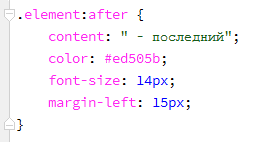
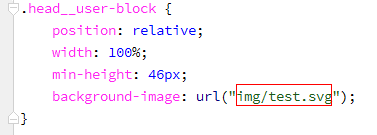
**Регламент для верстки макетов**:

1. По минимуму использовать float позиционирование + clearblock, заменить на flex, Никаких display:inline-block, font-size:0
2. Вся работа JS должна осуществляться на чистом JS + JQuery, возможно использование сторонних библиотек по согласованию, например, для слайдеров и прочего. Альтернативны библиотеки, аналоги JQuery не использовать.
3. Взаимодействие с элементами. Рендеринг должен максимально производиться на стороне CSS, а не манипуляции с DOM объектами. Например, если мы хотим скрыть кнопку, мы добавляем класс .hide, если мы хотим ее показать вновь мы убираем этот класс.
4. Не использовать текстовую и любую другую информацию, заданную непосредственно в JS, CSS файлах. Например, нельзя использовать такую конструкцию:  
   
5. Помнить о том, что макет будет изменяться со временем. Код должен быть удобен для просмотра и внедрения изменений. Так же прокомментирован.
6. Микроразметка. Нас всех страницах сайта должна присутствовать Микроразметка Open Graph. А также должна присутствовать микроразметка Schema.org, для всех элементов, Используемых на сайте, доступных в данной разметке. Меню, хлебные крошки, информация о компании, изображения, отзывы, товары и стоимость и прочее.  
   Подробная информация о разметке [есть здесь](https://yandex.ru/support/webmaster/schema-org/what-is-schema-org.xml), [сайт проекта](http://schema.org/docs/schemas.html).
7. Умение изменять представление элементов. (Важно!) Очень часто в дизайне встречаются элементы, которые ведут себя по-разному в зависимости от разрешения экрана, в частно речь идет об слайдерах. Реализация появления/исчезновения слайдера с помощью добавления скрытых блоков с контентом неприемлема. Слайдер должен включаться и отключаться при необходимости.
8. Верстка под Битрикс. Влечет за собой ряд особенностей, которые нужно учитывать в процессе разработки. Первой особенностью является разделение блоков страниц на секции, каждая логическая часть является обособленным элементом в системе и называется «Компонент». Представляет собой набор изображений, скриптов JS стилей, php/html кодов. Из этого следует, что код должен быть разделен на аналогичные смысловые блоки и такое разделение должно учитываться привёртке. Для дальнейшей работы программиста все логические блоки из html,js,css,img и прочего должны быть разделены на папки с понятными и логичными названиями, в дальнейшем они будут переработаны в «Компоненты». Примером логического блока может быть «Список новостей», «Новость детально», «Слайдер изображений», «Текстовая секция» и т.д.
9. Состояния элементов. Есть несколько состояний элемента, то все эти состояния должны быть описаны и продемонстрированы на этой же странице в доступном для восприятия виде. Если это элементы одного компонента, это должно быть так же определено, например, комментариями. Все нюансы верстки должны быть описаны в комментариях, все что сказано устно или в чате – не удобно, может забыться. Только то что есть в комментариях к макету учитывается, если что-то важное не описано в комментарии и описано где-то еще, считается не описанным.
10. Верстка делается под CMS. А это означает, что тексты и прочие элементы могут быть в последствии изменены, добавлены новые элементы, удалены, эти изменения должно быть удобным. Неприемлемо использование фиксированной ширины для размеров переключателей, кнопок и прочего, если это не является единственным возможным решением реализации.   
    В дизайне макета размеры блоков в большинстве случаев указаны верно. Не нужно указывать в текстовых блоках высоту фиксированную, потому что это лишний код, который усложняет работу с версткой.
11. Код должен быть максимально валидным.
12. Во всех макетах используется большое количество векторных изображений, они должны быть экспортированы как SVG код и интегрированы в верстку, если элементы должны быть анимированы, то анимация производится с помощью CSS. Использование SVG как изображения должно быть обусловлено, невозможностью использования SVG кода, вот пример подключения SVG как изображения  
      
    В 99% случаев SVG можно интегрировать в верстку и все будет отлично. SVG выгружается с лишними тегами группировки и не нужными ID, их нужно удалить перед внедрением.  
    Единичные изображения или другие элементы элементы дизайна, которые не предполагают частую смену, например, логотип должен быть реализован с помощью подключения SVG как изображения для его удобной замены в случае необходимости (нюанс 1С-Битрикс).
13. Спрайты изображений не использовать. Используется CMS и компоненты, а каждый компонент обособленная логическая единица и спрайты не получится использовать. (Ну и я простив) к тому же вектор хорошо ресайзится.
14. Проверка верстки. Вся верстка будет проверяться на соответствие указанным выше требованиям. Так же могут быть найденные в коде конструкции, которые не были регламентированы, но также будут не приемлемы. Все что будет не соответствовать требованиям, нужно будет переделать. Чтобы не срывать сроки из-за переделок и не мучать ими вас, макеты нужно **присылать постранично.**
15. Отделение общий CSS,JS, шрифтов и изображений. Мы используем компоненты, которые хранят в себе всю эту информацию. Но есть скрипты, стили и прочее, что используется на всем сайте, код и прочее, используемое на большинстве страниц сайта, которое нельзя отнести к компоненту должно быть выделено в отдельные файлы и подключаться на всех страницах в head
16. Ресайз и поведение элементов. В верстке может быть различное количество состояний в зависимости от разрешения экрана. Важно чтобы как бы мы не ресайзили окно браузера, сайт всегда отзывался правильно на изменение, не появлялись артефакты и не рушилась работа сайта.
17. Кроссбраузерность.   
    Сайта должен быть одинокого работать в следящих десктопных браузерах:
    1. Chrome
    2. Opera 12.4+
    3. Mazilla
    4. Internet Explorer 10+
    5. Edge
    6. Яндекс браузер
    7. Safari 6+

В мобильных браузерах (Планшеты и Смартфоны):

1. Safari
2. Chrome
3. Opera
4. Mazilla
5. Стандартный браузер Android
6. Работа с программами для верстки (плагинами). Бывают ситуации, когда предложенный вариант CSS кодов не самый оптимальный, это касается прозрачности, шрифтов и прочего, если есть отличие между макетами и прилагаемым вариантом, действовать на свое усмотрение. Оптимизировать стили.
7. Наследование. Особенно касается шрифтов, высоты строки и всего остального, что касается типографии. Перед началом работы над версткой, выделить повторяющиеся стили и вычленить их в отдельные классы для вашего удобства, легкости адаптации шрифтов и быстрой возможности изменить размеры шрифтов, сами шрифты, не прочесывая каждый макет, а в будущем каждый компонент на всех страницах.
8. Изменяемый контент, классы и стили. Вся информация, предполагающая изменение пользователем, тексты изображения и прочее, не должны иметь какие-то классы и прочие особые атрибуты. У них может быть родительский блок, с помощью которого задается стиль, например, для изображения или ненумерованного списка.
9. Шрифты, должны быть прогнаны через конвертеры шрифтов, на [подобии](https://www.fontsquirrel.com/tools/webfont-generator).   
   Проверка шрифта, используемого в макетах на возможность использования, проверка наличия артефактов в сочетании букв и прочего. В случае обнаружения обратиться к постановщику задачи, но прежде попробовать несколько конвертеров шрифтов.
10. Сторонние библиотеки и прочие ссылки на другие ресурсы. Верста должна быть самодостаточна, без подключения библиотек и прочего с сторонних ресурсов, это может в будущем повлиять на работу SSL и сам ресурс, с которого берется ссылка может прекратить свое существеннее. Исключениями могут являться «Карты», и другие сервисы, которые предполагаю подключение библиотек.
11. Анимирование. Даже если элемент не предполагает анимирование, нужно учитывать возможность ее возникновения. Возьмем для примера SVG, мы сделали стрелочку у слайдера и ссылаемся на файл SVG, если мы заходим его анимировать, нам придется либо внедрять все же этот SVG в код, либо использовать спрайты. Спрайты не используются, переделывать код – это лишняя работа. Потому нужно заранее это продумывать.
12. Использование современного кода, касательно CSS, JS и HTML. Код должен быть актуальным, не нужна поддержка IE 6 и прочего. Все в рамках регламента кроссбраузерности, не нужно использовать древние конструкции кода и стилей.
13. Оправданность использования JS. Если использование JS не оправдано или куски кода повторяются часто, значит вы что-то делаете не так и это придется переделывать.
14. Спорные моменты. Бывают ситуации, когда одно и тоже можно сделать разными способами и что-то кажется не таким существенным. Если постановщику задачи это кажется существенным, это придётся переделать. Аргументы всегда воспринимаются положительно, без тирании.
15. Header, Footer и Рабочая область страницы - отдельные подключаемые файлы, это нужно учитывать при формировании кода.
16. [Руководство](https://dev.1c-bitrix.ru/learning/course/?COURSE_ID=43&CHAPTER_ID=04640&LESSON_PATH=3913.4776.4640) по оформлению кода для Битрикс.
17. [Что такое компоненты](https://dev.1c-bitrix.ru/learning/course/index.php?COURSE_ID=43&CHAPTER_ID=04565&LESSON_PATH=3913.4565) (Логические/Смысловые блоки)
18. Рекурсивные включаемые области [это](https://dev.1c-bitrix.ru/learning/course/index.php?COURSE_ID=34&LESSON_ID=1879) – блок в котором может быть подключен компонент и/или статичный html код.
19. [Включаемые области](https://dev.1c-bitrix.ru/learning/course/index.php?COURSE_ID=43&CHAPTER_ID=04881&LESSON_PATH=3913.4564.4881) – это кусок кода в котором может быть php или html код