



ЭКСПРЕСС-КУРСЫ ВЕРСТАЛЬЩИКОВ

ТЕМА 4: Работа с JS скриптами

План

1. Введение;
2. Основы jQuery;
3. Принципы подключения скриптов. Анонимные функции;
4. Основные функциональные скрипты;
 - a. Многоуровневое меню;
 - b. Навигационная панель;
 - c. Фиксированная панель;
 - d. Слайдер;
 - e. Карусель;
 - f. Лайтбокс;
 - g. Почтовая форма;
 - h. Индикатор загрузки;
 - i. Параллакс;
 - j. Фоновое видео;
 - k. Табы;
 - l. Аккордион;
 - m. Google карта;
 - n. Кнопка “Вверх”;

1. Введение;

JavaScript является неотъемлемой частью хорошего продающегося шаблона. Скрипты позволяют значительно расширить функционал сайта, придать ему блеска и уникальности.

Большую роль в развитии легких в освоении и использовании скриптов сыграл JavaScript фреймворк под названием jQuery.

2. Основы jQuery

jQuery — это замечательный JavaScript Framework, который подкупает своей простотой в понимании и удобством в использовании.

В своей основе jQuery ориентировано на работу с элементами HTML страниц.

jQuery очень легко позволяет динамически манипулировать DOM структурой страницы.

Полностью изучать jQuery в рамках данной темы не имеет смысла ввиду обширности данной темы, к тому уже существует огромное количество информации по данной теме даже на официальном сайте - <http://jquery-docs.ru/>.

Тем не менее рассмотреть основные моменты все же стоит.

Рассмотрим понятие селектора jQuery.

Селектор jQuery очень схож с обычным CSS селектором. Разница только в том, что вызов селектора выполняется в специальной функции JQUERY() или \$().

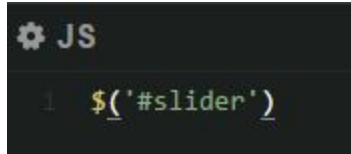
Символ \$ является алиасом для функции JQUERY.

Так, предположим что у нас в коде наблюдается следующая ситуация.

```
HTML
1 <section id="slider"></section>
2 <section class="well">
3   <div class="container">
4     <h2>Services</h2>
5     <div class="row">
6       <div class="grid_4">
7         <h3>Tuning</h3>
8         <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.
          Repellat consectetur quos tempora inventore a architecto impedit,
          delectus libero dignissimos dolore.</p>
9       </div>
10      <div class="grid_4">
11        <h3>Support</h3>
12        <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Cumque,
          eveniet! Debitis consequatur nisi similique aut aperiam, quia quos
          facilis asperiores!</p>
13      </div>
14      <div class="grid_4">
15        <h3>Marketing</h3>
16        <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ipsam
          reprehenderit deleniti itaque, autem quod officiis hic cum harum
          blanditiis necessitatibus.</p>
17      </div>
18    </div>
19  </div>
20 </section>
```

Нам необходимо получить доступ к секции с id='slider'.

В jQuery мы можем получить доступ к данному элементу с помощью следующего селектора.



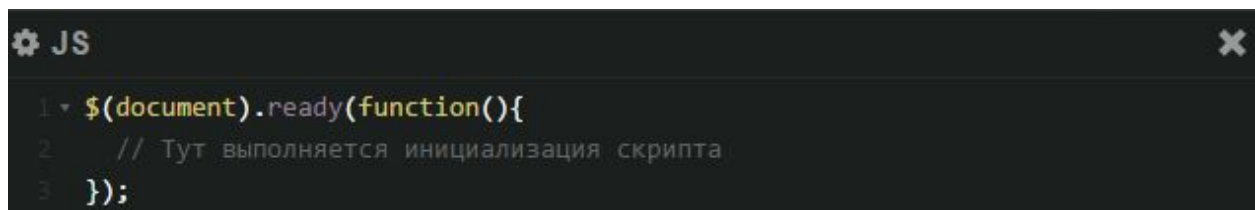
```
JS
1 $('#slider')
```

В качестве jQuery селектора можно писать любой доступный CSS селектор, будь то селектор по классу, атрибуту или чему либо другому.

В большинстве случаев, доступ к элементу HTML в JavaScript с помощью jQuery выполняется с целью инициализации определенного скрипта.

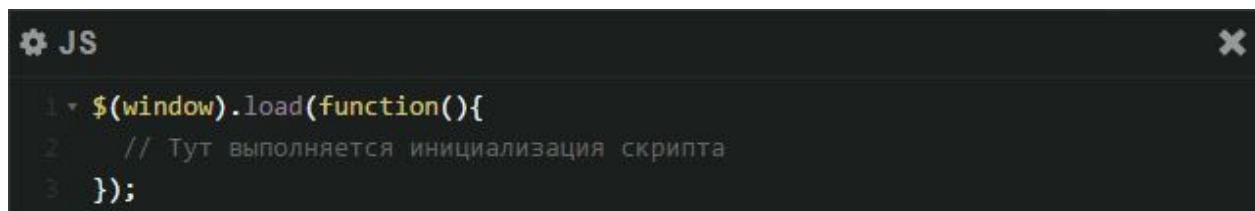
В таком случае инициализация скриптов должна выполняться по событию загрузки DOM структуры страницы или загрузки страницы в целом (включая файлы стилей, шрифты и т.д.).

Инициализация по событию загрузки DOM структуры страницы выполняется с помощью функции `ready()` для элемента `document`.



```
JS
1 $(document).ready(function(){
2     // Тут выполняется инициализация скрипта
3 });
```

Инициализация по событию загрузки страницы выполняется с помощью функции `load()` для элемента `window`.



```
JS
1 $(window).load(function(){
2     // Тут выполняется инициализация скрипта
3 });
```

Инициализация скриптов вне этих функций считается грубой ошибкой.

Также jQuery позволяет очень легко отслеживать различные манипуляции с элементами на странице.

Например, клик по элементу можно отследить следующими способом

```
JS
1 ▾ $(document).ready(function(){
2 ▾     $('.btn').click(function(){
3     console.log('Button is clicked');
4     })
5 });
```

В данном примере, при клике мышкой на элемент с классом `.btn` в консоль будет выведено сообщение "Button is clicked".

Список других функции для отслеживания манипуляций с элементом можно посмотреть тут: <http://jquery-docs.ru/Events/>.

Часто возникает ситуация, когда один и тот же участок кода, вынесенный в отдельный файл, должен выполняться только в том случае, когда на странице есть элемент с указанным в скрипте селектором.

Например, предположим, что мы хотим использовать обработчик для события по нажатию на кнопку с классом из предыдущего примера только в том случае, когда элементы с данным классом будут находиться на странице.

Данную проверку можно выполнить следующим способом.

```
JS
1 ▾ $(document).ready(function(){
2     var obj = $('.btn');
3
4     if(obj.length > 0){
5         obj.click(function(){
6             console.log('Button is clicked');
7         })
8     }
9 });
```

Поясним, как это работает. Дело в том, что селектор jQuery после его вызова возвращает массив из элементов, найденных на странице.

Соответственно, можно предположить, что если размер данного массива будет равен нулю, то по указанному селектору элементов найти не удалось, а значит они отсутствуют на странице.

3. Принципы подключения скриптов. Анонимные функции;

В наших шаблонах скрипты принято разделять на базовые и функциональные.

Под базовыми понимаются скрипты, которые необходимы для работы функциональных скриптов. Таким скриптами могут быть всевозможные библиотеки, фреймворки, полифилы и т.д.

Под функциональными понимаются скрипты, которые позволяют расширять функционал шаблона. Примерами данных скриптов могут послужить слайдеры, табы, меню, скрипты анимации и т.д.

В наших шаблонах в коде HTML страницы подключаются только базовые скрипты

Подключение и инициализация функциональных скриптов, при этом, вынесена в отдельный JS файл, под названием script.js, который подключается в самом конце HTML страницы, перед закрывающим тегом body.

В файле script.js скрипты подключаются внутри так званых анонимных функций.

Анонимные функции - это функции, которые не имеют названия и вызывают сами себя.

Также в файле script.js определены 2 функции под названием include() и isIE().

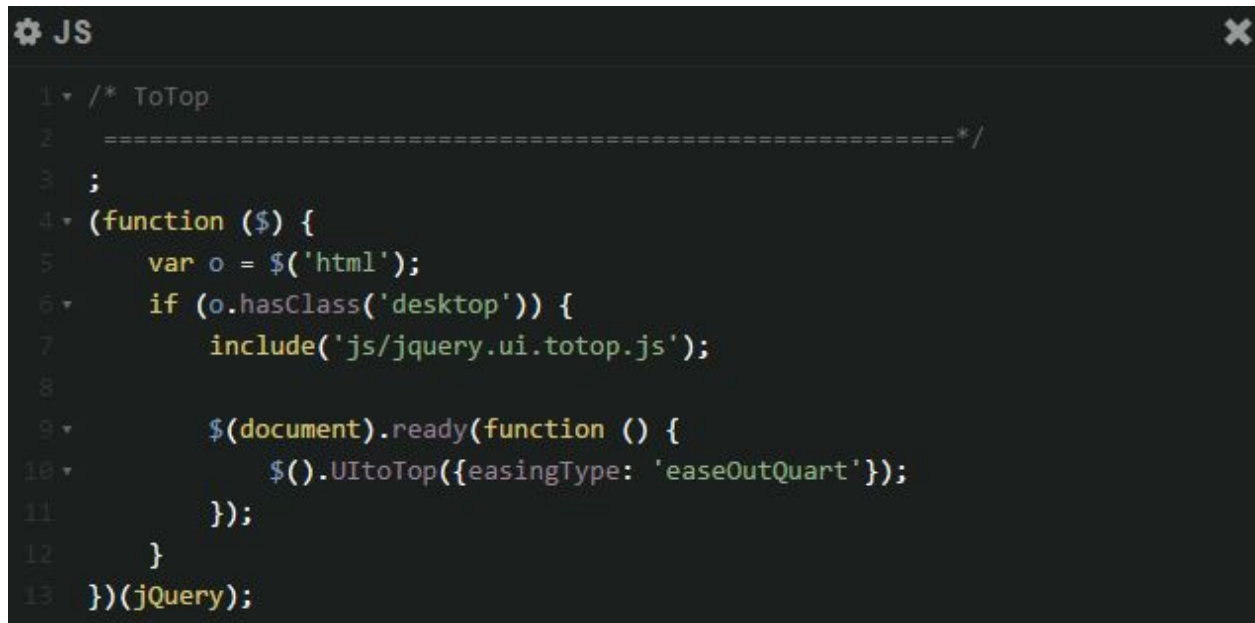
Функция include() позволяет подключать необходимые js файлы к странице.

Функция isIE() позволяет выполнить проверку, в какой версии браузера Internet Explorer запускается страница.

```
function include(scriptUrl) {
    document.write('<script src="' + scriptUrl + '"></script>');
}

function isIE() {
    var myNav = navigator.userAgent.toLowerCase();
    return (myNav.indexOf('msie') != -1) ? parseInt(myNav.split('msie')[1]) : false;
}
```

Рассмотрим принцип инициализации и подключения скриптов в наших шаблонах на примере подключения скрипта кнопки “Вверх”.



```
1  /* ToTop
2  =====*/
3  ;
4  (function ($) {
5      var o = $('html');
6      if (o.hasClass('desktop')) {
7          include('js/jquery.ui.totop.js');
8      }
9      $(document).ready(function () {
10         $.UItoTop({easingType: 'easeOutQuart'});
11     });
12 }
13 )(jQuery);
```

В данном примере, определяется анонимная функция `;(function ($){})(jQuery);`, выделенная комментарием с названием подключаемого скрипта.

Далее, внутри функции выполняется выборка элементов по селектору `$('html')`, то есть элементов соответствующих тегу `html`.

Затем выполняется проверка, имеет ли этот элемент класс `'desktop'` (Данный класс навешивается на элемент скриптом `device.js`, при инициализации секции `head`).

В случае, когда элемент имеет данный класс, выполняется подключение скрипта `'js/jquery.ui.totop.js'` с помощью функции `include()`.

После, по событию загрузки DOM структуры страницы, выполняется инициализация скрипта.

Подобным образом устроено подключение всех остальных скриптов в наших шаблонах.

4. Основные функциональные скрипты;

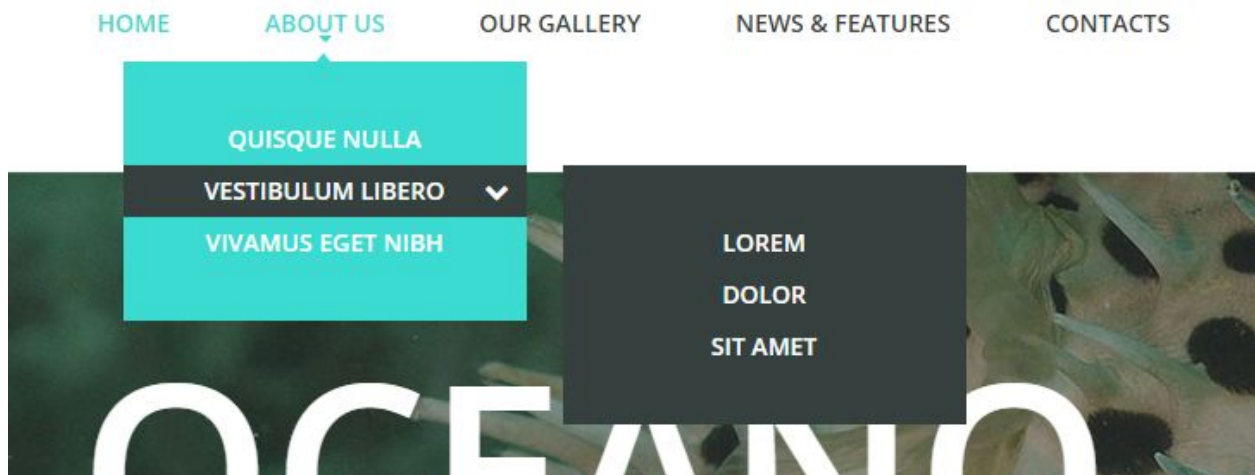
Теперь, имея необходимую информацию по принципах подключения скриптов в наших шаблонах, рассмотрим основные функциональные скрипты в наших шаблонах.

Все исходные файлы стилей и скриптов будут выданы в рамках домашнего задания.

а. Многоуровневое меню;

В наших шаблонах многоуровневое меню реализуется с помощью скрипта `superfish.js`.

Примером такого меню может послужить следующий участок шаблона.



Инициализацию данного скрипта необходимо производить с помощью следующего участка кода:

```
/* Superfish menus
=====*/
;(function ($) {
    include('js/superfish.js');
})(jQuery);
```

Данный скрипт производит инициализацию скрипта для списков с классом `"sf-menu"`.

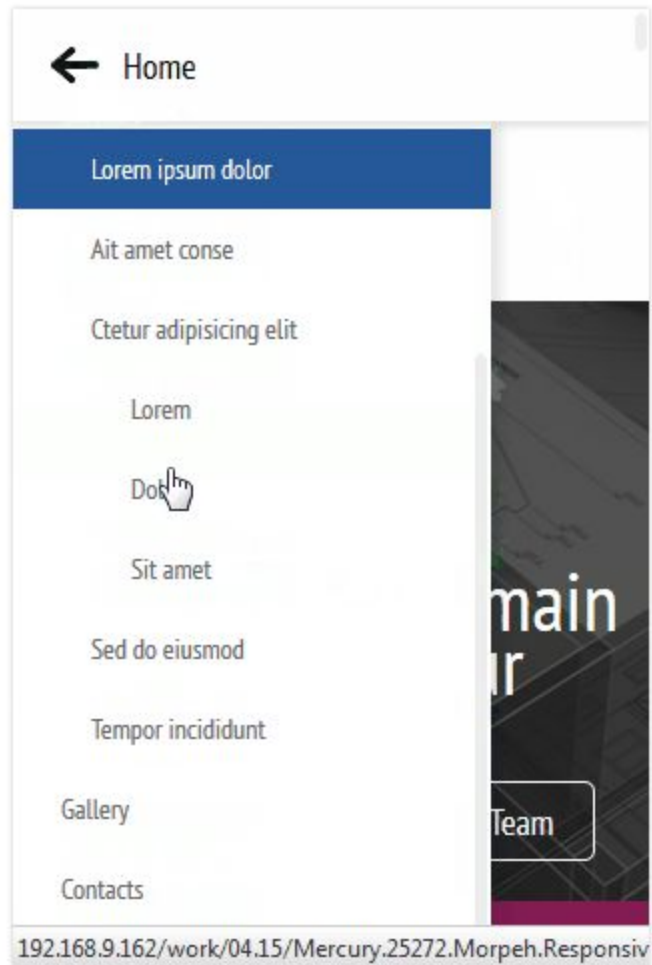
Соответственно, в HTML коде, применимо к данному примеру, также необходимо определять следующую структуру:


```
HTML
1 <nav>
2   <ul class="inline-list sf-menu">
3     <li><a href="#">Home</a></li>
4     <li>
5       <a href="#">About</a>
6       <ul>
7         <li><a href="#">Lorem ipsum dolor.</a></li>
8         <li>
9           <a href="#">Quo, cum, obcaecati.</a>
10          <ul>
11            <li><a href="#">Lorem ipsum dolor.</a></li>
12            <li><a href="#">Nihil, consectetur, exercitationem.</a></li>
13            <li><a href="#">Modi, blanditiis, eaque!</a></li>
14          </ul>
15        </li>
16        <li><a href="#">Eveniet, ab, impedit.</a></li>
17      </ul>
18    </li>
19    <li><a href="#">Our Gallery</a></li>
20    <li><a href="#">News&Features</a></li>
21    <li><a href="#">Contacts</a></li>
22  </ul>
23 </nav>
```

Дополнительная документация по данному скрипту доступна на сайте:
http://users.tpg.com.au/j_birch/plugins/superfish/options/

б. Навигационная панель;

К дополнению к многоуровневому меню в наших шаблонах необходимо использовать скрипт навигационной панели rd-navbar, которая заменяет обычное меню на мобильных разрешениях.



Подключать панель следует так:

```
/* Navbar
===== */
;(function ($) {
    include('js/jquery.rd-navbar.js');
})(jQuery);
```

А на меню следует указать data-type="navbar":

```
<nav class="nav">
  <ul class="sf-menu" data-type="navbar">
    <li class="active">
      <a href=".">Home</a>
    </li>
    <li>
      <a href="index-1.html">About us</a>
    </li>
    <li>
      <a href="index-2.html">Services</a>
    </li>
  </ul>
</nav>
```

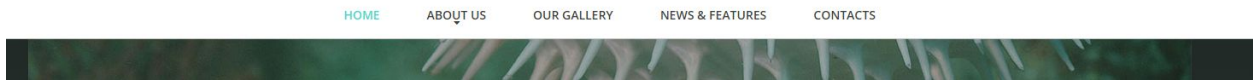
Скрипт дублирует меню в DOM, а отображение актуального меню происходит при помощи CSS. Таким образом, в случае необходимости, навигационную панель можно использовать как основное меню на всех разрешениях.

Полная документация по скрипту: <http://cms.devoffice.com/diversant/rd-navbar/>

с. Фиксированная панель;

Фиксированная панель в наших шаблонах реализуется с помощью нашего скрипта tmstickup.js

Примером, фиксированной панели может послужить все тот же шаблон.



В HTML необходимо размещать следующий код.

```
HTML
1 <div id="stuck_container" class="stuck_container">
2     <!-- Здесь ваш код -->
3 </div>
```

В файле script.js данную панель необходимо подключать следующим образом:

```
JS
1 /* Stick up menus
2  =====*/
3 ;
4 (function ($) {
5     var o = $('html');
6     if (o.hasClass('desktop')) {
7         include('js/tmstickup.js');
8     }
9     $(document).ready(function () {
10         $('#stuck_container').TMStickUp({})
11     });
12 }
13 })(jQuery);
```

Принцип работы данного скрипта состоит в следующем. Есть некоторый блок #stuck_container. При инициализации скрипта создается клон этого блока (.stuck_container.isStuck), со всем его содержимым, который отображается фиксированно в верхней части окна браузера, в том случае, когда оригинальный блок пропадает из viewport'a браузера.

Существует несколько правил по использованию данного скрипта.

- В стилях необходимо определять z-index для данной панели равный 999;
- При разрешении < 979px фиксированную панель необходимо скрывать с помощью свойства display: none !important;
- Высота фиксированной панели не должна превышать 75px;

Учитывая тот момент, что содержимое блока #stuck_container дублируется, в него необходимо помещать только те блоки, которые необходимо зафиксировать на панели.

d. Слайдер;

На данный момент слайдер в наших шаблонах реализовывается с помощью скрипта camera.js.

В качестве примера, можно предложить пример из этого шаблона.



Для успешного внедрения данного скрипта необходимо выполнить следующие действия.

В HTML коде необходимо указать следующую разметку:

```
HTML
1 <div class="camera_container">
2   <div id="camera" class="camera_wrap">
3     <div data-src="images/page-1_slide01.jpg">
4       <div class="camera_caption fadeIn"><!-- Контент для слайда 1 --></div>
5     </div>
6     <div data-src="images/page-1_slide02.jpg">
7       <div class="camera_caption fadeIn"><!-- Контент для слайда 2 --></div>
8     </div>
9   </div>
10 </div>
```

Для инициализации скрипта необходимо использовать следующий пример

```
JS
1 ▾ /* Camera
2   =====*/
3 ▾ ;(function ($) {
4   var o = $('#camera');
5   if (o.length > 0) {
6   if (!(isIE() && (isIE() > 9))) {
7     include('js/jquery.mobile.customized.min.js');
8   }
9
10  include('js/camera.js');
11
12 ▾ $(document).ready(function () {
13 ▾   o.camera({
14     autoAdvance: true,
15     height: '40.9375%',
16     minHeight: '300px',
17     pagination: false,
18     thumbnails: false,
19     playPause: false,
20     hover: false,
21     loader: 'none',
22     navigation: true,
23     navigationHover: false,
24     mobileNavHover: false,
25     fx: 'simpleFade'
26   })
27   });
28 }
29 })(jQuery);
```

Высоту камеры всегда необходимо указывать в процентах с помощью атрибута height.

Также необходимо указывать минимальную высоту слайдера с помощью атрибута minHeight. Значение минимальной высоты не должно превышать 350px.

Все остальные атрибуты указываются по необходимости.

Так как слайдер является компонентом с динамически загружаемым контентом, во время загрузки страницы может наблюдаться проблема с так званым “прыгающим контентом”.

Во избежание этой проблемы необходимо блоку с классом `.camera_container` указывать высоту в процентах от ширины по следующей формуле.

$$H / W * 100\%, \text{ где}$$

H - высота слайдера на PSD;

W - ширина слайдера на PSD;

Для этого можно воспользоваться особенностью свойства `padding`:

```
⚙ CSS (SCSS)
1 ▾ .camera_container {
2   position: relative;
3   padding-bottom: 64.83113069016153%;
4 }
```

Сам блок `#camera` при этом необходимо размещать абсолютно в области данного контейнера.

```
⚙ CSS (SCSS)
1 ▾ .camera_wrap {
2   display: none;
3   position: absolute;
4   left: 0;
5   top: 0;
6   width: 100%;
7   z-index: 0;
8   margin-bottom: 0 !important;
9   height: 100%;
10  background: #fff;
11 }
```

Учитывая наличие минимальной высоты для камеры, `padding` заданный в процентах необходимо заменить на значение минимальной высоты камеры в px при достижении необходимого разрешения.

Данное разрешение рассчитывается по формуле:

$\min H / (H / W)$, где

$\min H$ - минимальная высота слайдера;

H - высота слайдера на PSD;

W - ширина слайдера на PSD;

Полученные информацию можно представить в виде следующего набора стилей.

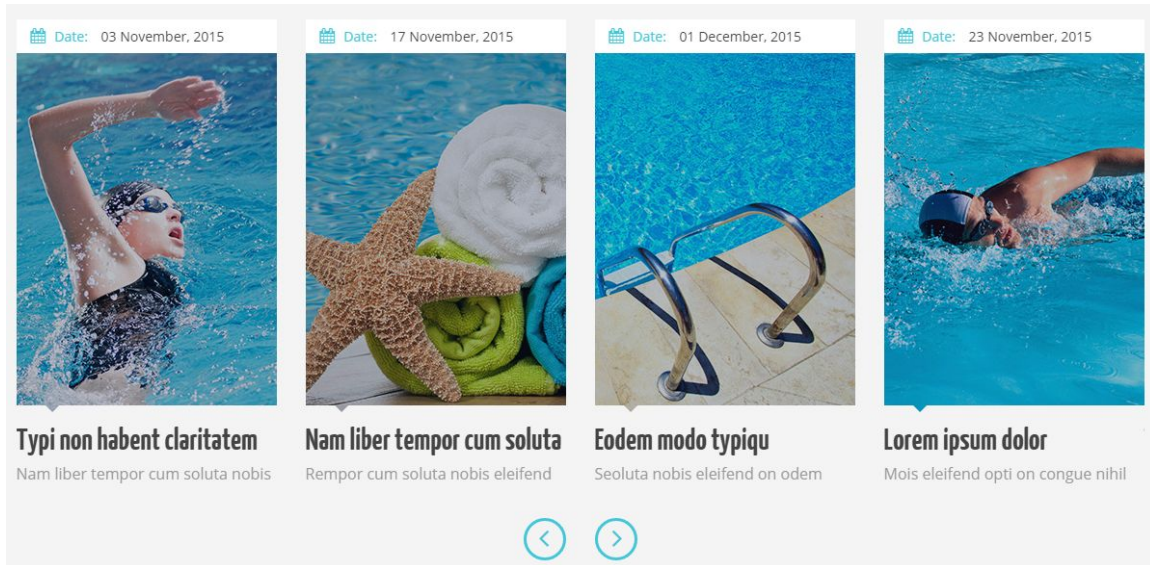
```
⚙ CSS (SCSS)
1 ▾ @media (max-width: 509px) {
2 ▾   .camera_container {
3     padding-bottom: 330px;
4   }
5 }
```

Дополнительную доментацию по данному скрипту можно почитать на официальном сайте: <http://www.pixedelic.com/plugins/camera/>

е. Карусель;

Скрипт карусели в наших шаблонах реализуется с помощью скрипта Owl Carousel 2.

Примером использования Owl Carousel может послужить следующий участок шаблона:



Для успешного внедрения данного скрипта в HTML коде необходимо описать следующую структуру:

```
HTML
1 <div class="owl-carousel">
2   <div class="item">
3     <!-- Ваш Код -->
4   </div>
5   <div class="item">
6     <!-- Ваш Код -->
7   </div>
8   <div class="item">
9     <!-- Ваш Код -->
10  </div>
11 </div>
```

В файле стилей карусели необходимо описать стили для навигации/пагинации в зависимости от дизайна шаблона.

Инициализацию скрипта необходимо производить с помощью следующего примера:

Документацию по данному скрипту можно посмотреть на офф. сайте:
<http://www.owlcarousel.owlgraphic.com/docs/started-welcome.html>.

```
JS
1 ▾ /* Owl Carousel
2 ▾ =====*/
3 ▾ ;(function ($) {
4 ▾     var o = $('.owl-carousel');
5 ▾     if (o.length > 0) {
6 ▾         include('js/owl.carousel.min.js');
7 ▾         $(document).ready(function () {
8 ▾             o.owlCarousel({
9 ▾                 margin: 30,
10 ▾                 smartSpeed: 450,
11 ▾                 loop: true,
12 ▾                 dots: true,
13 ▾                 dotsEach: 1,
14 ▾                 nav: true,
15 ▾                 navClass: ['owl-prev fa fa-angle-left', 'owl-next fa fa-angle-
16 ▾                     right'],
17 ▾                 responsive: {
18 ▾                     0: { items: 1 },
19 ▾                     768: { items: 2 },
20 ▾                     980: { items: 3 }
21 ▾                 }
22 ▾             });
23 ▾         });
24 ▾     })(jQuery);
```

f. Лайтбокс;

В наших шаблонах реализация лайтбокса выполняется с помощью скрипта fancybox.

Примером лайтбокса может послужить следующий участок шаблона:



Данное модальное окно открывается при активации соответствующей миниатюры в шаблоне.

Для успешного внедрения данного скрипта на странице в HTML необходимо обозначить следующий код:

```
HTML
1 <a class="thumb" href="images/page-1_img01_original.jpg">
2   
3   <span class="thumb_overlay"></span>
4 </a>
```

Подключение данного скрипта выполняется с помощью следующего участка кода.

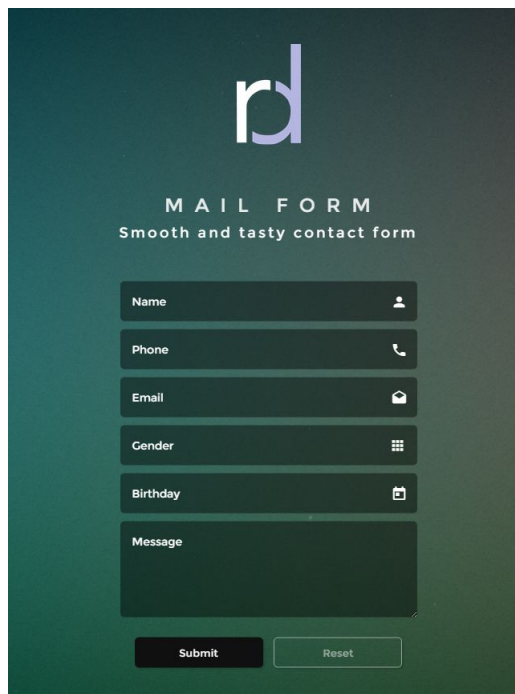
```
JS
1  /* FancyBox
2  =====*/
3  ;(function ($) {
4      var o = $('.thumb');
5      if (o.length > 0) {
6          include('js/jquery.fancybox.js');
7          include('js/jquery.fancybox-media.js');
8          include('js/jquery.fancybox-buttons.js');
9          $(document).ready(function () {
10             o.fancybox();
11         });
12     }
13 })(jQuery);
```

Официальная документация по данному скрипту доступна на сайте:

<http://fancyapps.com/fancybox/>

g. Почтовая форма;

Для создания контактных форм любого вида (формы подписки, формы заказа и т.д.) в наших шаблонах используется один скрипт - rd-mailform.

The image shows a contact form titled 'MAIL FORM' with the subtitle 'Smooth and tasty contact form'. The form is set against a dark teal background. It features a vertical stack of input fields: 'Name' with a person icon, 'Phone' with a telephone icon, 'Email' with an envelope icon, 'Gender' with a grid icon, and 'Birthday' with a calendar icon. Below these is a larger 'Message' text area. At the bottom, there are two buttons: a dark 'Submit' button and a light 'Reset' button. The 'rd' logo is at the top.

Подключать скрипт следует так:

```
/* Mailform
=====*/
;(function ($) {
    include('js/mailform/jquery.form.min.js');
    include('js/mailform/jquery.rd-mailform.min.js');
})(jQuery);
```

Разметка в HTML:


```

<form class='mailform' method="post" action="bat/rd-mailform.php">
  <input type="hidden" name="form-type" value="contact"/>
  <fieldset>
    <label data-add-placeholder data-add-icon>
      <input type="text"
        name="name"
        placeholder="Name"
        data-constraints="@LettersOnly @NotEmpty"/>
    </label>

    <label data-add-placeholder data-add-icon>
      <input type="text"
        name="email"
        placeholder="Email"
        data-constraints="@Email @NotEmpty"/>
    </label>

    <div class="mfControls">
      <button class="btn btn-lg btn-primary" type="submit">Submit</button>
      <button class="btn btn-lg btn-default" type="reset">Reset</button>
    </div>
    <div class="mfInfo"></div>
  </fieldset>
</form>

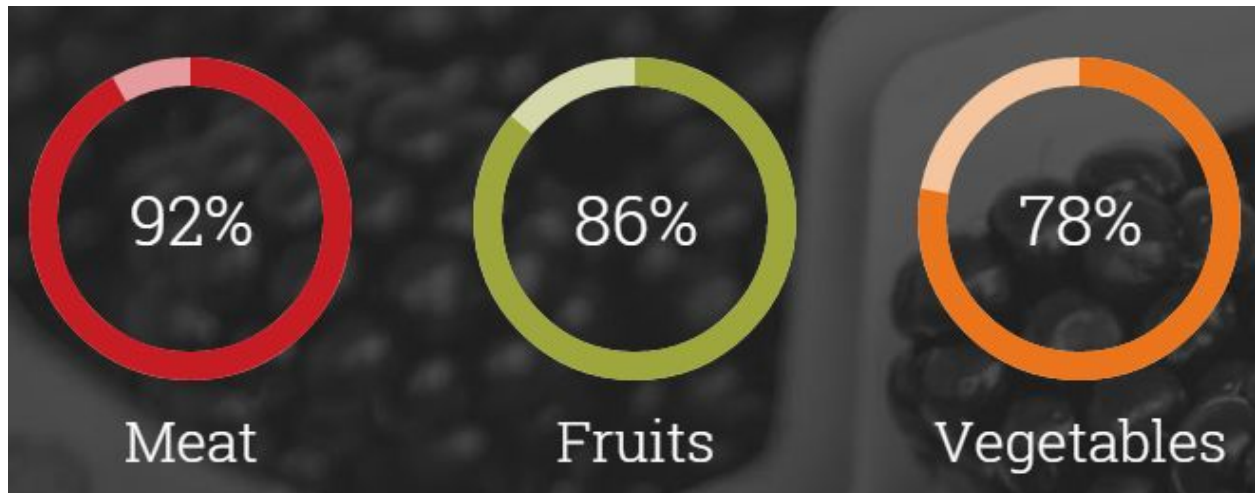
```

Полная документация по скрипту: <http://cms.devoffice.com/diversant/rd-mailform/>

h. Индикатор загрузки;

Для реализации радиального индикатора загрузки мы используем скрипт `radial-progress-bar`, плагин для `raphael.js`.

Примером данного скрипта может послужить следующий участок дизайна:



Данный скрипт требует подключения библиотеки `raphael.js`. Как уже упоминалось все базовые скрипты необходимо подключать в секции `head`.

```
HTML
1 <head>
2   ...
3 <script src='js/raphael/raphael.js'></script>
4   ...
5 </head>
```

Сам скрипт индикатора необходимо подключать в файле `script.js` с помощью следующего кода.

```

JS
1  /* Radial Progress Bar
2  =====*/
3  ;
4  (function ($) {
5      var o = $('.radial-progress');
6      if (o.length > 0) {
7          include('js/raphael/jquery.radial-progress-bar.js');
8      }
9  })(jQuery);

```

В HTML определяем наш индикатор с помощью следующего кода.

```

HTML
1  <div class="radial-progress"
2      data-border="15"
3      data-border-bg="#f7c6a1"
4      data-border-fg="#ed761f">
5      78%
6  </div>

```

Создаваемый индикатор автоматически растягивается по ширине контейнера в котором, лежит.

Рассмотрим атрибуты данного скрипта:

- data-border - задает ширину полосы загрузки;
- data-border-bg - задает цвет фона заднего плана индикатора;
- data-border-fg - задает цвет фона переднего плана индикатора;

Процесс загрузки индикатора автоматически подхватывается из числа определенного внутри блока с классом .radial-progress.

i. Параллакс;

Для реализации эффекта параллакса в наших шаблонах используется скрипт `jquery.rd-parallax.js`.

Как уже упоминалось в 1 теме, параллакс можно использовать для любого изображения заданного на всю ширину дизайн макета.

В файл `script.js` необходимо скопировать предложенный вариант инициализации скрипта:

```
/* Parallax
=====*/
;(function ($) {
    include('js/jquery.rd-parallax.js');
})(jQuery);
```

В HTML разметке на секцию, на которую мы хотим применить эффект нужно кинуть соответствующий класс и указать как `data-атрибут` изображение:

```
<section class="parallax well-md" data-url="images/parallax01.jpg" data-mobile="true">
  <div class="container">
    </div>
</section>
```

Полная документация по скрипту: <http://cms.devoffice.com/diversant/rd-mailform/>

j. Фоновое видео;

Для реализации фонового видео используется скрипт vide.js.

В файл script.js необходимо скопировать предложенный вариант инициализации скрипта:

```
/* VIDE
===== */
;(function ($) {
    var o = $('.vide');
    if (o.length > 0) {
        include('js/jquery.vide.js');
    }
})(jQuery);
```

В HTML разметке на секцию, на которую мы хотим применить эффект нужно кинуть соответствующий класс и указать как data-атрибут изображение:

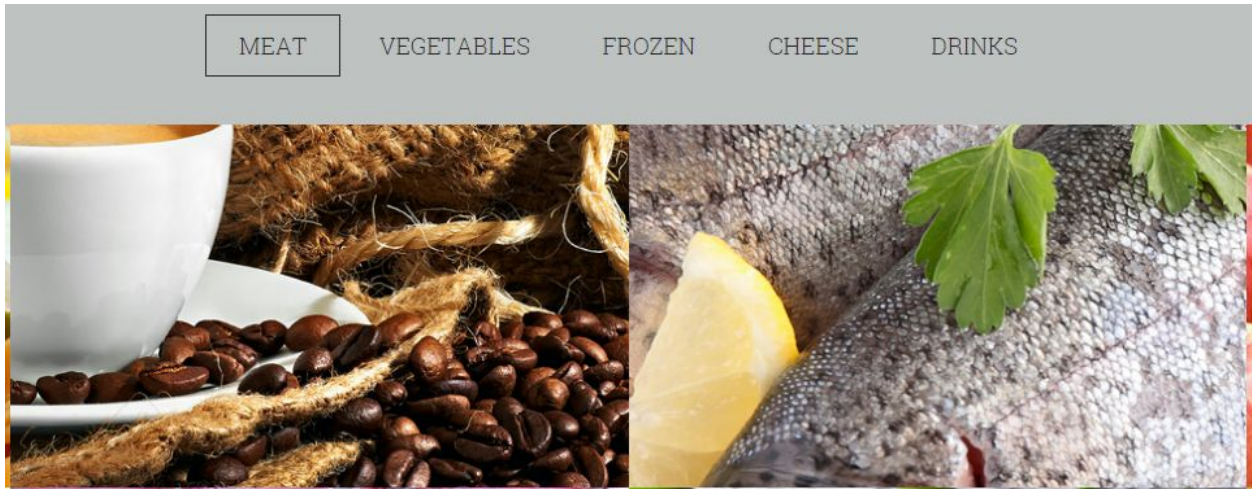
```
<!--=====
                                HEADER
=====-->
<header class="vide text-center" data-vide-bg="video/video-bg">

</header>
```

к. Табы;

Табы в наших шаблонах реализовываются с помощью jquery плагина responsive-tabs.

Примером использования данного скрипта может послужить следующий участок дизайна.



Для успешного внедрения данного скрипта, в HTML необходимо указать следующую разметку:

```
HTML
1 <div class="resp-tabs">
2   <ul class="resp-tabs-list">
3     <li>Название 1 вкладки</li>
4     <li>Название 2 вкладки</li>
5     <li>Название 3 вкладки</li>
6   </ul>
7   <div class="resp-tabs-container">
8     <div>
9       Контент 1 вкладки
10    </div>
11    <div>
12      Контент 2 вкладки
13    </div>
14    <div>
15      Контент 3 вкладки
16    </div>
17  </div>
18 </div>
```

Базовые стили для табов выглядят следующим образом:

```

1 ▾ /*===== Tabs =====*/
2 ▾ .resp-tabs-list {
3
4   }
5
6 ▾ .resp-tabs-list li{
7     cursor: pointer;
8   }
9
10 ▾ .resp-tabs-list li:hover{
11
12   }
13
14 ▾ .resp-accordion {
15     display: none;
16   }
17
18 ▾ .resp-tab-content {
19     display: none;
20   }
21 ▾ .resp-tab-content-active {
22     display: block;
23   }
24
25 ▾ @media (max-width: 767px) {
26 ▾   .resp-accordion {
27     display: block;
28   }
29
30 ▾   .resp-tabs-list {
31     display: none;
32   }
33 }

```

Инициализация скрипта выполняется следующим образом:

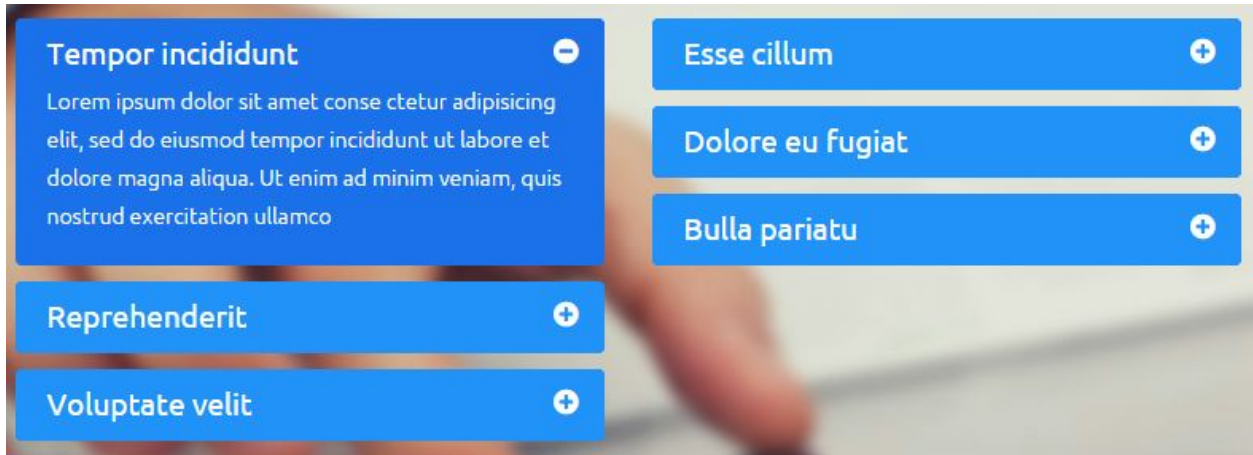
```
1 ▾ /* Responsive Tabs
2   =====*/
3   ;
4 ▾ (function ($) {
5     var o = $('.resp-tabs');
6 ▾   if (o.length > 0) {
7     include('js/jquery.responsive.tabs.js');
8
9 ▾     $(document).ready(function () {
10       o.easyResponsiveTabs();
11     });
12   }
13 })(jQuery);
```

Официальная документация к плагину доступна на сайте:
<http://webthemez.com/demo/easy-responsive-tabs/>

I. Аккордион;

Для реализации аккордиона в наших шаблонах используется скрипт `jquery.ui.accordion`.

Примером использования данного скрипта может послужить следующий участок дизайна:



Инициализация скрипта выполняется с помощью следующего кода:

```
1  /* Accordion
2  =====*/
3  ;
4  (function ($) {
5      var o = $('.accordion');
6      if (o.length > 0) {
7          include('js/jquery-ui-1.9.2.custom.min.js');
8          $(document).ready(function () {
9              o.accordion();
10          });
11      }
12  })(jQuery);
```

HTML и CSS структура зависит от ситуации и самого дизайна.

Официальная документация доступна на сайте: <http://api.jqueryui.com/accordion/>


```
1 ▾ /*===== GOOGLE MAP =====*/
2 ▾ .map_model {
3     height: 370px;
4 }
5 ▾ .map_model img {
6     max-width: none !important;
7 }
8 ▾ @media (max-width: 767px) {
9     .map_model {
10         height: 250px;
11     }
12 }
13 ▾ @media (max-width: 479px) {
14     .map_model {
15         height: 200px;
16     }
17 }
18 ▾ .map_locations {
19     display: none;
20 }
```

Инициализацию скрипта необходимо выполнять с помощью следующего кода:

```

1  ▾ /* Google Map
2      =====*/
3      ;
4  ▾ (function ($) {
5      var o = document.getElementById("google-map");
6  ▾  if (o) {
7      include('//maps.google.com/maps/api/js?sensor=false');
8      include('js/jquery.rd-google-map.js');
9
10 ▾    $(document).ready(function () {
11        var o = $('#google-map');
12 ▾      if (o.length > 0){
13        o.googleMap();
14      }
15    });
16  }
17 })
18 (jQuery);|

```

Ниже приведен список ответов на самые часто задаваемые вопросы:

В: Как отстайлить карту?

О: В инициализации скрипта нужно указать параметр styles и скопировать туда необходимые стили. (o.googleMap({ styles: *Массив ваших стилей* }));). Примеры стилизованных карт доступны по следующей [Ссылке](#).

В: Как указать координаты карты?

О: Нужно в <div id="google-map" class="map_model"></div> указать два data атрибута: data-x и data-y. Если не указать здесь данные координаты, они будут взяты по умолчанию со скрипта.

В: Как уменьшить/увеличить зум карты?

О: Нужно в <div id="google-map" class="map_model"></div> указать data атрибут: data-zoom. Если его не указать, он будет взят по умолчанию (14).

В: Как добавить маркер на карту?

О: Нужно в список <ul class="map_locations"> добавить новый пункт и определить там два data атрибута: data-x и data-y. Без этих координат скрипт проигнорирует этот пункт списка.

В: Как указать иконку маркеру?

О: Нужно в соответствующий пункт списка добавить 2 data атрибута – data-basic и data-active и указать пути к соответствующим базовой и активной иконкам. Если созданному маркеру не указывать путь к иконкам, то он будет указан по-умолчанию – gmap_marker.png и gmap_marker_active.png.

В: Как добавить всплывающее окно к маркеру?

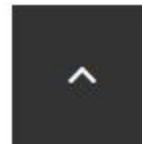
О: Просто пишем необходимый контент внутри пункта списка соответствующего маркеру.

п. Кнопка “Вверх”;

Кнопка “Вверх” в наших шаблонах реализуется с помощью скрипта `jquery.ui.totop.js`

Примером внедрения данного скрипта в шаблон может послужить следующий участок дизайна.

ociis natoque
ui. Fusce feugiat
n. Maecenas



Для успешного внедрения данного скрипта, необходимо в файле `script.js` определить следующий вариант инициализации.

```
JS
1  /* ToTop
2  =====*/
3  ;
4  (function ($) {
5      var o = $('html');
6      if (o.hasClass('desktop')) {
7          include('js/jquery.ui.totop.js');
8      }
9      $(document).ready(function () {
10         $.UItoTop({easingType: 'easeOutQuart'});
11     });
12 }
13 })(jQuery);
```

В основном файле стилей необходимо определить компонент `.toTop` с соответствующим оформлением.

При разрешении `< 1399` необходимо скрывать данный компонент, с помощью свойства `display: none! important;`