Project Manager

by Jed

30.06.2018

**ВВЕДЕНИЕ**

**Описание проекта**

Project Manager должен стать хорошей, лучшей альтернативой ведения отчетов, проблем, ошибок, информации, описания и т. д. проектов, вместо Microsoft Word документов.

Word документы слишком большие, поэтому находить и редактировать их слишком сложно. Нужно все время прокручивать их вниз для добавления новой записи и т. д. Это ужасно. Поэтому Web версия таких документов будет более лучше. Там будет возможность создавать задания, группировать их, описывать проекты, комментировать их и прочее.

Конечно, в интернете уже полно таких «велосипедов», но все они уж слишком не подходят. Какие-то слишком страшные. Другие платные. Третьи просто не поддерживают нужный функционал. Потому данный проект и станет решением всех этих проблем.

**Требования**

Проект должен быть создан на PHP Yii Framework 2. Также плюсом будет использование Bootstrap, Less/Sass или других препроцессоров.

В целом, создание проекта является способом наработки опыта и навыков с помощью фрейморков и библиотек. Поэтому большим плюсом будет использования большего количества дополнительных технологий.

**Важно:** это не движок и не похожее на это. Этот проект прямо нацелен на результат, так что не нужно думать над тем, чтобы сделать его универсальным. Для этого есть Lightness.

Кстати. А где же Lightness? Почему не с помощью него создавать данный сайт? Он ведь создавался, чтобы быть универсальным. Ответ: да, он бы неплохо подошел. Там есть уже готовые большие модули Core & Users, которые делают много работы. Пришлось бы лишь написать еще пару модулей. Но данный проект существует лишь ради изучения различных готовых технологий, поэтому Lightness, не подойдет.

**ПРОЕКТИРОВАНИЕ**

**#1 Методология HTML/CSS**

**Требования:**

* Абсолютная независимость блоков

**Предложение #1:** Методология якорей (пространств имен).

**Правила:**

1. Есть такие типы классов: блоки, элементы, модификаторы (как в БЭМ)
2. Каждому блоку всегда присваивается класс-якорь в виде *of-имя.*
3. Якоря присаиваются только блокам
4. Модификаторы могут применятся и к блокам, и к элементам. Записываются в виде *is-модификатор*
5. Блоки в CSS используются только в сочетании с якорями
6. Элементы в CSS всегда используются только в сочетании с родителями
7. Модификаторы используются только в сочетании со своим блоком или элементом.
8. Пространства имен можно сочетать друг с другом.
9. Имена пространств имен подбираются на основе принадлежности блока к определенной части страницы. Если данный блок будет использоваться на многих страницах или частях страниц, только и только тогда имя пространства имен может быть более обобщенным.

**Пример:**

.menu.of-profile {}

.menu.of-profile .item {}

.menu.of-profile .item.is-checked {}

**Преимущества:**

1. Блоки с одинаковыми названиями не будут смешиваться друг с другом, если находятся в разных пространствах имен.
2. Элементы всегда будут независимы так как используются только в сочетании с родителями. То есть, вне родителей их просто не существует.

**Недостатки:**

1. Длинные сочетания классов? (Но не такие длинные как в БЭМ) (Если вообще можно считать это недостатком при том, что оно делает блоки более независимыми).
2. При использовании библиотеки или фреймворка, если существует одиночный, не привязанный к чему-либо уникальному для этой библиотеки или фреймворка, блок, элемент или модификатор, классы могут смешаться. Например в библиотеке есть класс *.button* и в наших классах есть блок *.button.of-article*. Наш класс наследует свойства от класса библиотеки. **Но, в некоторых случаях, когда наследование было задумано изначально, этот недостаток не является недостатком.**
3. **ВСЕ ЕЩЕ НЕТ НЕЗАВИСИМОСТИ:** Если положить элемент блока, в другой блок, где есть элемент с таким же названием, то они смешаются.

**#2 Не срабатывают события Vue**

**Причина:** Запускалось сразу два vue экземпляра на одном и том же элементе #app. Видимо это приводит к некоррекктной работе vue.