

ЛАБОРАТОРНАЯ РАБОТА №4

Плагины jQuery. Работа с библиотекой jQuery UI

Цель работы: ознакомиться с основными возможностями фреймворка jQuery и библиотекой jQuery UI, научиться использовать плагины данной библиотеки.

Задача работы – последовательно выполнить поставленные задачи для получения практического опыта.

1. Теоретические сведения и практические упражнения

Библиотека jQuery UI позволяет реализовать на сайте множество полезных готовых решений: календарь, аккордеон, живой поиск, красивые эффекты анимации и многое другое.

1.1. Плагин FancyZoom

Наверно, каждому из Вас знаком скрипт Lightbox, который позволяет эффектно увеличивать изображения. Но в этом уроке мы рассмотрим не менее интересный скрипт, который, несомненно, может конкурировать с лайтбоксом по эффектности и привлекательности. Этот скрипт называется fancyZoom

Скачаем (<https://github.com/keegnotrub/jquery.fancyzoom>) и подключим к документу необходимые скрипты.

```
<link rel="stylesheet" href="css/fancyzoom.css">
<script src="http://code.jquery.com/jquery-1.11.0.min.js"></script>
<script src="js/jquery.fancyzoom.min.js"></script>
```

Внутри тега <body> пропишем следующий HTML:

```
<p><a href="img/big.jpg" class="fancy"></a></p>
<p><a href="img/big.jpg" class="fancy"></a></p>
<p><a href="img/big.jpg" class="fancy"></a></p>
<p><a href="img/big.jpg" class="fancy"></a></p>
```

Заполним его правильными ссылками на свои картинки.

В конец документа перед закрывающим тегом </body> поставим:

```
$(document).ready(function() {
    $(".a.fancy").fancyZoom();
});
```

Если все настроено правильно, то по клику на картинки будет появляться их увеличенный вариант.

Если добавить каждому элементу атрибут `title` и заполнить его информацией, то она будет выводиться как подпись под большой картинкой.

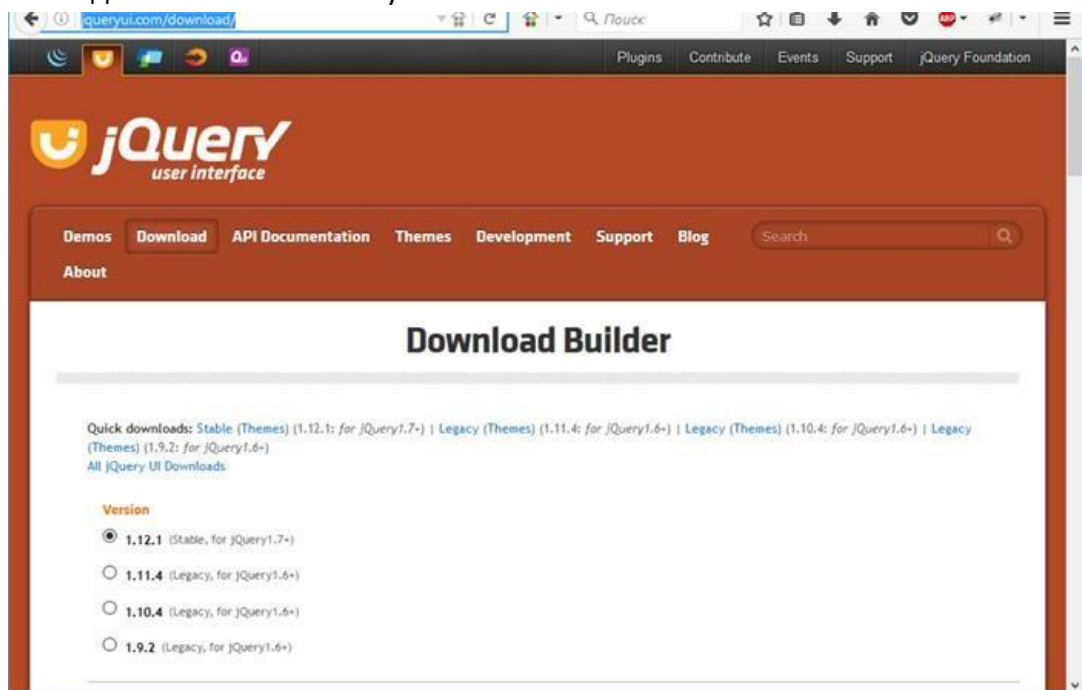
Если добавить каждому элементу атрибут `rel = "gr"`, то при отображении больших картинок будут появляться стрелочки влево/вправо позволяющие переключать большие фото. Значение атрибута `rel` играет роль идентификатора группы.

1.2. Библиотека jQuery UI. Установка и настройка

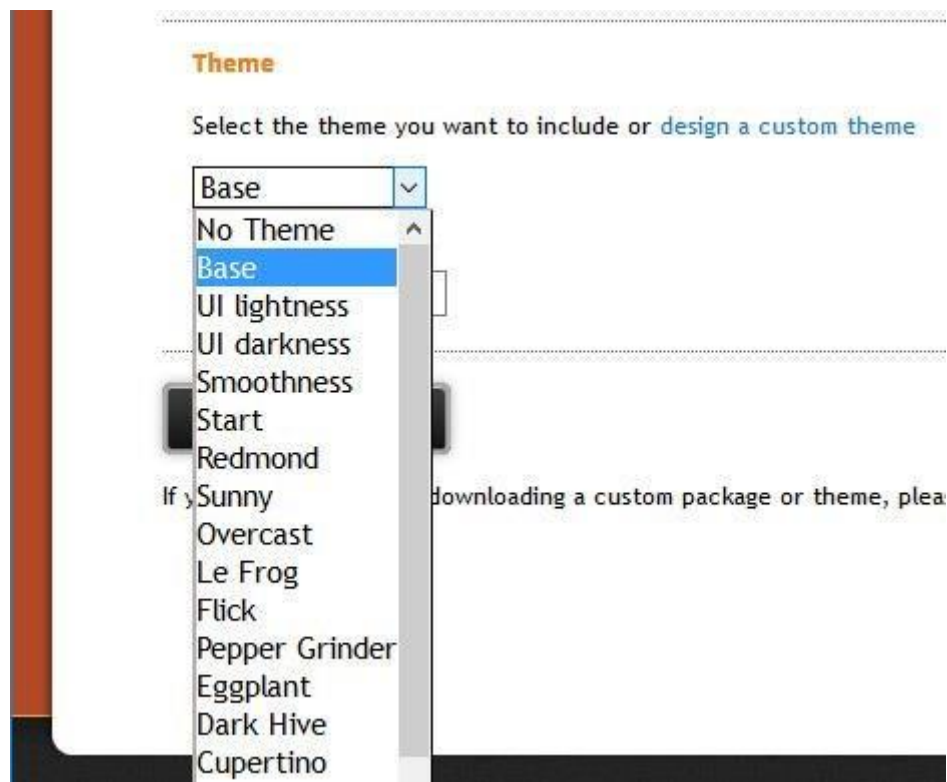
Итак, зачем нам нужна библиотека [jQuery UI](#)? Собственно, затем же, зачем мы обращаемся к сторонним плагинам – для установки и использования готовых решений на сайте. Вот только библиотека jQuery UI – это комплексное решение, т.е. здесь вы найдете не что-то одно, а сразу целый пакет виджетов, эффектов и плагинов для работы с различными событиями.

Вы можете скачать как весь пакет целиком, так и выбрать какой-то один или несколько виджетов. Также в комплекте библиотеки jQuery UI есть два десятка тем оформления практически на любой вкус. Это также существенный плюс.

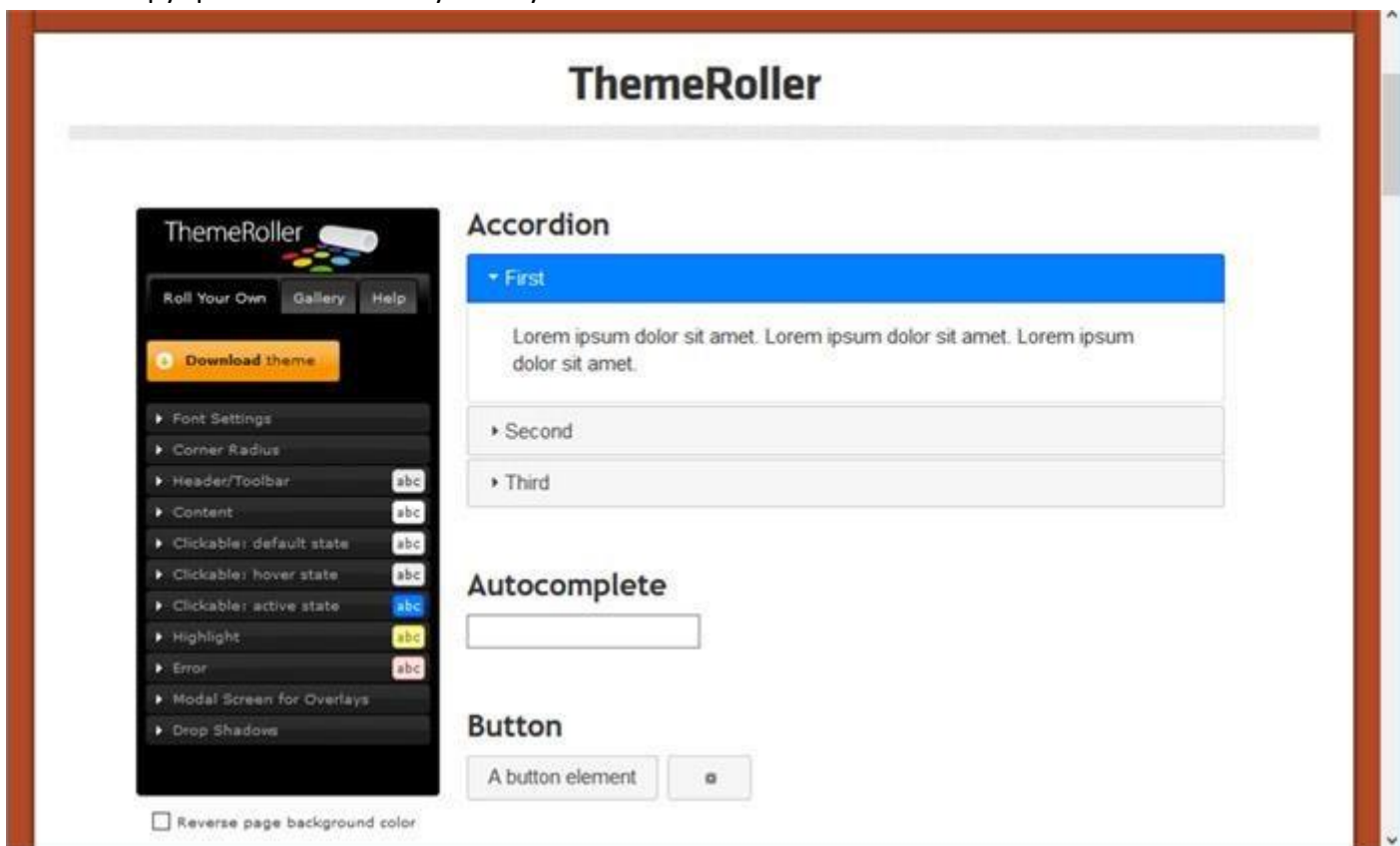
Перейдем от слов к делу: скачаем и установим библиотеку jQuery UI. Перейдем на официальный сайт и обратимся к [разделу Download](#). Именно здесь мы можем выбрать компоненты для скачивания и тему.



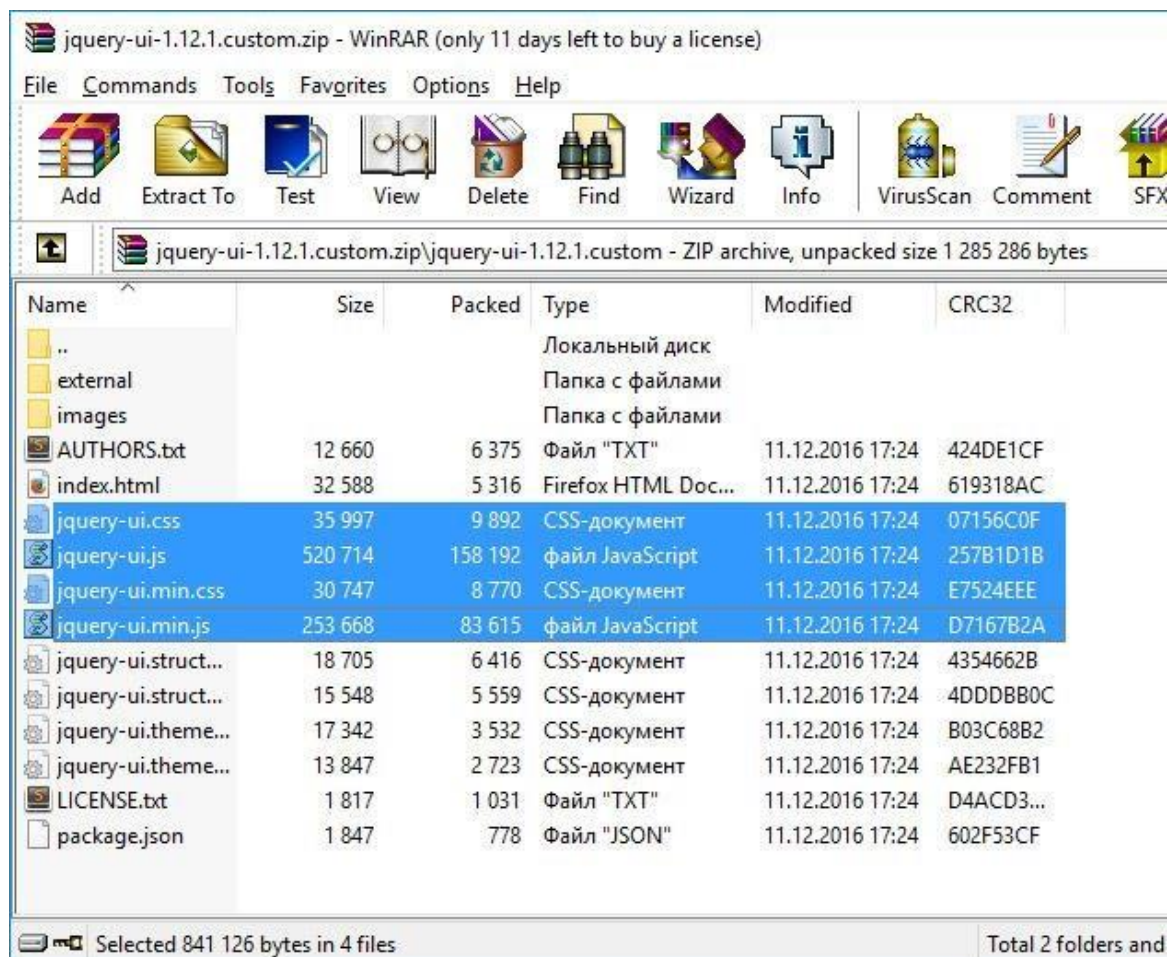
Тему можно выбрать из выпадающего списка внизу страницы.



Также при желании можно сконструировать свою тему с собственным оформлением, для этого достаточно кликнуть по ссылке [design a custom theme](#), которая находится сразу над выпадающим списком. На открывшейся странице мы можем изменять шрифты, цвета и прочее оформление элементов, наблюдая изменения в режиме онлайн, в общем, можем конструировать собственную тему.



Но вернемся на предыдущую страницу и скачаем все компоненты библиотеки jQuery UI с темой на ваш выбор (не Base).



В полученном архиве нам потребуется файл стилей (jquery-ui.css) и файл скриптов (jquery-ui.js). Оба файла предлагаются в обычной и сжатой версиях, поэтому можно выбрать любой. Ну и, само-собой, потребуется библиотека jQuery версии не менее 1.11.3 с сервиса Google.

В итоге страница со всеми подключениями будет выглядеть следующим образом:

```
<link rel="stylesheet" href="ui/jquery-ui.css">
```

```
<scriptsrc="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
```

```
<script src="ui/jquery-ui.js"></script>
```

Вот, собственно, и вся премудрость подключения и настройки библиотеки jQuery UI.

1.3. Виджет accordion библиотеки jQuery UI

Посмотреть примеры использования и документацию по плагину аккордеон можно в документации библиотеки jQuery UI в разделе [Widgets – Accordion](#). Для базовой работы аккордеона нам потребуется определенная разметка, а именно: некоторый блок, внутри которого идут секции. Каждая секция состоит из заголовка H3 и блока с содержимым секции.

Давайте создадим три секции с произвольным содержанием:

```

01. <div class="container content">
02.   <div id="accordion">
03.     <h3>Секция 1</h3>
04.     <div>
05.       <p>Lorem ipsum dolor sit amet, consectetur adipisicing
06.         elit. Explicabo, ea!</p>
07.     </div>
08.     <h3>Секция 2</h3>
09.     <div>
10.       <p>Lorem ipsum dolor sit amet, consectetur adipisicing
11.         elit. Explicabo, ea!</p>
12.     </div>
13.     <h3>Секция 3</h3>
14.     <div>
15.       <p>Lorem ipsum dolor sit amet, consectetur adipisicing
16.         elit. Explicabo, ea!</p>
17.       <ul>
18.         <li>List item one</li>
19.         <li>List item two</li>
20.         <li>List item three</li>
21.       </ul>
22.     </div>
23.   </div>
24. </div>

```

Секция 1

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Explicabo, ea!

Секция 2

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Explicabo, ea!

Секция 3

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Explicabo, ea!

- List item one
- List item two
- List item three

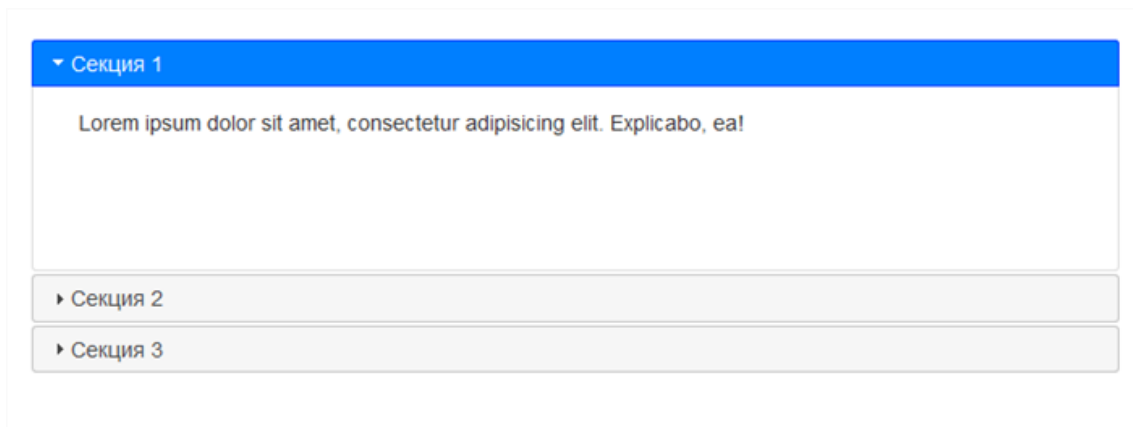
Пока что ничего особенного. Однако добавим немного магии библиотеки jQuery UI, тем более что сделать это крайне просто, всего одна строка кода:

```

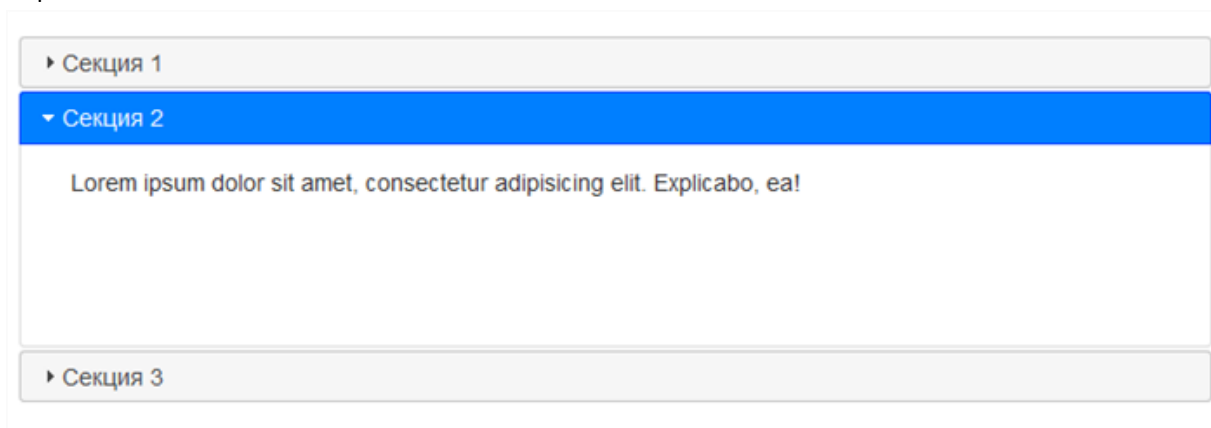
1. $(function(){
2.   $( "#accordion" ).accordion();
3. });

```

Здесь мы для блока обертки вызвали метод accordion, тем самым указав, что мы хотим получить одноименный виджет – аккордеон. Вот результат:



Буквально за минуту времени мы получили отличное решение в виде аккордеона секций. Клик по закрытой секции развернет ее, скрыв при этом предыдущую открытую секцию:



Также в метод `accordion` плагина мы можем передавать различные параметры, изменяя функционал по умолчанию. Список доступных параметров можно посмотреть в [документации](#). Например, базовый функционал предполагает всегда одну открытую секцию, при этом клик по активной секции в попытке закрыть ее – ни к чему не приведет. Если же мы хотим иметь возможность закрывать активную секцию, для этого достаточно лишь передать параметр `collapsible` со значением `true`:

```
$( "#accordion" ).accordion({  
    collapsible: true  
});
```

Теперь клик по активной секции свернет ее:



1.4. Виджет `autocomplete` библиотеки jQuery UI

Это действительно эффектный плагин, который позволяет в считанные минуты реализовать на сайте живой поиск по типу поиска Google или Яндекс. Т.е. пользователь начинает набирать поисковый запрос, а ему сразу же начинают предлагаться возможные варианты поисковых запросов.

Примеры и документацию к плагину `autocomplete` библиотеки jQuery UI можно найти на следующей [странице](#). Базовый пример продемонстрирует суть работы виджета. Попробуем ввести латинскую литеру `a` и увидим выпадающий список с подсказками – это

слова, в которых встречается набранная буква. Попробуем добавить еще одну букву, скажем латинскую s – список уже станет меньше, в нем останутся лишь варианты с сочетанием литер as.



Действительно эффективное решение, которое на самом деле еще и очень удобно для пользователей вашего сайта. Давайте попробуем реализовать простейший функционал. Итак, нам потребуется поле типа text для ввода поискового запроса.

```
1. <form class="form-inline">
2.   <div class="form-group">
3.     <label for="search">Поиск: </label>
4.     <input type="text" class="form-control" id="search">
5.   </div>
6. </form>
```

Поиск:

А теперь немного магии jQuery UI, точнее ее плагина autocomplete. На сайте вы, скорее всего, будете при начале ввода в поле поиска отправлять асинхронный запрос (AJAX) и получать ответ с результатами поиска от сервера. К слову, реализацию такого варианта можно посмотреть в [этом](#) и нескольких следующих уроках.

В данном примере мы не будем писать полноценное решение для организации поиска, а ограничимся поиском по некоторому массиву. Итак, нам нужен массив в JavaScript и всего один метода плагина autocomplete:

```

01. $(function(){
02.     var availableTags = [
03.         "ActionScript",
04.         "AppleScript",
05.         "Asp",
06.         "BASIC",
07.         "C",
08.         "C++",
09.         "Clojure",
10.         "COBOL",
11.         "ColdFusion",
12.         "Erlang",
13.         "Fortran",
14.         "Groovy",
15.         "Haskell",
16.         "Java",
17.         "JavaScript",
18.         "Lisp",
19.         "Perl",
20.         "PHP",
21.         "Python",
22.         "Ruby",
23.         "Scala",
24.         "Scheme"
25.     ];
26.     $( "#search" ).autocomplete({
27.         source: availableTags
28.     });
29. });

```

Вот, собственно, и все. Попробуйте ввести, как и в примере с официального сайта выше, букву а, затем s – и вы увидите готовое решение задачи.



Поиск работает. Как и другие плагины jQuery UI, плагин autocomplete можно настраивать. Список опций можно найти [здесь](#). Например, при реализации настоящего поиска вам, скорее всего, нужно будет начинать поиск не с первого символа, а, скажем, с третьего, чтобы не загружать сервер бессмысленными запросами. Сделать это поможет опция minLength:

```

$(           "#search"
).autocomplete({    source:
availableTags, minLength: 3
});

```

Теперь поиск начнется только после ввода третьего символа.

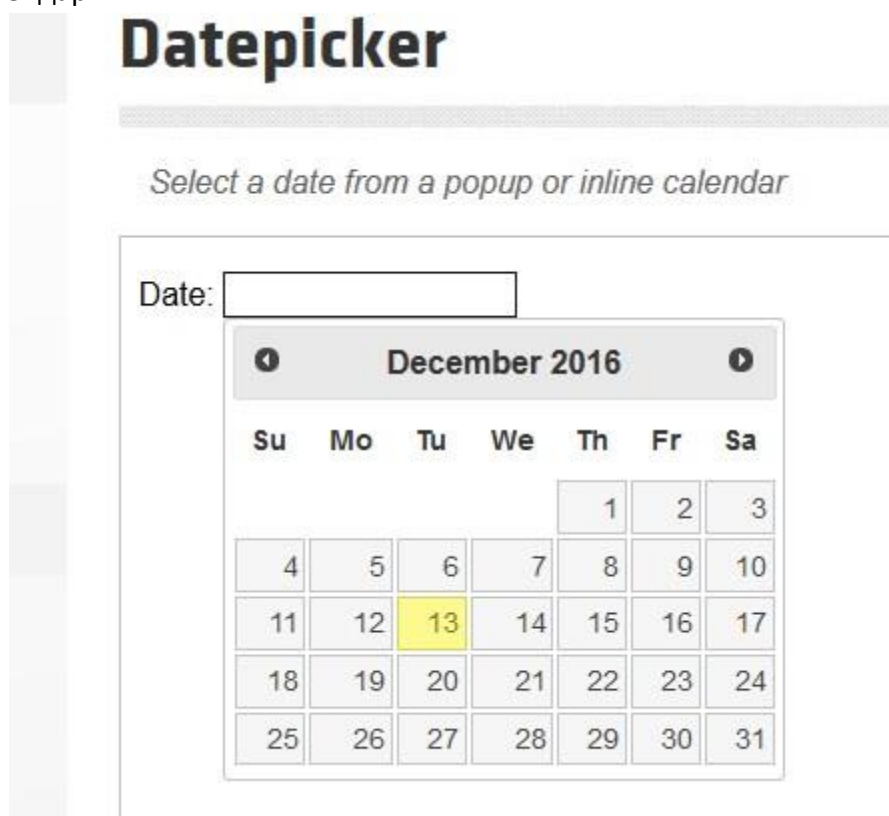
1.5. виджет DatePicker (календарь) библиотеки jQuery UI

Следующий на очереди плагин библиотеки jQuery UI – виджет Datepicker (календарь). Как следует из названия, плагин предлагает виджет календаря, который можно добавить к полю формы для выбора даты.

Зачем нам может понадобиться этот виджет? Например, у нас есть форма регистрации пользователей, в которой есть поле для ввода даты рождения пользователя. При этом дата рождения нам нужно в строго определенном формате. Календарь идеально здесь поможет, поскольку мы не даем пользователю вводить дату вручную, а предоставляем выбрать дату из календаря, ну а календарь уже вставит в поле выбранную дату в нужном формате.

Это первый и главный из плюсов виджета Datepicker. Второй немаловажный плюс – это красота. Посетителям нравятся разные красивые и необычные штуки на сайте, а календарь, безусловно, является таковой. И хотя многие уже привыкли к таким календарям, все равно календари не утратили своей привлекательности, это удобно, практично и красиво. Ко всему прочему, Календарь, как и прочие плагины jQuery UI, устанавливается в считанные минуты и гибко настраивается.

Примеры и документацию по плагину можно найти на соответствующей [странице документации](#). В базовом примере мы видим поле формы, клик по которому вызовет компактный календарик.



Итак, давайте создадим аналогичное поле для ввода даты на нашей странице:

Дата:

```

1. <div class="container content">
2.   <form class="form-inline">
3.     <div class="form-group">
4.       <label for="date">Дата: </label>
5.       <input type="text" class="form-control" id="date">
6.     </div>
7.   </form>
8. </div>

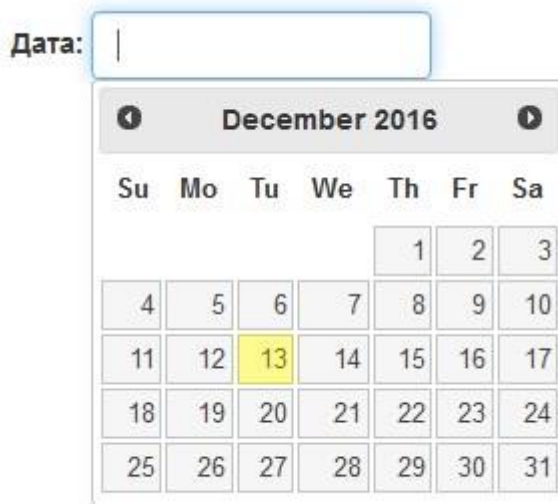
```

А теперь, как обычно, чуток магии jQuery UI для добавления календаря к имеющемуся полю:

```

$(function(){
    $("#date").datepicker();
});

```



Календарь появился, плагин Datepicker работает и очень прост в использовании. Давайте теперь попробуем выбрать какую-нибудь дату, скажем 31 декабря 2016 года.

Дата: 12/31/2016

Дата выбирается и подставляется в поле, вот только формат даты для нас не очень привычен. Однако это легко поправить с помощью [настроек](#) плагина Datepicker. В частности, нам поможет опция dateFormat:

```

$("#date").datepicker({
    dateFormat: "dd.mm.yy"
});

```

Вот теперь отлично, дата представлена в более понятном виде – День.Месяц.Год.

1.6. Виджет dialog библиотеки jQuery UI

Модальные окна – штука уже достаточно давняя в вебе. Однако, модальные окна до сих пор активно используются и не утратили своей актуальности. В первую очередь

благодаря привлекательности и компактности: посетителям нравятся всплывающие окна, ну а разработчику они позволяют разместить большой объем данных на странице компактнее. Например, мы вполне можем скрыть на странице форму авторизации или регистрации, которую покажем затем в модальном окне. Это может быть не обязательно форма, а любые другие данные, которые будут показаны в диалоговом окне при клике по кнопке или ссылке.

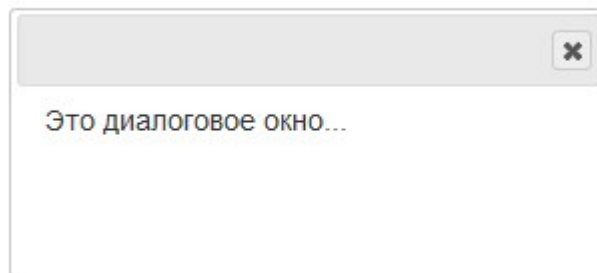
Плагин [dialog](#) библиотеки jQuery UI позволяет быстро создавать диалоговые окна для вашей страницы. Давайте на простом примере рассмотрим создание диалогового окна, которое будет показано при клике по кнопке.

Итак, для начала создадим некий блок с данными:

```
<div id="dialog">  
    <p>Это диалоговое окно...</p>  
</div>
```

Пока что ничего особенного, текст блока не скрыт и прекрасно виден на странице. Теперь попробуем использовать для данного блока метод плагина – dialog():

```
$(function(){  
    $('#dialog').dialog();  
});
```



В результате мы увидим диалоговое окно по центру страницы. Сам блок с текстом при этом не показывается на самой странице, только в модальном окне. Если мы закроем окно, кликнув по крестику, оно закроется, как и положено, текст при этом не будет появляться на странице.

Это хорошо, но не совсем то, что мы хотели, поскольку окно показывается сразу же после загрузки страницы. Нам же нужно, чтобы окно было скрыто и показывалось при нажатии кнопки.

Для реализации задуманного необходимо настроить модальное окно, используя [API плагина dialog](#). В частности, нам потребуется настройка autoOpen со значением false:

```
$('#dialog').dialog({  
    autoOpen: false  
});
```

Хорошо, окно мы скрыли, но как же теперь показать его по требованию? Ну для начала нам потребуется кнопка, которая будет открывать модальное окно.

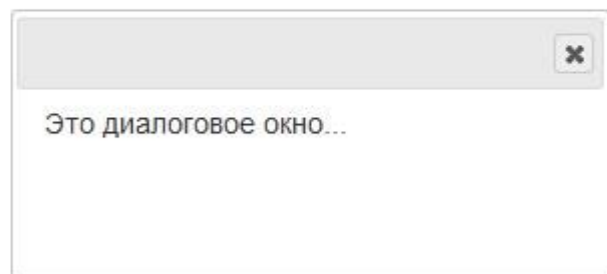
```
<button class="btn btn-success" id="open">Открыть окно</button>
```

Открыть окно

После этого остается отследить событие клика по данной кнопке и вызвать при наступлении события метод `dialog` с параметром `open`:

```
$('#open').click(function(){  
    $('#dialog').dialog('open');  
});
```

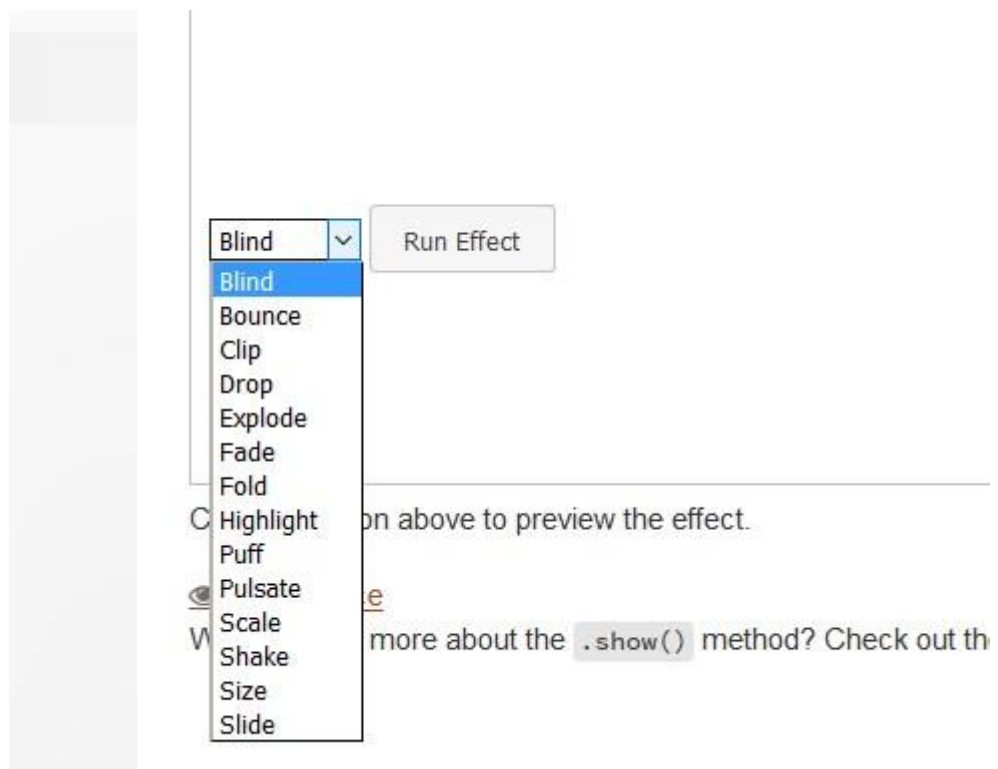
Открыть окно



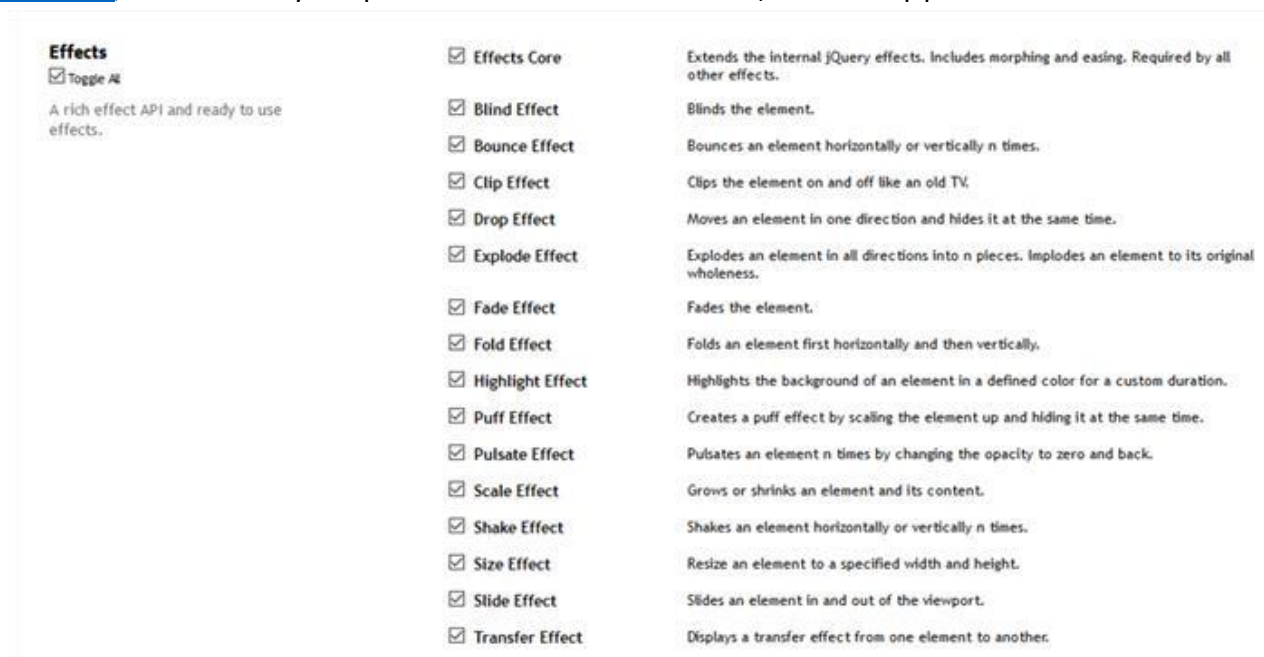
Вот теперь окно будет появляться при клике по кнопке. Обратите внимание, открытие и закрытие модального окна происходит достаточно обыденно, без эффектов. Но это легко поправить, при этом мы можем использовать как стандартные эффекты jQuery, так и эффекты библиотеки jQuery UI, которых достаточно большое количество.

1.7. Эффекты анимации библиотеки jQuery UI

Библиотека jQuery UI предоставляет в наше распоряжение свыше десяти [анимационных эффектов](#) на выбор.



Если нам не нужны абсолютно все эффекты, а необходимы лишь некоторые из них, тогда мы можем исключить ненужные на этапе сборки библиотеки jQuery UI на [странице загрузки](#), сняв отметку напротив того или иного анимационного эффекта:



Ну, а дальше все просто, собственно, как и при использовании ранее рассмотренных плагинов. Возьмем для примера тот же виджет dialog из предыдущего пункта и применим различные эффекты. Для появления пусть это будет, эффект clip, а для скрытия модального окна – эффект drop.

```
01. $(function(){
02.     $('#dialog').dialog({
03.         autoOpen: false,
04.         show: { effect: "clip", duration: 800 },
05.         hide: { effect: "drop", duration: 800 },
06.     });
07.     $('#open').click(function(){
08.         $('#dialog').dialog('open');
09.     });
10. });
```

Теперь открытие и закрытие диалогового окна смотрится куда как приятнее. Ну и, конечно же, можно обойтись стандартными эффектами анимации библиотеки jQuery, например, fadeIn/fadeout или slideDown/slideUp.

1.8. Виджет Selectmenu библиотеки jQuery UI

Как вы знаете, некоторые элементы форм достаточно неудобны в стилизации, проще говоря, средствами CSS их сложно привести к некоему нестандартному оформлению, при этом чтобы все выглядело красиво и кроссбраузерно.

К числу таких элементов относится, в частности, список select. Для стилизации таких элементов часто прибегают к услугам JavaScript и плагинам jQuery. Библиотека jQuery UI также предлагает свое решение для оформления списков select — это плагин [Selectmenu](#).

Итак, у нас есть стандартный выпадающий список select:

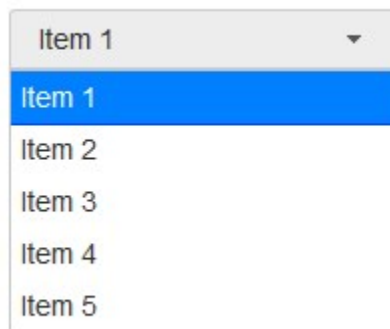
```
<select id="select">
  <option>Item 1</option>
  <option>Item 2</option>
  <option>Item 3</option>
  <option>Item 4</option>
  <option>Item 5</option>
</select>
```

Который выглядит также вполне стандартно.



А теперь используем одноименный метод плагина — selectmenu — и посмотрим на разницу в отображении, которую мы получим добавлением всего одной строки кода:

```
$(function(){
  $("#select").selectmenu();
});
```



Если вы хотите понять, как плагину удалось стилизовать выпадающий список, тогда можете заглянуть в консоль браузера, где увидите, что список select на самом деле скрывается, а вместо него создается список ul, который можно замечательно стилизовать по вашему усмотрению.

1.9. Виджет Menu библиотеки jQuery UI

Меню — это неотъемлемый атрибут любого сайта. Сейчас уже сложно кого-то удивить каким бы то ни было меню, а раньше, помнится, созданием обычного выпадающего меню для сайта можно было похвастаться на форуме. Были и такие времена 😊

Сегодня же для создания обычного выпадающего меню вам потребуется буквально десяток-другой строк кода CSS — и меню готово. Однако, если вам лень писать такой код,

и вы используете при этом библиотеку jQuery UI, тогда почему бы не обратиться к плагину [Menu?](#)

Начнем с разметки нашего меню. Если мы используем базовый вариант плагина, тогда разметка должна быть следующего вида:

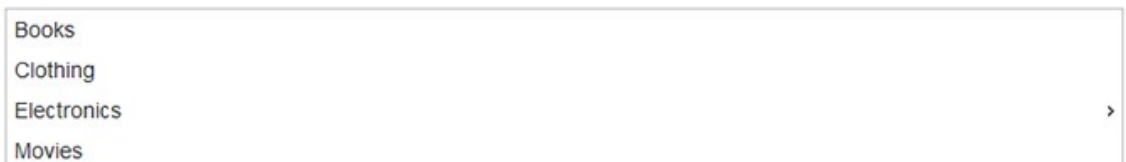
```
<ul id="menu">
  <li><div><a href="#">Books</a></div></li>
  <li><div><a href="#">Clothing</a></div></li>
  <li><div><a href="#">Electronics</a></div>
    <ul>
      <li><div><a href="#">Car Hifi</a></div></li>
      <li><div><a href="#">Utilities</a></div></li>
    </ul>
  </li>
  <li><div><a href="#">Movies</a></div></li>
</ul>
```

В результате мы получим следующую картину:

- Books
- Clothing
- Electronics
 - Car Hifi
 - Utilities
- Movies

Осталось вызвать одноименный метод плагина — `menu()`:

```
$(function(){
  $('#menu').menu();
});
```



Ну и для того, чтобы меню не выходило за рамки разумного, проще говоря, имело приемлемую ширину, добавим одно единственное правило стилей с нужным значением ширины:

```
.ui-menu { width: 150px; }
```

После этого меню в свернутом состоянии будет занимать указанную ширину в пикселях. Кстати, посмотрим, как оно выглядит в развернутом состоянии.



Неплохо. Кстати, а давайте попробуем убрать теги `div` внутри элементов списка `li`, в современных меню они просто лишние:



Меню немного поплыло, но проблема легко решается, просто сделаем тег ссылки блочным элементом и уберем подчеркивание:

```
ui-menu a{ display:
block; text-
decoration: none;
}
```

Вот теперь все отлично, верстка устраивает и меню работает отлично.

Однако стоит отменить, что на практике разработка такого вида меню как правило происходит полностью с помощью `css`. См. примеры: rilirb.ru, ntcea.ru, krm-group.ru и др.

1.10. Виджет Slider библиотеки jQuery UI

Штуку, которую позволяет реализовать виджет Slider библиотеки jQuery UI, я думаю, вы уже встречали не раз. Особенно популярен слайдер в интернет-магазинах при реализации подбора товаров по цене.

Например, мы хотим увидеть все товары в определенном диапазоне цен. Для этого можно было бы создать два поля, в первое из которых пользователь вводит стартовую цену, а во второе — максимальную цену товара. Более эффектный способ — использование ползунка, перетаскивая который мы изменяем диапазон значений цены.

Давайте начнем с базовой функциональности плагина. Создадим пусто блок `div` и вызовем для него метод `slider`: `<div id="slider"></div>` Иницилируем плагин:

```
$(function(){
    $('#slider').slider();
});
```

Вот такой результат мы получим:



Пока что не очень полезно. Давайте немного усложним и реализуем озвученный выше пример, т.е. создадим ползунок выбора диапазона цен. Для этого нам потребуется несколько настроек, описание которых можно найти в [документации](#) к плагину slider.

После непродолжительного знакомства с документацией вы вполне можете набросать вот такой пример:


```
<div class="container content">
  <div class="row">
    <div class="col-md-6">
      <form class="form-inline">
        <div class="form-group">
          <label for="start">Начальная цена</label>
          <input type="text" name="start" class="form-control" id="start">
        </div>
        <div class="form-group">
          <label for="end">Конечная цена</label>
          <input type="text" name="end" class="form-control" id="end">
        </div>
      </form>
      <br>
      <div id="slider"></div>
    </div>
  </div>
</div>
```

И код:

```
01. $(function(){
02.   $('#slider').slider({
03.     range: true, // используем диапазон
04.     min: 0, // минимальное значение
05.     max: 1000, // максимальное значение
06.     step: 10, // шаг изменения значения
07.     values: [50, 500], // начальные значения
08.     slide: function(event, ui){ // callback функция при
        изменении диапазона
09.       $('#start').val(ui.values[0]);
10.       $('#end').val(ui.values[1]);
11.     }
12.   });
13.   // записываем стартовые значения диапазона в соответствующие
        поля
14.   $('#start').val($('#slider').slider('values', 0));
15.   $('#end').val($('#slider').slider('values', 1));
16. });
```

В итоге мы получили вот такой симпатичный ползунок выбора диапазона цен:

Начальная цена Конечная цена



Пользователь теперь может как вводить цены вручную, так и просто перетаскать ползунок, что достаточно удобно.

1.11. Виджет Tabs библиотеки jQueryUI

Вкладки (табы, tabs) – это достаточно распространенная штука на современных сайтах. Особенно часто вкладки можно увидеть на страницах интернет-магазинов, где для карточки товаров можно компактно разместить большой объем информации: описание товара, его характеристики, фотографии, отзывы и т.п.

Вкладки как бы группируют большие объемы данных, делая страницу компактной и удобной для посетителей. Давайте попробуем создать такие вкладки, используя [виджет Tabs](#) библиотеки jQuery UI.

Для работы с плагином Tabs нам потребуется создать определенную структуру данных, определенную разметку. Названия вкладок должны представлять из себя список ul. Каждый элемент списка – это ссылка, которая по якорю будет связана с содержимым той или иной вкладки. Ну а контент вкладки – это блок div с идентификатором, соответствующим якорям ссылок. В общем, ниже необходимая нам разметка:

```
<div id="tabs">
  <ul>
    <li><a href="#tabs-1">Вкладка 1</a></li>
    <li><a href="#tabs-2">Вкладка 2</a></li>
    <li><a href="#tabs-3">Вкладка 3</a></li>
  </ul>
  <div id="tabs-1">
    <h2>Контент вкладки 1</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ex,
labore!</p>
  </div>
  <div id="tabs-2">
    <h2>Контент вкладки 2</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ex,
labore!</p>
  </div>
  <div id="tabs-3">
    <h2>Контент вкладки 3</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ex,
labore!</p>
  </div>
</div>
```

При этом в браузере мы сейчас увидим вот такую картину:

- Вкладка 1
- Вкладка 2
- Вкладка 3

Контент вкладки 1

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ex, labore!

Контент вкладки 2

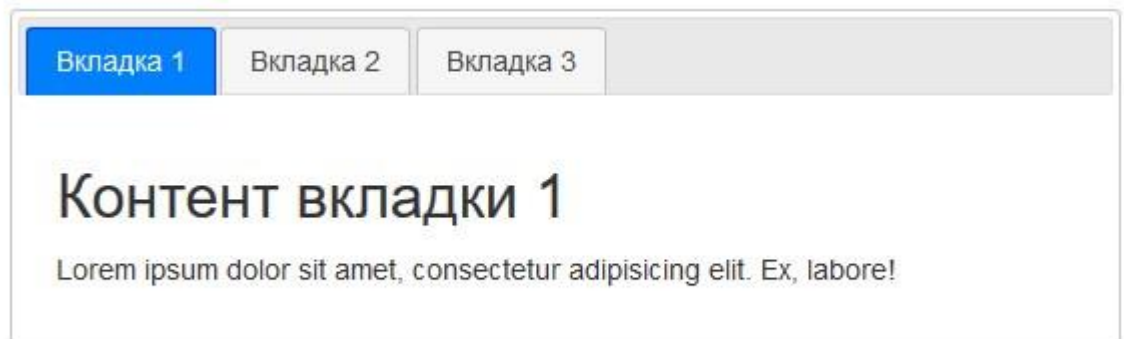
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ex, labore!

Контент вкладки 3

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ex, labore!

Но стоит нам обратиться к методу `tabs()` одноименного плагина, как картина чудесным образом преобразится.

```
$(function(){  
    $('#tabs').tabs();  
});
```



Теперь данные компактно сгруппированы по вкладкам, между которыми мы можем переключаться, получая содержимое выбранной вкладки. Очень удобно.

В [API документации](#) мы можем найти дополнительные настройки виджета Tabs. Например, не помешало бы добавить эффектов анимации при переключении вкладок. Это можно сделать следующим образом:

```
$(function(){  
    $('#tabs').tabs({ show: { effect: "fade",  
        duration: 300 }, hide: { effect: "fade",  
        duration: 300 },  
});  
});
```

Теперь переключение между вкладками происходит плавно и смотрится привлекательно.

1.12. Плагин Tooltip библиотеки jQuery UI. Создание пользовательских подсказок

Плагин библиотеки jQuery UI – Tooltip позволяет очень быстро и легко украсить добавить всплывающие подсказки, наполняемые атрибутом title html-элементов. К тому же он позволяет не только изменить дизайн подсказок, а и добавить различные анимационные эффекты в момент появления и скрытия подсказок.

Создадим следующую разметку:

```
<h2>Контент вкладки 1</h2>
```

```
<p>Lorem ipsum dolor ><a href="#" title="Текст подсказки">sit amet</a>, consectetur  
adipiscing elit. Ex,  
labore!</p>
```

Добавим код:

```
$(".td_top a").tooltip();
```

Как Вы видите, виджет Tooltip, успешно работает.

1.13. Плагин Sortable библиотеки jQuery UI

Как следует из названия, плагин работает с сортировкой элементов.

Часто данный плагин используется в административной части сайта, например, для управления позицией пунктов меню. Вы просто перетаскиваете пункты меню на нужные позиции и вместе с этим будет изменяться позиция элементов меню на сайте. Это очень удобно и выглядит действительно эффектно. Если вы работали с CMS WordPress, тогда вы знаете о чем речь, поскольку там для меню как раз используется такой плагин.

Давайте и мы попробуем реализовать такой эффект сортировки на практике. Как обычно, начнем с верстки. Меню верстается списками, поэтому нам потребуется такой список.

```
<ul id="sort" >  
  <li><a href="#">Item 1</a></li>  
  <li><a href="#">Item 2</a></li>  
  <li><a href="#">Item 3</a></li>  
  <li><a href="#">Item 4</a></li>  
  <li><a href="#">Item 5</a></li>  
</ul>
```

Перед нами обычный список элементов:

- Item 1
- Item 2
- Item 3
- Item 4
- Item 5

А теперь используем метод sortable() выбранного плагина:

```
$(function(){  
  $('#sort').sortable();
```


});

И попробуем перетаскивать элементы списка мышью.

- Item 2
- Item 3
- Item 1
- Item 4
- Item 5

Как видим, у нас получилось перенести первый элемент в середину списка. Ну и чтобы список красиво смотрелся, мы можем добавить несколько классов фреймворка Bootstrap, тем самым оформив список.

```
<ul id="sort" class="list-group">
  <li class="list-group-item"><a href="#">Item 1</a></li>
  <li class="list-group-item"><a href="#">Item 2</a></li>
  <li class="list-group-item"><a href="#">Item 3</a></li>
  <li class="list-group-item"><a href="#">Item 4</a></li>
  <li class="list-group-item"><a href="#">Item 5</a></li>
</ul>
```

Item 1
Item 2
Item 3
Item 4
Item 5

Также неплохо было бы изменить вид курсора, тем самым давая понять, что элементы можно перетаскивать, сортировать.

```
.list-group-item{cursor: move;}
```

Ну вот и все, сортировка элементов списка методом перетаскивания — готова.

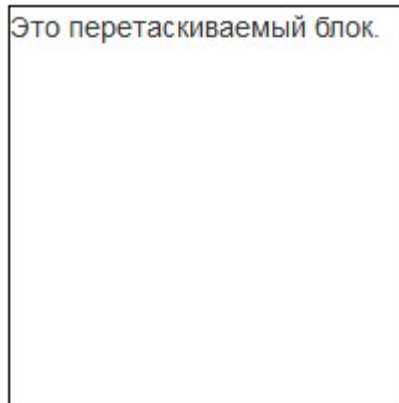
1.14. Плагин Draggable библиотеки jQuery UI

Плагин [Draggable](#), как и рассмотренный в ранее плагин Sortable, — это также одно из очень эффективных предложений от библиотеки jQuery UI. Данный плагин можно использовать, к примеру, вместе с указанным плагином сортировки, добавляя новые пункты меню в контейнер. Также можно встретить использование плагина перетаскивания элементов в интернет-магазинах, когда для того, чтобы добавить товар в корзину, его нужно перетащить мышкой в эту самую корзину.

В общем, сфера использования плагина Draggable ограничена лишь вашей фантазией. Давайте реализуем простейший пример перетаскивания элемента, используя [API плагина](#). Как обычно, нам потребуется некоторая разметка, в данном случае это должен быть элемент для перетаскивания. Пусть это будет обычный блок div:

```
<div class="draggable">
```

`<p>Это перетаскиваемый блок.</p>`
`</div>`



Осталось вызвать метод `draggable()` для элемента с одноименным классом:

```
$(function(){  
    $('.draggable').draggable();  
});
```

Собственно, все, можно проверять результат. Теперь блок можно свободно перемещать курсором мыши по странице.

Сейчас блок можно перетаскивать по странице в любом направлении. Если нам нужно ограничить перетаскивание только по оси X или Y, тогда для этого можно воспользоваться соответствующей опцией. Например, разрешим перетаскивать блок только по оси X:

```
$(function(){  
    $('.draggable').draggable({  
        axis: "x"  
    });  
});
```

Как видите, все достаточно просто, как и прочие плагины библиотеки jQuery UI.

1.15. Установка и инициализация плагина jCarousel

jCarousel это jQuery-карусель (слайд-шоу, слайдер), плагин для управления данными, отображаемых в виде горизонтального или вертикального списка. Данные могут быть представлены как обычный HTML контент или могут быть загружены с помощью AJAX. Списку можно задать прокрутку вперед или назад, с анимацией или без нее. Основным примечательным признаком карусели является его богатый функционал и широкие возможности настройки слайдера под свои нужды. Данные слайдер является "резиновым" и способен адаптироваться под любую верстку. Имеет возможность конфигурировать время задержки, прокрутки, тип анимации, размещать текстовые блоки и блоки с текстовым описанием слайдера уже из коробки. На основе плагина jCarousel можно легко создать полосы прокрутки как текстового наполнения, графического так и текстово-графического содержания.

Скачать вы можете на сайте GitHub: <https://github.com/jsor/jcarousel/releases>

Примеры реализаций на официальном сайте: <http://sorgalla.com/jcarousel/examples/>

Для работы карусели jCarousel необходимо создать следующую структуру HTML сущностей, подключить библиотеку jQuery, JS плагин и таблицу стилизации карусели(css файл):

```
<script type="text/javascript" src="/path/to/jquery.js"></script>
<script type="text/javascript" src="/path/to/lib/jquery.jcarousel.js"></script>
<link rel="stylesheet" type="text/css" href="/path/to/skin/skin.css" />
<div class="jcarousel"> <-----| Root element
  <ul> <-----| List element |
    <li>...</li> <---| Item element |
    <li>...</li> <---| Item element |
  </ul>
</div>
```

Вы можете создавать свою стилизацию карусели опираясь на файл css из примеров. Карусель jCarousel может состоять из различных HTML тегов, но обязательно должен сводиться к данной конструкции:

```
<div class="jcarousel"> <-----| Root element
  <div> <-----| List element |
    <p>...</p> <-----| Item element |
    <p>...</p> <-----| Item element |
  </div>
</div>
```

Для инициализация jCarousel необходимо разместить следующий код:

```
(function($) {$(function() {
  $(''.jcarousel').jcarousel({
    // Конфигурация инициализации
  });
});
```

Карусель jCarousel имеет ряд параметров, которые вы можете задать при инициализации карусели. Параметры определяют поведения карусели, типы анимации и описывают нестандартные ситуации.

Параметры инициализации карусели jCarousel

Параметр	Описание
<i>list</i>	<p>Задаёт указатель на элемент <code>list</code> в карусели <code>jCarousel</code>. Пример:</p> <pre>\$('.jcarousel').jcarousel({ list: '.jcarousel-list' });</pre>
<i>items</i>	<p>Задаёт указатель на элемент <code>item</code> в карусели <code>jCarousel</code>. Пример:</p> <pre>\$('.jcarousel').jcarousel({ items: '.jcarousel-item' });</pre>
<i>animation</i>	<p>Параметр задаёт скорость прокрутки слайдов карусели. Параметр принимает 2 значения: fast или slow. Также существует альтернативная конфигурация анимации <code>jQuery.animate</code>. Пример:</p> <pre>\$('.jcarousel').jcarousel({ animation: 'slow' }); \$('.jcarousel').jcarousel({ animation: { duration: 800, easing: 'linear', complete: function() {} } });</pre>
<i>transitions</i>	<p>Параметр определяет и жестко указывает использование аппаратных ускорителей анимации и CSS3. Принимает значения: true или false.</p> <p>Важно!!! <code>jCarousel</code> не умеет определять поддержку браузера стандарта CSS3. Вы должны самостоятельно определять поддержку CSS3 браузером. Пример:</p> <pre>\$('.jcarousel').jcarousel({ transitions: true }); \$('.jcarousel').jcarousel({ transitions: Modernizr.csstransitions ? { transforms: Modernizr.csstransforms, transforms3d: Modernizr.csstransforms3d, easing: 'ease' } : false });</pre>
<i>wrap</i>	<p>Параметр определяет поведения карусели при окончании цикла прокрутки. Принимает значения: first, last, both или circular. Пример:</p> <pre>\$('.jcarousel').jcarousel({ wrap: 'both' });</pre>
<i>vertical</i>	<p>Параметр задаёт ориентацию карусели как вертикальную. Если параметр не указан - карусель пытается автоматически определить ориентацию по соотношению сторон ширины и высоты. Принимает значения: true или false. Пример:</p> <pre>\$('.jcarousel').jcarousel({ vertical: true });</pre>
<i>rtl</i>	<p>Параметр задаёт режим прокрутки карусели <code>jCarousel</code> с права на лево. Принимает значения: true или false. Пример:</p> <pre>\$('.jcarousel').jcarousel({ rtl: true });</pre> <p>Пример с <code>dir</code> атрибутом:</p> <pre><div class="jcarousel" dir="rtl"> ... </div> <script> \$('.jcarousel').jcarousel(); </script></pre>

Обращение к методам

Query карусель jCarousel имеет возможность обращения к свойствам и методам классов jCarousel. В данном разделе мы рассмотрим самые популярные примеры. Полное описание классов и методов вы можете найти в документации на официальном сайте разработчика:

Документация на официальном сайте: <http://sorgalla.com/jcarousel/docs/>

Кнопки навигации карусели

Управление прокруткой карусели jCarousel предусмотрено при помощи кнопок навигации - управления. Для установки и инициализации кнопок управления каруселью необходимо создать следующую структуру HTML сущностей на своем сайте:

```
<!-- Wrapper -->
<div>
  <!-- Carousel -->
  <div class="jcarousel">
    <ul>
      <li>...</li>
      <li>...</li>
    </ul>
  </div>
  <!-- Controls -->
  <a class="jcarousel-prev" href="#">Prev</a>
  <a class="jcarousel-next" href="#">Next</a>
</div>

<script>
$(function() {
  $('jcarousel').jcarousel({
    // Конфигурация инициализации
  });

  $('jcarousel-prev').jcarouselControl({ target: '-1' });
  $('jcarousel-next').jcarouselControl({ target: '+1' });
});
</script>
```

Нумерация слайдов карусели - пагинация

jQuery карусель имеет возможность интеграции нумерации слайдов с навигационным функционалом. При клике на номерной инициал, jCarousel отображает элемент карусели, которому соответствует данный номер. Для установки и инициализации пагинации необходимо привести карусель к следующей структуре:

```
<!-- Wrapper -->
<div>
  <!-- Carousel -->
```

```

<div class="jcarousel">
  <ul>
    <li>...</li>
    <li>...</li>
  </ul>
</div>

<!-- Pagination -->
<div class="jcarousel-pagination">
  <!-- Pagination items will be generated in here -->
</div>
</div>

<script>
$(function() {
  $('.jcarousel').jcarousel({
    // Конфигурация инициализации
  });

  $('.jcarousel-pagination').jcarouselPagination({
    item: function(page) {
      return '<a href="#" + page + ">' + page + '</a>';
    }
  });
});
</script>

```

Автоматическая прокрутка карусели

Карусель имеет самое востребованное и популярное свойство - автопрокрутка содержимого карусели. При вызове данного метода вы можете задать свойства автопрокрутки, такие как: время задержки слайдов карусели; шаг прокрутки; автостарт. Для инициализации автопрокрутки карусели jCarousel необходимо вызвать метод jcarouselAutoscroll:

```

$(function() {
  $('.jcarousel')
    .jcarousel({
      // Конфигурация инициализации
    })

    .jcarouselAutoscroll({
      interval: 3000,      target:
      '+=1',
      autostart: true
    });
});

```


2. Самостоятельная работа

Задание делается самостоятельно.

Варианты заданий (**номер вашего варианта уточняйте у преподавателя**):

Необходимо разработать html-страницу с:

1. перечнем товаров, формой заявки
2. перечнем туров, формой заявки на тур
3. афишей мероприятий и формой покупки билета
4. авиарейсов и формой покупки авиабилета
5. ж/д расписаний и формой покупки ж/д билета
6. перечнем вакансий и формой подачи резюме
7. перечень типовых персонажей и формой создания нового персонажа (как в начале РПГ-игры)

Все варианты реализуются с использованием множества jQuery UI элементов.

Набор полей исходя из тематики варианта. Т.е., например, для авиабилетов будут поля город вылета, город прилёта. Для ж/д тип вагона и т.п. Приветствуется импровизация и креативные решения!

Обязательно должны быть:

- ползунки
- календарь
- аккордеон
- модальное окно формы
- выпадающий список с автозаполнением
- tooltip
- расчетная цена (рассчитывается из содержимого каких-то полей, например, количество человек * стоимость тура)
- модальное окно типа confirm, но реализованное на UI
- модальное окно типа alert, но реализованное на UI
- и др. на выбор

См. пример внизу

Вместе с отчетом должны быть и исходники.

