Правила на задачу верстки

Вер.1.8 <info AT hipot-studio DOT com>
от 17.09.2023

1.
При верстке используем `<!DOCTYPE>` для html5. Все новшества html5 рекомендуется использовать, если они работают во всех браузерах

2.
Верстаем под браузеры последней и предпоследняя версии (Firefox, Chrome, Safari). Во всех должно отображаться одинаково.

3.
Верстка должна быть адаптивной, т.е. работающей на любом экране в том числе мобильного устройства. При этом не должно быть горизонтальной прокрутки. Берем на данный момент минимальный размер экрана 360 x 667 точек.

4.
Все файлы (стили, скрипты, html) должны быть в кодировке utf-8, вложенные теги должны быть отбиты от родительских клавишей tab (или 4 пробела, как в настойках среды разработки).

Пример:

```
<!--noindex-->
<div class="col-xs-3 mobile-menu visible-xs">
    <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#navbar-header">
        <span class="sr-only"><?=GetMessage('TOGGLE_NAVIGATION')?></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
    </button>
</div>
<!--/noindex-->
```

5.
Структура папок и файлов при верстке:

```
./images/
./images_tmp/ -- временные файлы
./js/
./js/jquery/jqyery.js -- пример произвольной библиотеки js (опционально)
./js/scripts.js -- основные скрипты шаблона
./js/page1.js -- скрипты страницы page1 (опционально)
./template_styles.css -- основные стили верстки шаблона
./styles.css -- стили содержимого страниц (опционально)
./page1.html
./page2.html
...
```

В папку **images_tmp** складываем временные файлы (плейсхолдеры), которые при разработке будут заменены другими

файлами/блоками. Пример временных картинок: карта гугл, изображение в анонсе новости, картинка товара и т.д. Соответственно, все остальные «полезные» картинки, используемые в шаблоне, складывать в images. Все стили всех страниц необходимо размещать в файле `template_styles.css`. Подробно про формат этого файла читай ниже.

Примечание: имена html-страниц просто даны для примера: ./page1.html
Сами файлы нужно называть по-смыслу, напр, как названы файлы psd-дизайна либо страницы в Figma (news.psd —> news.html, Главная страница —>main_page.html)

6.
Как уже говорилось, список всех стилей должен быть расположен в файле template_styles.css. Порядок стилей в этом файле: общие, общие макета, частные страниц. Каждая группа css-стилей должна быть отделена комментарием, показывающим к какой странице эти стили относятся. Пример файла template_styles.css:

```
/***** main styles – общие стили ******/
body, html {height:100%;}
body {padding:0px; margin:0px;background:url(images/page_bg.jpg) center top no-repeat #fff;}

a  {color:#5d2d70; outline:0;}
img {border:none;}
form {margin:0px; padding:0px;}
textarea {resize:none;}

h1 {display:inline; padding:0px; margin:0px; font-size:25px; line-height:29px; text-shadow:1px 1px 0px #d1cfca;}
h2 {padding:10px 0px 20px 0px; margin:0px; font-size:18px; line-height:20px; color:#444; font-weight:normal;}
h3 {padding:0px 0px 18px 0px; margin:0px; font-size:13px; line-height:16px; color:#444; font-weight:bold;}

.dot0-0, .dot0-0-box {position:relative; left:0px; top:0px; width:100%; z-index:1;}
.dot0-0 {font-size:1px; line-height:1px; height:1px;}

/***** main maket styles – общие стили макета (колонки страницы, прижатый подвал и проч.) ******/
.fixer {margin:0px auto; width:900px;}
#all_content {height:auto !important; height:100%; min-height:100%;}
#all_content .fixer {padding:0px 0px 160px 0px; text-align:left; position:relative; z-index:1;}

/**** ДАЛЕЕ ИДУТ Частные СТИЛИ БЛОКОВ СТРАНИЦ, ВСЕ ДОЛЖНЫ БЫТЬ ПОДПИСАНЫ комментариями ******/

/*************** main page styles – стили для главной страницы *****************/
…
/*************** END main page styles *****************/

/*************** hotel_lent – стили ленты отелей *****************/

/*************** END hotel_lent *****************/
…
```

Примечание: Стили внутренних блоков для отдельных страниц можно выделить в еще один отдельный файл styles.css, как это предполагает вендор платформы.

7.
Короткие стили файла `template_styles.css` желательно писать в одну строку. Допускается перенос на новые строки, когда стилей очень много. Для удобства располагать параметры желательно в определенном порядке, например, сначала ширина, потом высота, а не наоборот, или, сначала высота, а потом отступы. Вот порядок, в котором следует располагать параметры стиля:
— позиционирование и расположение (position top left float)
— размерности (width height)
— отступы и интервалы (margin padding)
— границы и задний фон (border, background)
— шрифт, текст (color text-align text-decoration font-family font-weight font-size)
— всякие разности (cursor, ...)

Пример:

```
.header {position:absolute; top:10px; left:200px; width:50%; height:240px; border:2px solid #FFFFFF;}
```

8.
Краткая форма CSS. Все параметры стилей, имеющие свою краткую форму, желательно нужно в ней и писать. Обязательно это касаемо для таких, как padding, margin, background.

9.
Комментарии в html. Необходимо комментировать следующие вещи:

1/ **тематический блок** (т.е. напр. начало новости, конец новости), в виде <!— block —> <!— END block —>
2/ **закрывающий тег </tag>**, если его открывающий расположен более чем на 40+ строк кода. Вид комментария: `<!-- #div.root -->` (начинается с решетки и содержит информацию о том, какой тег тут закрываем).
Особенно это касается дивок, которые идут через весь макет (окатываются в хедере, закрываются в футере), т.к. зачастую в футере не понятно, за что конкретные закрывающие дивки отвечают.
3/ **изменения, внесенные в верстку запущенного сайта,** напр. ()

Пример:

```
<!-- main menu - комментарий о тематическом блоке -->
<ul class="mmenu">
    <li class="root">
        <div><a href="#">Каталог</a></div>
    </li>
    <li class="root">
        <div><a href="#">Учебный центр</a></div>
    </li>
    <li class="root">
        <div><a href="#">Услуги</a></div>
        <ul>
            <li><a href="#">Обучение</a></li>
            <li><span class="selected">Система дистанционного обучения</span></li>
        </ul>
    </li><!--#li.root - комментарий о том, какой тег тут закрываем -->
</ul>
<!-- END main menu - комментарий о тематическом блоке -->

<!-- Пример комментария переработки запущенного проекта  -->
<div class="video-container"><!--! добавить класс  video-container, убрать класс row-->
    <div class="video-item"><!--! добавить класс  video-item, убрать класс col-md-6-->
        <div class="video-wrapper"><!--! добавлена обёртка для iframe с классом video-wrapper-->
        </div>
    </div>
</div>
```

10.
Блоки с текстом нужно делать тянущимися по высоте, т.к. заранее трудно предсказать какой длины текст будет добавлен в эти блоки. Либо продумывать иные сценарии добавления более длинного контента в текстовые блоки, верстка в этом случае должна это учитывать.

11.
Очень рекомендуется **повторно использовать блоки на страницах сайта**, напр. **постраничная разбивка обычно делается одна на весь сайт**, поэтому она не должна зависеть от дивки, например, с лентой новостей. Так не надо делать: `.news_list .pager {}`, нужно делать `.page_content .pager {}` (где `.page_content` – это центральная часть шаблона всех страниц), либо просто `.pager {}`

12.
Заголовок страницы всегда должен быть в теге h1, **без использования любых атрибутов**, т.е. так делать нельзя: `<h1 class="className">Заголовок</h1>`
Это нужно для SEO-оптимизации.

13.
Проверять верстку нужно на разных размерах экранов, используя панель разработчика и включив адаптивный режим.

14.
Желательно использовать векторную графику SVG по максимуму (особенно для иконок и спрайтов).

15.

Необходимо по-максимуму использовать **css-спрайты**, особенно для иконок, т.е. все иконки можно собрать в один фай и разным позиционированием его в фоне добиться результата.

Внимание нужно уделить маркерам списков. Для них нужно делать отдельную картинку, т.к. спрайт предполагает, что элемент будет одной высоты, а элемент списка li может иметь любую высоту.

16.

Третье пространство. `z-index` не нужно задавать больше 100, обычно достаточно до 10. в противном случае конфликты с панелью упраления CMS вендора.

110 — для оверлея и попапов самое то (всплывающих форм).

16.

Альтернативный текст картинок. У всех картинок (img) обязательно нужно установить альтернативный текст с описанием того, что изображено на картинке. Можно установить `alt=""` (пустой), но он должен быть для поисковой оптимизации.

17.

Цвет фона по умолчанию. У тега body должен быть задан цвет фона. В противном случае при определенных настройках браузера будет проступать фон, заданный в настройках браузера. т.е.

```
body {background-color:#fff;}
```

18.

Жадные стили. Сточку `* {margin: 0; padding: 0;}` использовать нежелательно. Либо при ее использовании нужно хорошо проверить и обработать нюансы содержимого страниц.

Краткое пояснение: Постоянно возникают проблемы отбивки абзацев P друг с другом и с другими элементами. Ведь каждый браузер «по своему» строит отбивку у абзацев в зависимости от размера шрифта, теней, сглаживания и интерлиньяжа, а фиксированная отбивка всего этого не делает. Т.е. в большинстве случаев это мешает, чем упрощает жизнь.

19.

Вспомогательные блоки: оверлей, плавающие окошки, всплывающие формы регистрации и прочие блоки, болтающиеся в body нужно перемещать в самый конец страницы (перед закрывающим тегом /body).

Причина: поисковики получают первые N кб текста, лучше пусть они сразу видят «полезный» контент, а не эти блоки.

Для разработчика: еще круче, чтобы эти блоки подгружались аяксом, тогда вес страницы будет меньше.

20.

Поле поиска в шаблоне сайта с вводом запроса в него. Если в поле есть подсказка, убираемая при фокусе в поле — то ее цвет должен быть серым. Если мы ввели запрос в поле — то текст запроса должен стать черным.

21.

Еще материалы для прокачки.

Code Guide: Standards for developing consistent, flexible, and sustainable HTML and CSS.