык гипертекстовой разметки») — стандартизированный язык разметки документов во Всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML. Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства.

Язык HTML является приложением SGML (стандартного обобщённого языка разметки) и соответствует международному стандарту ISO 8879.

Язык XHTML является более строгим вариантом HTML, он следует всем ограничениям XML и, фактически, XHTML можно воспринимать как приложение языка XML к области разметки гипертекста.

Во всемирной паутине HTML-страницы, как правило, передаются браузерам от сервера по протоколам HTTP или HTTPS, в виде простого текста или с использованием шифрования.

Язык HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986—1991 годах в стенах ЦЕРНа в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. HTML успешно справлялся с проблемой сложности SGML путём определения небольшого набора структурных и семантических элементов — дескрипторов. Дескрипторы также часто называют «тегами». С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Помимо упрощения структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были добавлены позже.

Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако современное применение HTML очень далеко от его изначальной задачи.

Текстовые документы, содержащие разметку на языке HTML (такие документы традиционно имеют расширение .html или .htm), обрабатываются специальными приложениями, которые отображают документ в его форматированном виде. Такие приложения, называемые «браузерами» или «интернет-обозревателями», обычно предоставляют пользователю удобный интерфейс для запроса веб-страниц, их просмотра (и вывода на иные внешние устройства) и, при необходимости, отправки введённых пользователем данных на сервер.

**2.1.2. Каскадные таблицы стилей**

**2.1.2.1. 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.

Правила CSS пишутся на формальном языке CSS и располагаются в таблицах стилей, то есть таблицы стилей содержат в себе правила CSS. Эти таблицы стилей могут располагаться как в самом веб-документе, внешний вид которого они описывают, так и в отдельных файлах, имеющих формат CSS.

**2.1.2.2. SASS / SCSS**

Sass (Syntactically Awesome Stylesheets) — это скриптовый метаязык, который компилируется в обычные CSS-стили.

Расширения SASS-файлов могут быть .sass и .scss — это зависит от выбранного синтаксиса. Браузер, впрочем, не понимает ни одного из них, поэтому для взаимопонимания нужно использовать компилятор. Его задача — привести SASS в понятный классический CSS, который будет распознан любым браузером.

Роль компилятора может выполнять серверный js или программа, установленная у вас на рабочей машине и мониторящая изменения в рабочих файлах.

Независимо от синтаксиса, SCSS обратно совместим с CSS. То есть любой CSS обязательно будет валидным SCSS-кодом.

Из-за отсутствия скобок и точек с запятой обратной совместимости у SASS-синтаксиса с CSS нет.

Преимущества SASS:

* SASS позволяет назначать переменные — и это одно из ключевых преимуществ. Переменная, по аналогии с php, начинается со знака доллара ($), значения присваиваются при помощи двоеточия. Переменные в Sass можно разделить на 4 типа:
  + число (int)
  + строка (string)
  + логический тип (да/нет, boolean)
  + цвета (имя, имена)
* Вложенные правила (nesting): возможность вкладывать правила в другие правила, что при компиляции создает нечто вроде пространства имен. Визуально вложенные в Sass внутрь родительского элемента правила будут сохранены с той же иерархией с учетом правил CSS.
* Дополнения (mixin): части кода, которые в CSS обычно приходилось дублировать, здесь можно сохранить в отдельной переменной и вставлять в нужных местах. Компилятор, встретив такую переменную, сохранит вместо нее нужный кусок кода.
* Наследование (extend): создав однажды любое правило, мы можем использовать его внутри другого. Наследуемый элемент получит все свойства исходного класса, которые мы можем дополнить любыми другими.

**2.1.2.3. LESS**

LESS – это надстройка над CSS. Это значит, что любой CSS код – это валидный LESS, но дополнительные элементы LESS не будут работать в простом CSS. Это замечательно, потому что существующий CSS уже является работоспособным LESS кодом, что уменьшает порог вхождения в новую технологию.

LESS добавляет много нужных динамических свойств в CSS. Он вводит переменные, операции, function-like элементы и примеси. Возможность писать таблицы стилей модульно избавит вас от многих хлопот.

Есть два способа использования LESS. Вы можете создать LESS файл и конвертировать его при помощи Javascript на лету или скомпилировать его заранее и использовать получившийся CSS файл.

LESS обеспечивает сравнимую функциональность с SASS, однако имеет ряд недостатков:

* не содержит логики/циклов
* не имеет helper (SASS позволяет использовать Compass)
* не имеет библиотеки адаптивных стилей (SASS позволяет использовать MQ-SASS)
* не позволяет проводить математические операции
* не позволяет генерировать спрайты

**2.1.4. Сборщики проектов (таск-менеджеры)**

Таск-менеджер — небольшое приложение, которое используется для автоматизации скучных и рутинных, но, от того, не менее важных, задач, которые приходится постоянно выполнять в процессе разработки проекта. Такие задачи включают в себя, к примеру, запуск модульных тестов, конкатенацию файлов, минификацию, препроцессинг CSS. Довольно простая идея, которая позволяет сэкономить очень много времени, и помогает сохранять фокус на задачах, связанных непосредственно с разработкой проекта.

**2.1.4.1. Grunt**

Grunt — это инструмент для сборки javascript проектов из командной строки с использованием задач. Релиз вышел совсем недавно, автор Ben «Cowboy» Alman, проект есть на github.

С Grunt-ом, JS разработчики имеют очень гибкий инструмент под рукой, который помогает при автоматизации повторяющихся задач. После настройки, вы можете настроить задачи по времени, сделать это волшебство в фоновом режиме, или интегрировать Grunt в ваш IDE или редактор для кодирования как внешнюю команду разработки (которая прекрасно работает в Sublime Text, например).

Код проверки и тестирования повышает общее качество и поможет вам поддерживать постоянный стандарт кодирования, что делает Grunt также идеальным инструментом для команд, работающих на крупномасштабных проектах.

Grunt устанавливается как NPM (Node Package Manager) модуль.

**2.1.4.2. GulpJS**

Gulp [/gʌlp/ /галп/] — система потоковой сборки. Он позволяет автоматизировать повторяющиеся задачи (сборка и минификация CSS- и JS-файлов, запуск тестов, перезагрузка браузера и другие). Тем самым Gulp ускоряет и оптимизирует процесс веб-разработки.

Сам по себе Gulp очень скуден на возможности. Но все, что вам нужно вынесено в отдельные плагины.

Gulp.task определяет наши задачи. В качестве аргументов принимает название, зависимости (массив) и функцию (основные действия). Зависимостей может и не быть.

Gulp.src указывает на файлы, которые мы хотим использовать. Он использует .pipe доступа к файлам через плагины.

Gulp.dest указывает на папку, в которую мы хотим сохранить измененные файлы.

Gulp.src и gulp.dest используется для простой копии файлов.

В gulp встроена система реагирования на изменения файлов (gulp.watch).

Благодаря своей архитектуре, основанной на потоках, справляется с идентичными заданиями на 50-80% быстрее, чем Grunt, а также содержит меньше кода в списке задач.