Javascript и DOM

Установка

После установки Node.js и npm, создайте каталог project1 и распакуйте туда содержимое приложенного project2.zip файла. Архив содержит test-table.html, где вы должны выполнять персональные задания

Чтобы установить инструмент контроля качества кода, JSHint, выполнив следующую команду в каталоге project2:

npm install

Это позволит получить JSHint в подкаталоге node\_modules. Вы сможете запустить его для всех файлов JavaScript в каталоге project1, выполнив команду:

npm run jshint

Код, который вы отправляете, должен начинаться с «use strict»; в каждом файле JavaScript и при запуске JShint не должен выводить никаких предупреждений. Наличие предупреждний снижают общий балл за лабораторную

Задание 1. разработка элемента управления «выбор даты (datepicker)» (15 баллов)

То, как примерно должен выглядеть ваш datepicker: <https://www.google.ru/search?tbm=isch&q=datepicker>

В каталоге project1, создайте новый файл datepicker.js, в котором вы будете разрабатывать класс DatePicker. Использоваться этот класс должен так, как представлено ниже

var datePicker = new DatePicker("div1", function (id, fixedDate) {

console.log("DatePicker with id", id,

"selected date:", fixedDate.month + "/" + fixedDate.day + "/" + fixedDate.year);

});

datePicker.render(new Date("July 4, 1776"));

Конструктор принимает аргументы:   
*id* – htmlId элемента в котором будет отрисован datepicker  
*callback –* коллбек функция которая должна быть вызвана, когда дата выбрана. Параметры коллмек функции:  
 *id* – html Id элемента с datepicker  
 *fixedDate* – объект, содержащий три свойства month, day, year, например *{month: 1, day: 30, year: 2016}*

Созданный класс должен иметь метод *render,* который принимает в качестве аргумента объект типа Date. Внутри он должен отрисовать дни месяца из переданного объекта. Когда метод render вызывается, он должен стереть все содержимое контейнера DIV дейтпикера, и сгенерировать новый html содержащий небольшой 1-месяцевый календарь.



Требования к выводимому календарю:

* Календарь отображает дни выбранного месяца в таблице, где одна строка = одной неделе, и колонка = дню недели
* Заголовки должны отображать дни недели, как например"Su", "Mo", и т.д.
* Ячейка таблицы – день - число
* Первая и последняя недели могут содержать дни не из выбранного месяца – они либо не должны быть отрисованы(пустые ячейки), либо выглядеть неактивными   
  
* Количество строк в календаре не должно быть фиксировано и может варьироваться в зависимости от количества недель в месяце (4-6)
* Календарь должен отображать название месяца и года в заголовке. Дополнительно может содержать указатели (например "<" and ">") на смену месяца
* Нажатие на ячейку со днем **выбранного** месяца должно вызывать коллбек функцию описанную выше. Неактивные дни (из прошлого\следующего) месяца не должны нажиматься

Для созданного вами класса в архиве присутствует файл datepicker.html, содержащий два пустых DIV элемента-контейнера, и немного javascript кода, который создает и дейтпикеры для каждого из контейнера. Два дейтпикера должны существовать независимо и управлять только прорисовкой своего элемента-контейнера

datepicker.html не содержит каких-либо стилей для календаря, поэтому вы должны создать datepicker.css, содержащий необходимые css классы, чтобы календарь выглядел красиво.  
Ссылка на подключение css файла уже присутствует в html.  
<link rel="stylesheet" type="text/css" href="datepicker.css" />

Задание 2. шаблонизатор таблиц (10 баллов)

В каталоге project, создайте новый файл TableTemplate.js.

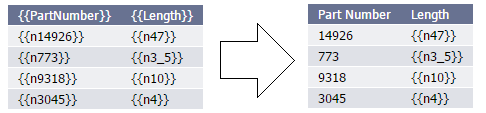
Создайте класс шаблонизатора (TableTemplate) который имеет **статический** метод *fillIn*.

Требования для метода fillIn

* Метод принимает три аргумента:   
  id атрибут для <table> элемента.   
  Объект – словарь  
  columnName – строка – название колонки
* Метод должен в первую очередь обработать заголовочную строку в таблице и заменить любой текст {{property}} соответствующим значением property из переданного объекта-словаря. Затем, если параметр columnName передан, то заменяем содержимое ячеек только из соответсвующей колонки. Если не передан – обрабатываем всю таблицу. Примеры вызовов и того, что должно получиться см в файле test-table.html
* Метод предполагает, что первая строка таблицы содержит наименования колонок. Названия колонок в таблице должны быть уникальные. Если параметр columnName не определен, метод должен обработать всю таблицу.
* Если колонка с именем не найдена, метод должен вернуть необработанную таблицу

Например если вы нашли элемент td в соответствующей колонке (считаем, что параметр columnName передан), который имеет текст "The date is the {{day}} day of month {{month}} of the year {{year}}" и переданный в метод «Объект – словарь» такой {month: "January", day: "30", year: "2016"}, то вы должны обновить текст внутри td чтобы он стал таким "The date is the 30 day of month January of the year 2016".

Вызвав TableTemplate.fillIn("table", dict, 'Part Number'), где "table" это таблица слева и dict это массив строк, вы должны сгененировать таблицу справа:



Если шаблон содержит свойство, которого нет в исходном массиве, то оно должно замениться на пустую строку. Пример: "This {{undefinedProperty}} is cool" -> "This is cool".

Допускается обработка только правильно оформленных шаблонов. Необходимо возвращать undefined в случаях:

Вложенных свойств - {{foo {{bar}}}} or {{{{bar}}}} or {{{bar}}}

Несбалансированных скобок - {{bar}}}

В качестве помощи используйте template-processor.js из предыдущей лабораторной работы.

Будьте внимательны – браузеры автоматически добавляют элемент tbody вокруг всех строк таблицы, если его нет в вашей разметке.

Как только функци закончит обработку, она должна изменить значение visibility  свойства таблицы. Если значение hidden , то оно должно стать visible

Файл test-table.html показывает пример использования шаблонизатора.

Дополнительные требования и замечания

Нельзя использовать jQuery и подобное. Низкоуровневое программирование dom на javascript будет полезно.

Все внутренние функции JS могут быть использованы, например:

* Методы Date класса
* parseInt, parseFloat, isNaN.

Полезными будут свойства элементов DOM (прочитайте о них дополнительно, например в htmlbook.ru):

* cells
* firstChild
* nextSibling
* rows
* tagName
* textContent
* innerHTML

Очень полезна функция console.log для отладки, которая выводит содержимое переданного параметра в лог. Сам лог доступен в браузерах "Tools -> JavaScript console"

Кроме того в логе выводятся ошибки в JS файлах, если они есть. (Если вы открыли страницу и у вас ничего не работает). Потратьте время на изучение возможностей отладки в Chrome DevTools – это сэкономит массу времени в дальнейшем.

Проект непростой, поэтому все вопросы задавайте, лично, по почте, в телеграме.

бонус. стиль (5 баллов)

Начисляется, если js код чистый, комментированный и JSHint  не возвращает предупреждения