**Проектная работа «Сложно сосредоточиться»**

В этом проекте перед вами серьёзная задача: сверстать адаптивный сайт в светлой и тёмной темах. Работы много, но вы справитесь. Основные моменты мы подсветим в рекомендациях ниже.

В последнем уроке этой темы, «Сдача проектной работы», к вам в аккаунт GitHub автоматически скопируется стартовый репозиторий. Поэтому лучше всего двигаться в таком порядке: ознакомиться с описанием проекта в этом уроке, чек-листом — в следующем, получить заготовку — в последнем. Когда склонируете к себе на компьютер заготовку, вернитесь в этот урок и приступайте к выполнению проекта.

**Определитесь с подходами**

Перед тем как приступить к написанию кода, внимательно изучите [макет](https://www.figma.com/file/lCqDbWjgllgJtb2hmCqfyX/%236-%D0%A1%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE-%D1%81%D0%BE%D1%81%D1%80%D0%B5%D0%B4%D0%BE%D1%82%D0%BE%D1%87%D0%B8%D1%82%D1%8C%D1%81%D1%8F?type=design&node-id=0-1&t=APAUXUcSQ3Pijp4O-0). Вначале вам нужно принять решение, какие подходы использовать при вёрстке.

1. Mobile first или Desktop first. Выберите, с какого расширения начать. От этого будет зависеть, какие стили писать в первую очередь, без медиавыражений. Мы рекомендуем начать с мобильного, но финальное решение за вами!
2. Чёрное или белое? Решите, какую тему верстать как основную. Это не повлияет на то, какая тема будет включена у пользователя при входе на сайт. Нам кажется более логичным начинать с тёмной темы. Можете ориентироваться на тему в вашей операционной системе.

**Изучите стартовый код**

В этой работе вы не будете начинать с чистого листа. В заготовке проекта уже есть нужная структура со всеми файлами, шрифты, картинки и скрипт переключения темы.

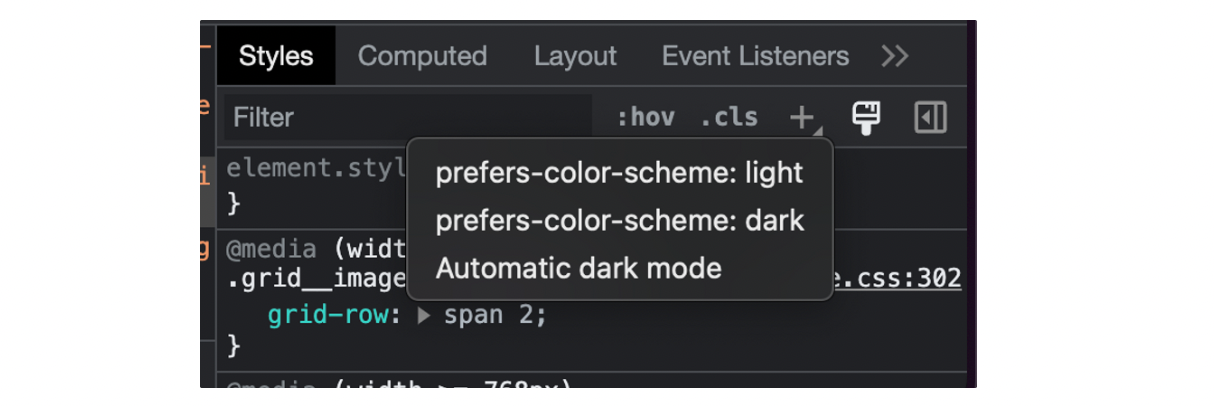
Вам нужно написать разметку, перенести контент из макета и всё это стилизовать при помощи CSS. Стили нужно писать в разных файлах, поэтому посмотрите на имена файлов и почитайте комментарии внутри. Так будет понятно, где какие стили писать.

В HTML уже есть блок с кнопками, переключающими тему принудительно, вне зависимости от выбранной на компьютере пользователя. Пожалуйста, не трогайте разметку этого блока, не меняйте классы. Иначе JavaScript-код сломается, и темы не будут переключаться.

В файле globals.css мы сбросили стандартные отступы элементов и написали несколько важных свойств, без которых никак не обойтись. Вы можете дополнять или менять эти стили по необходимости.

**Общие рекомендации**

* Стили для основной темы и основной ширины экрана пишите в файле style.css. Ниже в этом же файле вы напишете медиавыражения для адаптива под разные экраны.
* При вёрстке старайтесь использовать логические свойства там, где это возможно. Пишите, например, margin-block-start вместо margin-top или border-inline-end вместо border-right.
* Практически во всей работе для расстановки элементов удобно использовать гриды. С их помощью можно адаптировать сайт под разные экраны, практически не меняя стили. Советуем использовать их максимально.
* Верстайте так, как вам удобно. Возможно, вам удобнее работать блоками. Например, написать разметку для шапки, потом стили для тёмной и светлой тем, потом адаптировать под все экраны. И только после этого переходить дальше. А возможно, вы захотите работать над всей страницей целиком. Любой подход будет верным.
* Обязательно убедитесь, что сайт совпадает с макетом на основных разрешениях, а в промежуточных состояниях не ломается и блоки выглядят хорошо. Вспомните про «резиновую» вёрстку.
* Отдельно обратите внимание на горизонтальный скролл. Он не должен появляться ни на каких разрешениях и ни при каких условиях.
* Всегда держите инструменты разработчика открытыми. Как минимум там удобно переключать автоматические темы.



* Помните, что вёрстка должна совпадать с макетом. Автотесты будут проверять этот момент. Поэтому периодически накладывайте макет на страницу и проверяйте, что ничего не съехало.
* Запланируйте себе время на рефакторинг. Отложите код после окончания работы хотя бы на день. Затем вернитесь и посмотрите, можно ли где-то сделать проще и изящнее. Может быть, вы увидите, что использовали лишние обёртки или стили можно записать короче. Это хорошая привычка на будущее и воспитание в себе критического взгляда на собственную работу.

**Переменные**

Большинство цветов, шрифт, размеры текста для разных элементов и отступы удобно хранить в переменных, потому что они часто используются в разных местах кода. А ещё это нужно, чтобы быстро менять темы простым переопределением переменных. Изучите макет и запишите переменные в файл variables.css.

Задавая значения для переменных цветов, запишите только нужные для той темы, которую вы выбрали основной. Цвета для второй темы вы допишете в соответствующий файл позже.

Выбирайте для переменных имена, которые будут иметь смысл в обеих темах. Старайтесь без особой надобности не плодить сущности. Не нужно создавать для одного цвета две переменные для светлой и тёмной темы. Создайте одну переменную, а потом, если нужно, поменяйте её значение.

Не надо так:

Скопировать кодCSS

:root {

--accent-color-dark-theme: #ff0070;

--accent-color-light-theme: #ff8dcb;

}

Лучше так:

Скопировать кодCSS

/\* Файл variables.css \*/

:root {

--accent-color: #ff0070;

}

/\* Файл light.css \*/

:root {

--accent-color: #ff8dcb;

}

Не бойтесь пропустить какую-то переменную на этом этапе. В ходе работы этот список можно изменять и дополнять.

**Подключение файлов, метатеги**

Подключите все нужные файлы стилей в HTML-разметке. Внимательно выбирайте порядок подключения. Помните про каскадную сущность CSS: чем ниже в коде, тем приоритетнее. Поэтому важно, чтобы более общие стили были подключены выше менее общих.

Например, при подключении файлов для цветовых тем советуем такой порядок:

Скопировать кодHTML

<link rel="stylesheet" href="./styles/variables.css">

<link rel="stylesheet" href="./styles/style.css">

<link rel="stylesheet" href="./styles/dark.css">

<link rel="stylesheet" href="./styles/light.css">

Для остальных файлов стилей выберите порядок самостоятельно.

Чтобы браузер понял, что у нашего сайта есть тёмная и светлая темы внутри <head>, очень важно прописать строку с метатегом для color-scheme. На первом месте укажите ту тему, которую верстаете как основную. Если это тёмная тема, то запись будет такой:

Скопировать кодHTML

<meta name="color-scheme" content="dark light">

После этих действий браузер научится отображать страницу в той теме, которая выбрана в операционной системе. Например, на скринкасте видно, что страница не белая, как обычно, а с тёмным фоном.

После этого подключите фавиконку для нашей страницы, чтобы вкладка выглядела аккуратно.

**Шапка**

Для расположения элементов в шапке сайта отлично подходят гриды и свойства индивидуального выравнивания грид-элементов.

Если присмотреться к макетам в разных темах, то можно заметить, что в правом верхнем углу в тёмной теме есть небольшая надпись REC. Это декоративный элемент, который скрывается в светлой теме. Пожалуй, это единственный пустой тег в нашей разметке и единственный элемент, к которому уместно применить абсолютное позиционирование.

Точку слева от надписи реализуйте через псевдоэлемент — одного пустого тега в HTML достаточно.

Не забудьте про скринридеры. Поскольку это декоративный элемент, его нужно будет скрыть при помощи aria-атрибута от читалок.

В правом верхнем и нижнем левом углах есть декоративны уголки. Их можно реализовать при помощи псевдоэлементов и указания границ для двух сторон.

Чтобы заголовок в шапке красиво адаптировался и при этом хорошо выглядел на больших и маленьких экранах, используйте clamp(). Ниже вы найдёте формулы расчёта размера текста. Посмотрите на неё внимательно и подумайте, почему используются именно такие значения и для какого разрешения применить формулу:

Скопировать кодCSS

/\* Для мобильных \*/

clamp(7.25rem, 7.0115rem + 1.0178vw, 7.5rem)

/\* Для планшета и десктопа \*/

clamp(7.5rem, 0.5625rem + 14.4531vw, 9.8125rem)

Формулы сгенерированы с помощью [https://clamp.font-size.app/](https://clamp.font-size.app/?config=eyJyb290IjoiMTYiLCJtaW5XaWR0aCI6IjMyMHB4IiwibWF4V2lkdGgiOiIxNDU2cHgiLCJtaW5Gb250U2l6ZSI6IjE0cHgiLCJtYXhGb250U2l6ZSI6IjIwcHgifQ%3D%3D). Этот сайт вам пригодится, чтобы адаптировать текст на других разрешениях.

У текста в заголовке задана тень. Вот стили для неё: text-shadow: 4px 4px 0 var(--accent-color);.

Параграфу в шапке задан фоновый цвет. Такой же декоративный приём встречается в основном блоке сайта. Используйте отдельный класс title-decor, чтобы удобно задавать фон и цвет текста всем схожим элементам.

**Кнопки переключения тем**

Разметку этого блока мы заготовили заранее. Всё потому что по нажатию кнопок отрабатывает JavaScript-код, который принудительно меняет темы сайта. Как это работает?

**Кнопка «День»**

При нажатии на кнопку для <html> добавляется класс theme-light. При этом должна отобразиться светлая тема.

**Кнопка «Неон»**

При нажатии на кнопку для <html> добавляется класс theme-dark. При этом должна отобразиться тёмная тема.

**Кнопка «Авто»**

При нажатии на кнопку для <html> добавляется класс theme-auto. При этом должна отобразиться та тема, которая выбрана у пользователя в системе. Сработает медиафича prefers-color-scheme.

Для нажатой кнопки устанавливается класс .header\_\_theme-menu-button\_active. Кнопка должна стать недоступна для клика. Поможет свойство pointer-events: none.

Нужно стилизовать состояния кнопок по ховеру и по фокусу. Для стилизации фокуса используйте :focus-visible. Стили всех состояний можно найти в макете.

Из-за того, что темы на нашем сайте будут переключаться не только в зависимости от настроек пользователя, но и принудительно, по классу, при написании стилей нам придётся дублировать стили тем. Один раз для класса, а второй раз для медиафичи prefers-color-scheme.

Предположим, вы выбрали тёмную тему как основную. Тогда структура кода внутри файлов стилей будет такой.

Файл variables.css:

Скопировать кодCSS

:root {

/\* Переменные тёмной темы \*/

}

Файл dark.css:

Скопировать кодCSS

:root.theme-dark {

/\* Дублируем переменные для тёмной темы \*/

}

/\* Остальные стили \*/

Файл light.css:

Скопировать кодCSS

:root.theme-light {

/\* Переменные светлой темы \*/

}

/\* Остальные стили \*/

@media (prefers-color-scheme: light) {

:root {

/\* Дублируем переменные светлой темы \*/

}

}

/\* Остальные стили \*/

**Основной контент**

Перед тем, как верстать основной контент страницы, задайте для .page фоновое изображение, расположите его по центру и пусть оно закрывает собой всего родителя. После этого пропишите background-attachment: fixed;, чтобы фон оставался на месте при прокрутке страницы. Чтобы менять фоновое изображение в зависимости от темы, вам нужно будет использовать селекторы .theme-dark .page и theme-light .page.

В основном контенте встречается обычный текст и заголовки разных уровней. Разбейте весь текст на секции, определите, какого уровня нужны заголовки. Просмотрите макеты для всех разрешений и примерно прикиньте, какие обёртки вам понадобятся для удобного адаптива этих блоков. Переиспользуйте класс title-decor там, где текст находится на цветном фоне. Старайтесь задавать похожим элементам одинаковые классы, чтобы писать стили только один раз. Повторений нам и так хватает.

В тексте встречаются ссылки. Им тоже задана тень. Возьмите значения для свойства text-hadow из макета и реализуйте этот стиль по аналогии с заголовком в шапке. Не забудьте про ховер и фокус для ссылок.

**Галерея**

В галерее всем картинкам должен быть задан атрибут alt с описанием изображения. Поскольку галерея находится сильно ниже первого экрана, который увидит пользователь, мы можем сократить время загрузки страницы и задать для <img> атрибут, отвечающий за «ленивую» загрузку.

Сложная разметка здесь не понадобится, адаптировать галерею под разные устройства удобно при помощи свойств гридов.

**Футер**

Футер во многом повторяет шапку сайта. Тот же размер, декоративные элементы, стили заголовка.

Есть маленький подвох: стили заголовка отличаются от стилей в шапке, пусть и совсем немного. Будьте внимательны.

Помните, что хотя заголовки выглядят одинаково, они не могут быть одного уровня. На странице должен быть только один заголовок <h1>, и его место в шапке, а не в подвале.

Декоративные уголки можно переиспользовать из шапки. Для этого пропишите стили отдельному, независимому от положения в документе классу.