**Проектная работа «Оно тебе надо» от Яндекс практикума**

<https://habr.com/ru/articles/866514/?utm_source=habrahabr&utm_medium=rss&utm_campaign=866514> December 14, 2024

Всем привет, меня зовут Ольга. Я начинающий фронтенд разработчик, а также домохозяйка и в прошлом руководитель высшего звена. Я обучаюсь фронтенд разработке на Яндекс Практикуме и хочу поделиться опытом разработки первого проекта.

Начало

Для начала стоит отметить, что обучение построено на коротких читаемых лонгридах. Сотрудники объясняют это тем, что информация лучше воспринимается и откладывается в голове, когда ее преподносят в читаемом формате, нежели в формате видео. Поэтому найди любое CSS свойство, которое забыл, не так уж и трудно, потому что, не надо хотя бы смотреть видео. Также одним из уроков был разбор полноценного кейса прошлых лет, чтобы ученики максимально точно проанализировали процесс разработки HTML страницы (как лучше всего называть классы, код-стайл, как работать с Pixel Perfect и много чего другого). Я считаю, что этот урок очень полезный и важный для начинающих (но поскольку я далеко не начинающий в мире информационных технологий, особого внимания на разборе кейса я не заострял).

Теперь можно приступать к разработке. Для ознакомления прикладываю ссылку на макет:

Казалось бы, достаточно много элементов и вообще не понимаешь с чего начать. Нам же советуют:

1. Найдите повторяющиеся компоненты. Например, блок с адресом, оверлей поверх изображений — эти блоки, скорее всего, содержат одинаковый CSS-код. Запишите свои наблюдения, они пригодятся.
2. Посмотрите, какие в макете размеры и толщина шрифтов.
3. Отметьте, что на странице один шрифт.
4. Выпишите элементы, которые кажутся необычными и вызывают вопросы. Эти вопросы помогут отделить сложное от простого уже на старте: 1. Что за странные отступы между буквами в первой секции? 2. А что за странные разделители между ссылками меню в шапке?

То есть нужно начинать с глубокого анализа макета, то есть, размеры отступов между элементами, какой контейнер лучше использовать (Grid или Flex) и тд. С чего собственно я и начала. Благо макет представлен в Figma, соответственно путем комбинаций клавиш Cntrl + Alt можно узнать размеры отступов любых элементов, а если еще использовать инструменты разработчика, то вообще практически весь CSS код будет доступен. После так называемого анализа макета, можно потихоньку начинать писать код. Забыл добавить, что все файлы проекта мы клонировали с удаленного репозитория Яндекс Практикума, и все файлы картинок и ссылки шрифтов у нас уже есть, более того, нам даже задали стандартные размеры для body и подключили все шрифты.

Начинаем писать HTML код. Любая HTML страница начинается с

<!DOCTYPE html><html lang="ru"> <head> <link rel="stylesheet" href="./fonts/fonts.css"/> <link rel="stylesheet" href="./styles/global.css"/> <link rel="stylesheet" href="./styles/style.css"/> <title>Оно тебе надо — аукцион вещей, в которые никто не верил</title> </head>

Думаю, что здесь все супер легко и понятно. Язык русский, в теге head подключаем все шрифты и файл со стилями (style.css). Далее Яндекс Практикум нам рекомендует называть класс так, как они хотят, ну и это естественно правильно (все классы должны указывать на код, который принадлежит им, и что вообще мы пишем), как минимум, в крупных IT-компаниях, нужно писать код, чтобы другие разработчики тоже его понимали (об этом как нибудь в другой статье)

Следующим шагом является тег body, в котором лежит основной код страницы, который видит пользователь. Сначала сверстаем тег header - семантический тег.

<header class="header">

<nav class="header\_menu">

<ul class="menu\_links">

<li class="link\_list main\_link\_marker">

<a class="link\_main" href="#0">главная</a> </li> <li class="link\_list"> <a href="#0">посмотреть лоты</a> </li> <li class="link\_list"> <a href="#0">об аукционе</a> </li> </ul> </nav> <a> <img class="header\_logo" src="./images/logo-black.svg" alt="image" /> </a> <address> <ul class="address"> <li> <!--удалил класс и сделал общий для стилизации --> <a href="tel.+99995555555">+9 999 555 5555</a> </li> <li> <a href="mailto:info@sobaka.ge">info@sobaka.ge</a> </li> <li> <!-- удалил класс --> наб. Принсенграхт 263-<br /> 265, Амстердам </li> </ul> </address> </header>

На комментарии в коде не обращайте внимание, я исправлял ошибки, которые указал ревьюер. Теперь пробежимся по коду. Для навигационного меню мы используем тег nav, это не является каким то строгим правилом, но лучше делать через него, так более

правильно.

Дальше в nav'e у нас лежит условный список в html - ul, а уже в нем у нас лежат элементы: "главная", "посмотреть лоты", "об аукционе" (через тег li). И обратите внимание что все сами слова находятся в ссылке (a - тег для ссылок), поскольку у нас нет точно адреса, куда эти ссылки ведут, то и в href мы пишет 0 или #0, кому как нравится.

Что по стилям? Все просто и понятно. Тег header мы определяем как грид контейнер, для более удобного расположения элементов, чтобы не нужно было на глаз подгонять отступы как в макете (все равно можно использовать плагин Pixel Perfect). Отступы padding определяем как в макете figma. Само меню мы называем классом header\_menu - именно так Яндекс Практикум рекомендует называть классы, и в этом классе мы пишем flex-контейнер ( конечно же можно и grid, но мне больше хотелось flex), и с помощью свойства justify-self позиционируем элементы по центру. Класс menu\_links непосредственно о наших (главная, посмотреть лоты, об аукционе) элементах, естественно формируем контейнер, с помощью figma ставим отступы между словами (свойство gap) и конечно же свойство list-style-type для того, чтобы между словами появились

кружочки

. С классом link\_list все предельно просто, по figma пишем какие стили будут иметь сами слова. А теперь важный момент, в макете перед словом

главная

нету кружочка, соответственно нам нужно написать дополнительный класс(см html код) для этого слова со свойством list-style-type: none, а у внутреннего класса в ссылке просто пишем свойство, чтобы текст не подчеркивался.

.header {

display: grid;

grid-template-columns: 1fr min-content 1fr;

align-items: center;

padding-top: 92px;

padding-left: 24px;

padding-bottom: 92px;

padding-right: 24px;

}

.header\_menu {

display: flex;

justify-self: center;}.menu\_links { display: flex; gap: 32px; list-style-type: circle; padding-right: 7px;}.link\_list { font-size: 18px; font-weight: 400; text-underline-offset: 4px;}.main\_link\_marker { list-style-type: none;}.link\_main { text-decoration: none;}

Переходим к следующему подблоку - address.

<address>

<ul class="address">

<li> <!--удалил класс и сделал общий для стилизации -->

<a href="tel.+99995555555">+9 999 555 5555</a>

</li>

<li>

<a href="mailto:info@sobaka.ge">info@sobaka.ge</a>

</li>

<li> <!-- удалил класс -->

наб. Принсенграхт 263-<br />

265, Амстердам

</li>

</ul>

</address>

Здесь интересная достаточно ситуация, Яндекс Практикум нам рекомендует сделать адрес ссылкой и в href написать то, что вы видите в html коде, честно, я не знаю зачем нужно так писать и почему нельзя обойтись просто 0 или #0, но видимо это какие то общепринятые правила *хорошего тона*. В принципе не сильно вижу смысла что то здесь объяснять, единственное, про тег br, полезный тег для переноса текста или слова на следующую строку.

Теперь перейдем к стилизации. Тут все просто и нету никаких замудренностей, грубо говоря переписываем весь код из figma и получаем хорошо сверстаную шапку страницы.

.address { padding-left: 185px; font-size: 18px; line-height: 24.3px; letter-spacing: 0.18px; font-family: "Raleway", sans-serif; list-style-type: none; font-style: normal; }

Ну что? Переходим к более интересному моменту верстки, картинки, лоты, и слова на каждой новой строке на картинке. Но начнем по порядку.

Второй семантический тег - main, опять же, вы можете использовать обычный div, очередное общепринятое правило

хорошег*о тона*

. И в этом теге у нас будут лежать тег section, то есть секция с большой картиной,

Ознакомимся с кодом для

перво**й секции**

.

<section class="cover">

<div class="overlay"></div>

<h1 class="cover\_title"> <!-- изменен заголовок -->

<span class="text\_cover"> о<span class="spacing">н</span><span class="no\_spacing\_ok">o</span> </span> <span class="text\_cover text\_center"> те<span class="spacing">бе</span> </span> <span class="text\_cover text\_right"> <span class="spacing">н</span><span class="no\_spacing\_ok">а</span><span class="half\_spacing">д</span>о </span> </h1> <div class="cover\_description"> <p class="cover\_text">Аукцион вещей, в которые никто не верил</p> <button class="button">СДЕЛАТЬ СТАВКУ</button> </div> </section>

Как видим, здесь мы будем использовать тег span - тег для стилизации строчного контейнера. Как видим, текст расположен на трех разных строках, поэтому span сюда идеально подходит. Обращаю ваше внимание, что каждый слог слова обернут в отдельный span для удобной стилизации. Далее нам нужно задать оверлей, это такой стиль для затемнения картинки по макету. Ну и в конце кнопка, как ни странно тег button, ничего сложного нет.

Теперь перейдем к стилям данной секции.

.cover {

position: relative;

background-image: url(../images/cover.jpg);

background-size: cover;

color: white;

text-transform: uppercase;

z-index: 0;

background-repeat: no-repeat;

background-position: center;}.overlay { top: 0; left: 0; right: 0; bottom: 0; position: absolute; background-color: rgba(0, 0, 0, 0.6); width: 100%; height: 100%; z-index: -1;}.cover\_title { display: flex; flex-direction: column; padding: 165px 50px 147px; gap: 20px; text-align: center;}.cover\_title .span { font-size: 100px; font-weight: 700; line-height: 100%; letter-spacing: 30px;}.text\_cover { align-self: flex-start; line-height: 115%; transform: translateY(2px); text-align: center; word-spacing: 5px; }.text\_center { align-self: center; text-align: center; }.text\_right { align-self: flex-end; text-align: right; } .text\_cover .spacing { letter-spacing: 60px; } .text\_cover .no\_spacing\_ok { letter-spacing: 0px; /\* сброс межбуквенного разрыва \*/ }.text\_cover .half\_spacing { letter-spacing: 60px;}.text\_cover .no\_spacing { letter-spacing: 60px;}.cover\_description { display: flex; align-items: baseline; justify-content: space-evenly; padding: 0px 50px 55px 78px; gap: 79px;}.cover\_text { font-size: 22px; font-weight: 400; line-height: 28px; word-spacing: 18px;}.button { width: 256px; height: 51px; border: 1px solid white; background-color: transparent; font-family: "Raleway", sans-serif; color: #fff; font-size: 22px; line-height: 51px; font-weight: 500; text-align: center;}

Разберемся с каждым селектором по порядку.

.cover

, position: relative; задаем для относительного позиционирования элементов, background-repeat: no-repeat; задаем для того, чтобы картинка не повторялась, background-position: center; чтобы картинка расположилась по макету, z-index: 0 устанавливаем чтобы все остальные элементы находились сверху с z-index: >=1 , а сам оверлей будет с отрицательным значением 1.Далее рассмотрим оверлей, в нем мы обнуляем координаты, чтобы на глаз не задавать margin отступы, а высоту и ширину задаем на 100% чтобы уложилось ровно в картинку, цвет задаем полупрозрачный черный. В

.cover\_title

задаем классический flex-контейнер, flex-direction: column; для расположения дочерних элементов вертикально в padding задаем отступы строго по макету,

gap: 20px

задаем для отступа между элементами в flex-контейнере. Ну и отдельным правилом задаем для

.cover\_title

и

span

задаем классические стили текста по макету. В

.text\_cover

из интересного единственное что могу отметить, это свойство align-self: flex-start, данное свойство идеально справляется с выравниванием элемента вдоль поперечной оси. Остальные свойства по макету или подгоняем под макет. Отдельно в классах для span задаем отступ буквенного разрыва по макету (свойством letter-spacing). Про

.cover\_description

максимально кратенько, определяем flex-контейнер, через свойство align-items задаем чтобы элементы выравнивались по базовой линии в контейнере, ну и дальше стандартные отступы, которые представлены в figma. Далее для того же селектора задаем обычный css код из figma (font-weight, font-size, line-height, word-spacing). Теперь перейдем к кнопке. Высоту и ширину задаем ровно как в макете, свойство border отвечает за толщину рамки, все остальное тоже строго по макету. Не забываем сделать ровным текст по центру в самой кнопке.

Секция с карточками лотов

.

<section class="lots">

<h2 class="title\_lots">Лоты</h2> <!-- изменен заголовок-->

<div class="container\_lots">

<div class="card\_lots">

<article class="card card\_about\_cinema">

<div class="overlay"></div>

<h3 class="card\_title">

Фильм режиссёра, который бросил киношколу

</h3>

<p class="text\_on\_card"> <!-- добавлен параграф-->

Не просто бросил, а ушёл с первой лекции. Какой была бы ваша

ставка, если бы вы не знали, что режиссёр — Пол Томас Андерсон?

</p>

</article>

</div>

<div class="card\_lots">

<article class="card card\_about\_book">

<div class="overlay"></div>

<h3 class="card\_title">

Книга, где описан один скучный день из жизни рекламного агента

</h3>

<p class="text\_on\_card"> <!-- добавлен параграф-->

Объёмом почти в тысячу страниц. Иногда без знаков препинания и с

переходами на древнеанглийский. В ней одновременно

рассказывается о 16 июня 1904 года и об истории литературы,

начиная с античных времён. И это всё накладывается на эпическую

поэму Гомера. Сколько бы вы поставили на «Улисса» Джеймса

Джойса?

</p>

</article>

</div>

<div class="card\_lots">

<article class="card card\_about\_picture">

<div class="overlay"></div>

<h3 class="card\_title">

Картина, которую повторит даже 5-летний сын маминой подруги

</h3>

<p class="text\_on\_card"> <!-- добавлен параграф-->

Ну действительно, там линия, тут кружочек, а здесь и вовсе

что-то похожее на инфузорию-туфельку. Никаких полей, лесов, котиков... Сколько бы вы на такое поставили, если бы не знали, что это — одна из революционных работ Кандинского? </p> </article> </div> </div> <a href="#0" class="view\_more">посмотреть больше</a> </section>

Давайте разбираться. Для начала title, не очень понимаю почему, но ревьюер мне сказал, что его нужно обернуть в тег h2 - второй заголовок, но что с div, что с h2, разницы никакой

<h2 class = 'title\_lots'>Лоты</h2>

. Далее через div создаем контейнер с лотами, следующий div с классом card\_lots и дальше 3 карточки с текстом и заголовком. Поскольку три одинаковые карточки (различаются только текстом), то более

релев*антным*

решением будет определить единый класс заголовков карточек и единый класс для текстов карточек. И не забываем обернуть

пос*мотреть больше*

в ссылку (тег а) и задать отдельный класс.

Давайте пройдемся по styles карточек.

.title\_lots {

padding-top: 179px;

padding-bottom: 26px; padding-left: 64px; color: #000; font-size: 40px; font-weight: 700; text-transform: uppercase;}.container\_lots { display: flex; flex-wrap: wrap; list-style-type: none; gap: 25px; justify-content: flex-start; /\* указал start \*/ padding: 0; margin: 0 auto; margin-left: 20px; /\* левый отступ чтобы совпадало с макетом \*/ }.card\_lots { max-width: 334px; min-height: 563px; flex-grow: 1;}.card { position: relative; height: 100%; display: flex; flex-direction: column; z-index: 1; padding: 51px 40px 48px; justify-content: space-between; background-size: cover; background-repeat: no-repeat; background-position: center;}.card\_about\_cinema { background-image: url(./../images/card-lot-01.jpg);}.card\_about\_book { background-image: url(./../images/card-lot-02.jpg);}.card\_about\_picture { background-image: url(./../images/card-lot-03.jpg);}.card\_title { color: white; font-size: 22px; font-weight: 700; line-height: 120%; text-decoration-line: underline; text-transform: uppercase;}.text\_on\_card { color: white; font-size: 20px; font-style: normal; font-weight: 400; line-height: 120%;}.view\_more { display: block; text-underline-offset: 3px; padding-top: 30px; color: black; font-size: 20px; font-weight: 400; line-height: 23.48px; margin-left: 64px;}

В

.title\_lots

задаем код ровно такой же, как и в figma, и проставляем padding отступы как в макете. В контейнере для лотов определяем flex-контейнер, обнуляем padding и margin для выравнивания и задаем свойство justify-content со значением flex-start для выравнивания по главной оси flex-контейнера. В селекторе card\_lots задаем размеры для наших карточек (опять же как в макете), flex-grow: 1 для пропорциональности. Далее идет селектор card (родительский) в котором мы задаем относительное позиционирование, flex-контейнер, z-index:1 для того, чтобы заголовок и текст лежали сверху карточек, justify-content: space-between для выравнивания карточек по центру как в макете, и далее стандартный стили для картинок, которыми мы уже пользовались ранее (при стилизации первой секции). Далее следует 3 селектора, в которые мы просто помещаем путь наших картинок.

Обращаю ваше внимание

, что называть классы через 1, 2, 3, лучше не стоит, потому что в один момент, в разработке может поменяться нумерация классов и все придется переделывать. Такой фидбэк мне дал ревьюер при проверке моего проекта, поэтому ссылаясь на классический код-стайл, называем классы так, чтобы они хоть как нибудь объяснили свое содержимое. В

.card\_title

мы задаем те же свойства, что и в фигме (свойство text-transform необязательное, оно для заглавных букв). В

.text\_on\_card

аналогично, такие же стили и свойства, как и в макете, ничего своего придумывать не надо. Для класса

посмотреть больше

выравниваем отступы как в макете, делаем такой же цвет, и свойством

text-underline-offset: 3px;

подгоняем дефолтное подчеркивание ссылки под макет.

Мы уже близки к завершению, осталось совсем немного. Секция

about

.

<section class="about">

<div class="about\_container">

<div class="about\_logo">

<img

src="./images/logo-white.svg"

alt="image"

class="about\_logo\_image"

/>

</div>

<div class="about\_content">

<h2 class="title\_about">ОБ АУКЦИОНЕ</h2>

<div class="about\_text">

<p class="text\_up">

<!-- добавлен параграф-->

Здесь вы не встретите очередное пафосное cобрание невероятно<br />

дорогого антиквариата. Наши лоты вообще не должны были

попастьни<br />

на один аукцион. Потому что кому нужен дневник

девочки-подростка<br />

или картина, которую может нарисовать даже ребёнок? Кому

нужны<br />

все эти странные вещи, созданные любителями?

</p>

<p class="text\_down">

<!-- добавлен параграф-->

Слишком сложные или, наоборот, слишком простые. Опережающие

<br />своё время. В пух и прах растерзанные критиками.

Непринятые и<br />

непонятые. Когда-то они казались просто неудачными Но,

несмотря<br />

на критику, кажущуюся простоту или сложность, сейчас без этих<br />

лотов невозможно представить современную культуру. Когда в

эти<br /> вещи не верил никто, продолжали верить их создатели. И сейчас <br /> эти лоты стали культовыми. </p> </div> </div> </div> </section>

Достаточно легкая секция для верстки, давайте же разберем подробнее. Задаем отдельным дивом изображение, и отдельным дивом для текста. Вы наверное спросите, почему не задать картинку в стилях отдельным селектором? Я вам отвечу, тут опять дело вкуса, поскольку картинка всего лишь одна, ее можно задать в html коде (для большего количества картинок, лучше задавать в стилях, чтобы было меньше строк кода). Внутри дива про контент, определяем див для заголовка и самого текста, ну и снова видим старый добрый тег <br>. Здесь не буду на чем то заострять внимание, по структуре кода здесь все супер легко.

Перейдем к стилям данной секции.

.about {

margin-top: 148px;

}

.about\_container {

display: grid; grid-template-columns: 334px repeat(1, 1fr); gap: 25px; padding: 34px 24px 0px;}.about\_logo { justify-self: center; background-color: #000; border-radius: 50%; width: 240px; height: 240px; display: flex; justify-content: center; align-items: center;}.about\_logo\_image { width: 148px; height: 114px;}.about\_content { display: flex; flex-direction: column; gap: 24px; padding-top: 100px;}.title\_about { color: black; font-size: 40px; font-weight: 700; line-height: 46.96px;}.about\_text { align-self: flex-start; font-family: "Raleway", sans-serif; color: black; font-size: 20px; font-weight: 400; line-height: 24px;}.text\_up { padding-top: 76px;} .text\_down { padding-top: 25px;}

В селекторе

.about

задаем отступ, чтобы сама секция отодвинулась от предыдущей секции, как в макете. В контейнере задаем grid-контейнер с очень интересным свойством -

grid-template-columns: 334px repeat(1, 1fr);

.Данное свойство нужно для определения количества и размеров колонок в grid-layout. 1fr - это размер каждой повторяющиеся колонки, 1 - количество повторений, а 334px - фиксированная ширина для первой колонки. Как обычно свойство gap для расстояния между элементами как в макете, ну и в принципе на этом все. В селекторе картинки задаем flex-контейнер, и свойство border-radius - для того, чтобы сделать картинку круглой, как в макете. Стандартные ширина и высота для картинки, ну и обычные свойства для выравнивания по центру (характерные для контейнеров: align-items, justify-content, justify-self). В родительском теге про контент, снова задаем flex-контейнер и для него задаем знакомое вами свойство flex-direction со значением column для определения направления flex-контейнера. В стилях для заголовка ничего интересного нету, поэтому останавливаться здесь не будем. Стили для текста в очередной раз определяем стандартными, которые указаны в фигме, и через отступы выравниваем по макету.

Переходим к последнему блоку данного проекта!

Подвал

сайта или же

footer

.

<footer class="footer">

<address>

<ul class="address footer\_address">

<li>

<a href="tel.+99995555555">+9 999 555 5555</a>

</li>

<li>

<a href="mailto:info@sobaka.ge">info@sobaka.ge</a>

</li>

<li class="header\_\_address">

наб. Принсенграхт 263-<br />

265, Амстердам

</li>

</ul>

</address>

<nav class="footer\_menu">

<ul class="footer\_menu\_links">

<li class="footer\_links">

<a href="#" class="footer\_links footer\_main"> главная </a>

</li>

<li class="footer\_links">

<a href="#" class="footer\_links"> посмотреть лоты </a>

</li> <li class="footer\_links"> <a href="#" class="footer\_links"> об аукционе </a> </li> </ul> </nav> <div class="footer\_images"> <div> <a href="#0"> <img src="./images/yt.svg" alt="image" class="images\_footer" /> </a> </div> <div> <a href="#0"> <img src="./images/vk.svg" alt="image" class="images\_footer" /> </a> </div> <div> <a href="#0"> <img src="./images/pinterest.svg" alt="image" class="images\_footer" /> </a> </div> </div> </footer>

В так называемом подвале сайта мы наблюдаем уже знакомые нам пункты меню, как в первом блоке, и новые картинки в ряд. Максимально по простому можем скопировать код из шапки сайта и вставить его сюда. Аналогично со стилями, только обращаю ваше внимание, что

главная

,

посмотреть лоты

,

об аукционе

, расположены друг под другом и выравнены по центру (как в макете). Ниже представлен код со стилями.

.footer {

display: grid;

grid-template-columns: auto 1fr auto;

padding: 180px 50px 88px;

gap: 140px;

align-items: flex-start;

}

.footer\_address {

font-style: normal;

padding-left: 0px;

}

.footer\_menu { padding-left: 35px;}.footer\_menu\_links { list-style-type: none; text-underline-offset: 3px;}.footer\_links { color: black; text-align: center; font-size: 18px; font-weight: 400; line-height: normal;}.footer\_main { text-decoration: none;}.images\_footer { width: 48px; height: 48px;}.footer\_images { display: flex; gap: 48px;}

Здесь представлены практически такие же стили, как в шапке сайта. То есть grid-контейнеры, отступы, свойство

list-style-type: none;

и так далее. Обязательно не забываем обернуть три картинки в ссылку. Для них кстати оформлен flex-контейнер.

The End

Ну вот я и сделал разбор первого проекта на Яндекс Практикуме, он очень интересный и своеобразный. Впереди еще много крутейших проектов, про них я тоже сделаю разборы. Спасибо тем, кто прочитал эту статью, надеюсь вы прекрасно провели время и узнали что-то новое для себя. Весь проект я выложил у себя на Гитхабе -

https://github.com/Maksim-d[riller/ono-tebe-nado](https://github.com/Maksim-driller/ono-tebe-nado)

.