**Инструменты использующиеся в разработке Web модуля приложения**

Руководство по Node.js, часть 4: npm, файлы package.json и package-lock.json <https://habr.com/ru/company/ruvds/blog/423703/>

Объясняем современный JavaScript динозавру <https://habr.com/ru/company/mailru/blog/340922/>

Туториал по Jade для начинающих <https://habr.com/ru/post/278109/>

SASS - <https://ru.wikipedia.org/wiki/Sass>

SASS основы - <https://sass-scss.ru/guide/>

В Web части приложения использовался фреймворк Angular. Для упрощения и ускорения работы использовался npm менеджер пакетов. За автоматизацию сборки модулей в единые пакеты был использован Webpack. Pug и Sass препроцессоры предоставили возможность для быстрой верстки.

**NPM**

Npm (node package manager) — это менеджер пакетов Node.js. В npm имеется более полумиллиона пакетов, что делает его самым большим в мире репозиторием кода, написанного на одном языке. Это позволяет говорить о том, что в npm можно найти пакеты, предназначенные для решения практически любых задач.

Изначально npm создавался как система управления пакетами для Node.js, но в наши дни он используется и при разработке фронтенд-проектов на JavaScript. Для взаимодействия с реестром npm используется одноимённая команда, которая даёт разработчику огромное количество возможностей:

* *Загрузка пакетов*

С помощью команды npm можно загружать пакеты из реестра. Ниже мы рассмотрим примеры её использования.

* *Установка всех зависимостей проекта*

Если в проекте имеется файл package.json, то установить все зависимости этого проекта можно такой командой: *npm install*. Эта команда загрузит всё, что нужно проекту, и поместит эти материалы в папку node\_modules, создав её в том случае, если она не существует в директории проекта.

* *Установка отдельного пакета*

Отдельный можно установить следующей командой: npm install <package-name>

Часто можно видеть, как эту команду используют не в таком вот простом виде, а с некоторыми флагами:

* Флаг --save позволяет установить пакет и добавить запись о нём в раздел dependencies файла package.json, который описывает зависимости проекта. Эти зависимости используются проектом для реализации его основного функционала, они устанавливаются в ходе его развёртывания на сервере (после выхода npm 5 записи об устанавливаемых пакетах в разделе зависимостей делаются автоматически, и без использования этого флага).
* Флаг --save-dev позволяет установить пакет и добавить запись о нём в раздел, содержащий перечень зависимостей разработки (то есть — пакетов, которые нужны в ходе разработки проекта, вроде библиотек для тестирования, но не требуются для его работы) файла package.json, который называется devDependencies.
* *Обновление пакетов*

Для обновления пакетов служит следующая команда: npm update

Получив эту команду, npm проверит все пакеты на наличие их новых версий, и, если найдёт их новые версии, соответствующие ограничениям на версии пакетов, заданным в package.json, установит их.

Обновить можно и отдельный пакет: npm update <package-name>

Эти и многие другие возможности npm делают его необходимым в современном мире разработке приложений.

**Webpack**

В большинстве языков программирования есть возможность импортирования кода из одного файла в другой. Изначально в JS такой возможности не было, потому что этот язык разрабатывался только для исполнения в браузере, без доступа к файловой системе на клиентской машине (по причинам безопасности). Так что долгое время для организации JS-кода в нескольких файлах требовалось загружать каждый файл с глобально доступными переменными.

Webpack - это инструмент для сборки модулей в единые пакеты, имеющий доступ к файловой системе. Получающиеся пакеты совместимы с браузером, которому не нужен доступ к файловой системе. В нашем случае бандлер нужен для поиска всех выражений require (имеющих ошибочный, с точки зрения браузера, JS-синтаксис) и замены на настоящее содержимое каждого требуемого файла. В финале получается единый JS-файл без выражений require.

**Pug**

Pug - это препроцессор HTML и шаблонизатор, который был написан на JavaScript для Node.js. Pug — ребрендинг Jade. Был переименован из-за жалобы некоего одноименного бренда, не связанного с IT. Pug это именно то средство, которое предоставляет вам возможность написания разметки совершенно по новому, с целым рядом преимуществ по сравнению с обычным HTML.

Три основные черты Pug:

* Простые теги;
* Добавление атрибутов в теги;
* Блоки текста.

Полезные функции, которые делают разметку умнее. Среди них:

* JavaScript

Pug реализован на JavaScript, по этому использовать JavaScript в Pug довольно просто.

* Циклы

Pug использует прекрасный синтаксис для написания циклов, так что вам не придётся прибегать к JavaScript.

* Интерполирование

В строке создаётся конструкция ${...}, внутри которой вы можете поместить любую переменную или выражение.

* Миксины

Миксины, они как функции, они принимают параметры в качестве входных данных и генерируют соответствующую разметку. Миксины оглашаются с помощью ключевого слова mixin.

**SASS**

Sass — это метаязык на основе CSS, предназначенный для увеличения уровня абстракции CSS кода и упрощения файлов каскадных таблиц стилей.

Язык Sass имеет два синтаксиса:

* sass — отличается отсутствием фигурных скобок, в нём вложенные элементы реализованы с помощью отступов;
* SCSS (Sassy CSS) — использует фигурные скобки, как и сам CSS.

Написание CSS само по себе интересное занятие, но когда таблица стилей становится огромной, то становится и сложно её обслуживать. И вот в таком случае нам поможет препроцессор. Sass позволяет использовать функции недоступные в самом CSS, например, переменные, вложенности, миксины, наследование и другие приятные вещи, возвращающие удобство написания CSS.

Препроцессор обрабатывает ваш Sass-файл и сохраняет его как простой CSS-файл, который можно использовать на любом сайте.

Полезные функции SASS, которые делают таблицы стилей удобными в использовании:

* Переменные

Переменные - это способ хранения информации, которую используется на протяжении написания всех стилей проекта. Можно хранить в переменных цвета, стеки шрифтов или любые другие значения CSS, которые нужно использовать. Чтобы создать переменную в Sass нужно использовать символ $.

* Вложенности

Sass позволяет вкладывать CSS селекторы таким же образом, как и в визуальной иерархии HTML. Но чрезмерное количество вложенностей делает документ менее читабельным и воспринимаемым, что считается плохой практикой.

* Фрагментирование

Создавать фрагменты Sass-файла, которые будут содержать в себе небольшие отрывки CSS, которые можно будет использовать в других Sass-файлах. Это отличный способ сделать CSS модульным, а также облегчить его обслуживание. Фрагмент — это простой Sass-файл, имя которого начинается с нижнего подчеркивания, например, \_partial.scss. Нижнее подчеркивание в имени Sass-файла говорит компилятору о том, что это только фрагмент и он не должен компилироваться в CSS. Фрагменты Sass подключаются при помощи директивы @import.

* Расширение/Наследование

Это одна из самых полезных функций Sass. Используя директиву @extend можно наследовать наборы свойств CSS от одного селектора другому. Это позволяет держать Sass-файл в «чистоте». Класс-шаблон - особый тип классов, который выводится только при использовании расширения - это позволит сохранить ваш скомпилированный CSS чистым и аккуратным.

* Математические операторы

Использовать математику в CSS очень полезно. Sass имеет несколько стандартных математических операторов, таких как +, -, \*, / и %.