Стандарт HTML-Верстки компании MSA-IT

Общие требования

- Соответствие макету (шрифты, отступы, цвета, отсутствие лишних скроллбаров), Pixel Perfect (для блоков и графики).
- Кроссбраузерность, кроссплатформенность: согласно ТЗ, по умолчанию верстаем от IE10+
- Максимальная валидность кода (W3C)
- Адекватная скорость загрузки и рендеринга (не менее 80 пунктов в Google Page Speed)
- Семантичность (удобная и логичная разметка)
- Семантика HTML5
- Отсутствие «брошенного» текста – текста, не обрамленного в теги
- SEO-friendly - h1-h6, alt у картинок, title у ссылок
- Устойчивость к "разуплотнению" длинным текстом (там, где есть пользовательский ввод)
- Кодировка по умолчанию – UTF-8 без BOM, если это не оговорено отдельно
- Удалять неактуальные/временные части кода перед сдачей проекта (console.log, закомментированные варианты реализации и тп)
- Удалять неактуальные/временные файлы (изображения, документы и т.д.) перед сдачей проекта
- Для каждого проекта ведется GIT-репозиторий
- Обязательное использование одного из препроцессоров (по умолчанию SASS(SCSS))
- Сборка проекта осуществляется с помощью GulpJS
- Нейминг по умолчанию согласно БЭМ-методолгии.
- Каскадность при БЭМ именовании допустима при сбросе стилей, при динамическом изменении стилей, либо при использовании редактируемого контента (например, через встроенный редактор Битрикса). Для реализации необходимо использовать родительский класс для всей редактируемой области, например, edit-content, от которого будет строиться каскад.

Структура файлов на проекте

.dev_tools ***(сервисная папка)***

- gulpfile.js
  - node_modules

build ***(собранный проект)***

- index.html
- ...
  - css
    - style.css
  - js
    - main.js
  - libs
    - normalize.css
    - jquery.min.js
    - ...
  - img
    - logo.png
    - ...
  - fonts
    - icomoon
    - ...

src ***(исходники проекта)***

- index.html
- ...
  - template
    - inc.header.html
    - inc.footer.html
    - ...
  - style
    - style.scss
    - template
      - header.scss
      - footer.scss
      - ...
    - global
      - _variable.scss
      - _mixin.scss
      - ...
  - js
    - main.js
    - _popup.js

- _menu.js
- ...

- libs
  - normalize.css
  - jquery.min.js
  - ...
- img
  - logo.png
  - ...
- fonts
  - Icomoon
  - ...

Требования к HTML

- Разметка HTML5
- Обязательно использовать отдельные html-файлы для повторяющихся компонентов. Например, с помощью php include или gulp rigger
- Именование классов по умолчанию должно вестись в терминологии БЭМ
- В селекторе сначала идет атрибут «class=””», затем «id=””», затем все остальное
- Значения атрибутов должны быть заключены в двойные кавычки
- Названия классов и id должны по смыслу соответствовать применению. Например, header, menu, footer, news. Транслит в названиях недопустим

Не правильно	Правильно
novosti	news
box-3	news__item_size_3

- Названия блоков и элементов должны соответствовать их назначению и функции, а не тому, как они отображаются. Если блоки выполняют разные функции, но выглядят одинаково, их нужно оформить как один блок с нейтральным названием, отражающим их сходство.
- Не допускается использование двух одинаковых id на странице
- Все ссылочные блоки должны являться ссылками (a href="#", а не div class="onclick"). Даже когда внутри тега должны быть блочные элементы (http://html5doctor.com/block-level-links-in-html-5/)
- Все элементы формы должны быть обернуты в form. В верстке не должно быть отдельно стоящих input, select, etc
- С помощью тега img вставляются только изображения, относящиеся к контенту - фотографии, изображения товаров, лого и.т.д. Интерфейсная графика - только в background / иконочный шрифт
- Соблюдать структуру иерархических отступов. Квант – один таб
- Кастомные атрибуты задаем с префиксом data-
- Если нужно избежать пробелов между тегами, между ними вставляется пустой HTML-комментарий
- Разделять общие HTML блоки комментариями. Так:
  <!-- BEGIN FOOTER -->
  <!-- END FOOTER →

Требования к CSS

- Стили присваиваем только для классов. Не вешаем стили на идентификаторы!
- Общие используемые переменные и миксины должны быть вынесены в отдельный файл и подключаться по мере необходимости.
- Файлы SCSS должны разбиваться по компонентам.
- Общие файлы SCSS собирать с помощью @import или gulp rigger
- К конечной странице подключать скомпилированный и минифцированный CSS
- Все нестандартные шрифты делаем по умолчанию через @font-face
- При использовании CSS хаков оставлять комментарии, что это и для какого браузера
- Хаки для IE 6-8 выносить в отдельный файл(ы) и подключать условным комментарием
- Если сайт будет адаптивный, и для мобильных устройств будет использоваться meta-tag viewport, то значение width должно быть «device-width», а не константа
- Использовать спрайты в тех местах, где они принесут пользу. Особенно это касается элементов с состоянием hover, clicked и прочих динамических элементов.
- Не должно быть инлайновых стилей (div style="font-weight:bold") кроме задаваемых динамически, либо через админку. Подключение CSS файлов должно быть с помощью одного файла (style.css), который содержит @import других стилевых файлов
- Использовать “rgba” модель для свойств поддерживающих прозрачность
- Не использовать классы с префиксом «js-» для стилизации
- Каждое правило должно располагаться в начале новой строки. После правила следует пробел, фигурная скобка, далее на новых строках правила и закрывающая фигурная скобка на новой строке. У свойства после названия двоеточие, пробел, затем значение. Отступы – 1 таб.

Плохо	Хорошо

.article__item, .article__box{
 	
color:#777;}
	.article__item,
	
	.article__box {
	
	color: #777;
	
	}

- В начале идут свойства, отвечающие за поведение блока, а затем - за оформление

Приблизительная схема расположения свойств	Плохо	Хорошо

{
Позиционирование
Параметры блока
Размеры
Таблицы / списки
Свойства текста
Шрифт
Цвет
}
	.article {
	color: #777;
	font-size: 16px;
	font-style: italic;
	line-height: 30px;
	left: 0;
	margin-top: -30px;
	position: absolute;
	text-align: center;
	top: 50%;
	width: 100%;
	}
		.article {
		position: absolute;
		top: 50%;
		left: 0;
		width: 100%;
		margin-top: -30px;
		line-height: 30px;
		text-align: center;
		font-size: 16px;
		font-style: italic;
		color: #777;
		}

Требования к JS

- Все сценарии анимации перед началом верстки должны быть зафиксированы и утверждены. Даже если анимация делается на усмотрение разработчика – обязательно нужно зафиксировать предполагаемый сценарий анимации, чтобы избежать переделок.
- Отсутствие необоснованного JS в разметке страницы
- Для классов, используемых в JS использовать префикс «js-»
- Использовать для старта jquery-выборки только классы с префиксом js или id элемента (для ускорения выборки). Исключения - код, на который нельзя повлиять (например, сгенерированный плагином)
- Кэшировать jQuery-объекты
- Называть jQuery-объекты с $
- Назвать булевые переменные с is-
- Не оставлять в коде console.log: он может присутствовать только в момент, когда вы непосредственно работаете с кодом. Также он может использоваться при использовании debug-переменной
- debug-переменная может находится в положении “true” при разработке. При передаче верстки на интеграцию, ревью или перед показом ее нужно отключать
- не делать длинную цепочку из .parent().parent()... без крайней необходимости
- Если внесены изменения в библиотечный код, дописывать к имени ".edit", комментировать что и для чего вы поменяли. Оригинал оставляем рядом с измененным скриптом.
- Дописывая чужой код, придерживаться исходной стилистики. Если нужно отметить место, которое писали именно вы, напишите комментарий
- Для инпутов использовать аттрибут placeholder=”Your name”, а не value=”Your name” с JS обработчиком; для старых браузеров подключать placeholder.js
- Не используем onclick – используем события
- При вызове функции в файле, отличном от местоположения функции – комментируем ее местонахождение
- Не должно быть ошибок, связанных с отсутствием подключенной библиотеки (читай: не должно в коде быть вызова функций библиотек, которые не подключены)

Инструменты для проверки

- Оценка производительности страницы https://developers.google.com/speed/pagespeed/insights?hl=ru
- Оценка валидности верстки http://validator.w3.org/

Референсы по технологиям

- Соглашение по именованию БЭМ https://ru.bem.info/method/naming-convention/
- Сборка проектов с помощью gulpJS http://gulpjs.com/
- Работа с препроцессором SASS (SCSS) http://sass-scss.ru/