Geekbrains

**Разработка веб–сайта для мебельной компании**

IT–специалист:

Frontend–программист. Цифровые профессии

Лошакова Л. Ю.

Краснодар

2023

Оглавление

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Введение | …………………………………………………………………………….. | | | | | | | | | | | | | | | | 3 |
| Глава 1 Основы создания веб–сайта | | | | | | | | ………………………………………..…....... | | | | | | | | | 5 |
| 1.1 Что такое веб–сайт и для чего он нужен | | | | | | | | | | | | ……..…………………………. | | | | | 5 |
| 1.2 Основные инструменты для создания сайта | | | | | | | | | | | | | ……………………………. | | | | 9 |
| 1.3 Виды и методы добавления эффектов наведения и нажатия | | | | | | | | | | | | | | | | …………. | 10 |
| 1.4 Что такое семантические элементы | | | | | | | | | | | …………………………………....... | | | | | | 15 |
| Глава 2 Подготовка к написанию сайта | | | | | | | | | | ………………………………………....... | | | | | | | 21 |
| 2.1 Создание макета в Figma | | | | | | | …………………………………………........... | | | | | | | | | | 21 |
| 2.2 Разработка изменений сайта при нажатии и наведении | | | | | | | | | | | | | | ……………....... | | | 30 |
| Глава 3 Разработка веб–сайта | | | ………………………………………………........... | | | | | | | | | | | | | | 33 |
| 3.1 Создание шапки сайта | | | | | ………………………………………………....... | | | | | | | | | | | | 33 |
| 3.2 Создание контентной части | | | | | | | | | …………………………………………....... | | | | | | | | 38 |
| 3.3 Разработка подвала сайта | | | | | | | ………………………………………………... | | | | | | | | | | 41 |
| 3.4 Добавление адаптива | | | | ………………………………………..................... | | | | | | | | | | | | | 45 |
| 3.5 Проверка внешнего вида на разных разрешениях экрана | | | | | | | | | | | | | | | …………...... | | 48 |
| 3.6 Проверка на валидность | | | | | | | ……………………………………………....... | | | | | | | | | | 51 |
| 3.7 Проверка типографики | | | | | | …………………………………………………. | | | | | | | | | | | 52 |
| Заключение | | ………………………………………………………………………….. | | | | | | | | | | | | | | | 56 |
| Список используемой литературы | | | | | | | ……………………………………………....... | | | | | | | | | | 57 |

Введение

Первый в мире сайт info.cern.ch появился 6 августа 1991 года. Его создатель, Тим Бернерс–Ли, опубликовал на нём описание новой технологии World Wide Web, основанной на протоколе передачи данных HTTP, системе адресации URI и языке гипертекстовой разметки HTML. На сайте были описаны принципы установки и работы серверов и браузеров. Сайт стал и первым в мире интернет–каталогом, так как позже Тим Бернерс–Ли разместил на нём список ссылок на другие сайты.

Со временем технологии улучались и на данный момент у веб–разработчиков есть возможность непросто размещать текст на странице сайта и прикреплять к нему ссылки, появились такие возможности, как добавление стилизации, позиционирование, трансформирование, анимированные и так далее.

На сегодняшний день веб–сайты играют важную роль в жизни каждого человека. С помощью сайтов мы можем записаться к врачу, купить одежду, узнать о новых компаниях или найти полезную и интересную информацию.

Актуальность данной темы обусловлена тем, что в наше время каждая компании нуждается в узнаваемости, при этом веб–сайт является главным инструментом для продвижения бизнеса, так как на нем может быть размещена информация о компании, список товаров и услуг, форма подписки на рассылку с новыми предложениями, возможность покупки в онлайн режиме, не выходя из дома, а также возможность заказать звонок. Вся эта информация, размещённая в удобном для пользователя формате, позволяет клиенту, не выходя из дома ознакомиться с компанией и принять решении о покупке товаров или услуг.

В соответствии с выбранной темой, разберем разработку сайта для мебельной компании, так как данный бизнес покрывает не маловажную часть потребностей клиентов. При этом многие клиенты не знают, где можно заказать качественную мебель, подходящую под их размеры, вид деятельности и желаемый внешний вид.

Цель: Изучить особенности разработки веб–сайта на примере мебельной компании.

Задачи:

1. Изучить литературу, касающуюся темы исследования;
2. Рассмотреть основные виды и методов добавления эффектов наведения и нажатия;
3. Ознакомиться с семантическими элементами;
4. Создать в Figma дизайн веб–сайта;
5. Разработать сайт в соответствии с макетом;
6. Добавить адептов сайта и различного рода эффекты.

Инструменты: Figma, JS, HTML, CSS, SASS, Валидатор, Типограф.

Глава 1 Основы создания веб–сайта

1.1 Что такое веб–сайт и для чего он нужен

Веб–сайт – это интерактивная веб–страница или набор страниц, предоставляющих пользователю доступ к информации, изображениям, видео и другим мультимедийным элементам через интернет. Он может быть создан с различными целями, включая продажу товаров и услуг, предоставление информации о компании и ее продуктах, общение с пользователями, обучение, развлечение и т. д.

Дизайн и функционал сайта могут быть разнообразными, включая такие элементы, как интерактивность, анимация, аудио и видео контент, формы обратной связи, панели управления, базы данных, онлайн–чаты и другие функции. Сайты могут быть статическими, где контент фиксирован, или динамическими, где контент генерируется и изменяется в зависимости от действий пользователя.

Для создания сайтов используются разнообразные технологии, языки программирования и инструменты, такие как HTML, CSS, JavaScript, PHP, Python, Ruby, CMS (системы управления контентом), фреймворки и другие. Кроме того, важно уделять внимание аспектам безопасности, скорости загрузки страниц и оптимизации для поисковых систем.

Сегодня сайт является неотъемлемой частью современного бизнеса и представляет собой мощный инструмент для привлечения клиентов и продвижения бренда. Он может быть использован как малыми предприятиями, так и крупными компаниями, и становится все более доступным благодаря развитию технологий и услуг веб–разработки.

Веб–сайт состоит из множества элементов, каждый из которых выполняет уникальную функцию и направлен на улучшение пользовательского опыта. Давайте рассмотрим некоторые из основных элементов:

1. Шапка (Header) – это верхняя часть сайта, которая содержит логотип компании, основное меню, контактную информацию и другие важные элементы навигации. Она является визитной карточкой сайта и помогает пользователям быстро ориентироваться.
2. Навигация (Navigation) – это элемент, который облегчает поиск нужной информации на сайте. Он может быть представлен в виде меню, ссылок, выпадающих списков и других интерактивных элементов. Хорошая навигация помогает пользователям быстро и легко перемещаться по сайту, что повышает удобство пользования.
3. Содержание (Content) – это самая важная часть сайта, которая должна быть структурирована и информативна для посетителей. Хорошо организованное содержание помогает пользователям быстро находить нужную информацию и улучшает их взаимодействие с сайтом. Важно, чтобы контент был понятным, легкочитаемым и содержал полезную информацию для посетителей.
4. Боковая панель (Sidebar) – это дополнительный блок на сайте, расположенный по бокам страницы, который содержит различные виджеты и информацию. В боковой панели можно размещать рекламу, теги, популярные посты и многое другое.
5. Футер (Footer) – это нижняя часть сайта, которая содержит дополнительную информацию о сайте. В футере обычно размещаются ссылки на социальные сети, контактную информацию, копирайт и другие полезные элементы.
6. Кнопки (Buttons) – это элементы интерфейса, которые помогают пользователям взаимодействовать с сайтом. Кнопки могут выполнять различные функции, например, они могут быть использованы для совершения покупки, заказа звонка или подписки на рассылку.
7. Формы (Forms) – это элементы, которые позволяют пользователям отправлять информацию на сайт. Формы могут использоваться для обратной связи, регистрации пользователя, оформления заказа и других целей. Они позволяют пользователям удобно и безопасно передавать нужные данные.
8. Изображения и видео (Images and Video) – это важные элементы, которые помогают визуализировать информацию на сайте и делают его более привлекательным для посетителей. Они могут быть использованы для улучшения пользовательского опыта и привлечения внимания к ключевым сообщениям.
9. Социальные кнопки (Social Media Buttons) – это инструменты, которые помогают пользователям легко делиться контентом сайта в социальных сетях. Они увеличивают его видимость и способствуют распространению информации, что может привести к привлечению новых посетителей и повышению уровня вовлеченности.
10. Карта сайта (Sitemap) – это страница, которая содержит структуру сайта и помогает поисковым системам более эффективно индексировать его содержимое. Это существенно для улучшения видимости и доступности сайта, так как позволяет посетителям быстро найти нужную им информацию и лучше понять его структуру.

Каждый из этих элементов имеет свою уникальную функцию и может быть использован для улучшения пользовательского опыта на сайте. Эффективное сочетание всех этих элементов поможет сделать сайт более привлекательным и удобным для посетителей.

На сегодняшний день существует множество разных типов веб–сайтов, каждый из которых обладает своими уникальными особенностями и функционалом. Поэтому знакомство с различными типами сайтов может помочь вам сделать правильный выбор и найти подходящий тип сайта для решения вашей конкретной задачи.

Один из самых распространенных типов сайтов – это сайты–визитки. Они часто используются для представления компании, ее товаров или услуг. Обычно на таких сайтах содержится основная информация о компании, ее контактные данные и некоторые изображения, позволяющие представить визуальное представление о компании.

Еще один тип сайтов – это корпоративные сайты. Они предоставляют детальную информацию о компании, ее продуктах, услугах и контактных данных. Кроме того, на таких сайтах может быть раздел для новостей и пресс–релизов, а также документация для скачивания.

Однако это только два примера из множества возможных типов сайтов. В зависимости от ваших потребностей, вы можете также рассмотреть другие типы, такие как интернет–магазины, блоги, форумы и многое другое. Важно определить свои цели и требования, чтобы выбрать тип сайта, который наилучшим образом соответствует вашим потребностям.

Интернет–магазины предоставляют возможность пользователям приобретать товары и услуги онлайн. Они обычно оснащены удобными каталогами продуктов, функцией добавления товаров в корзину и системой оплаты, что делает процесс покупки максимально удобным и безопасным.

Блоги являются площадкой, где авторы публикуют свои мнения, статьи, рассказы и другой интересный контент. Они часто содержат категории, тэги, комментарии и форму для обратной связи, что позволяет читателям активно обсуждать и выражать свое мнение по поводу опубликованного материала.

Форумы представляют собой площадку для общения пользователей, где они могут создавать темы, вступать в дискуссии и обмениваться мнениями. Форумы обычно имеют разделы для различных тем и подразделы для более углубленного обсуждения конкретных вопросов. Это идеальное место для обмена опытом, получения советов и нахождения ответов на интересующие вопросы.

Социальные сети являются платформами, где пользователи могут создавать свои профили, общаться друг с другом, делиться фотографиями и видео, играть в игры и многое другое. Кроме того, эти площадки также предоставляют возможности для рекламы и продвижения бренда.

Сайты–каталоги, в свою очередь, являются местами, где собраны ссылки на различные площадки, разделенные по тематикам. Они могут быть как универсальными, так и специализированными.

Кроме вышеперечисленных типов платформ, существуют также порталы, агрегаторы новостей, онлайн–сервисы и другие. Каждый из них обладает своими уникальными особенностями и возможностями, поэтому важно выбрать наиболее подходящий вариант, отвечающий вашим потребностям.

1.2 Основные инструменты для создания сайта

Инструменты веб–разработки существенно облегчают процесс создания веб–сайтов и значительно расширяют их функциональные возможности. Эти инструменты можно разделить на несколько категорий:

* Компьютер и операционная система – необходимые инструменты, которые понятны и очевидны.
* Инструменты для работы с текстом – включают текстовые редакторы, которые используются для написания программного кода.
* Веб–браузеры – это программы, которые позволяют пользователям просматривать и отображать веб–страницы. Если разработчик хочет, чтобы его веб–сайт был доступен для пользователей разных браузеров, то необходимо протестировать его в разных веб–браузерах.
* Отладчик браузера – это инструмент, который может быть использован в качестве черновика при написании кода. Внося изменения в отладчик, можно быть уверенным, что веб–сайт не сломается. Если результат не устраивает, достаточно просто обновить страницу. Кроме того, отладчик помогает обнаружить ошибки в коде, возможные проблемы со стилями и так далее.

В процессе разработки веб–проектов часто используются различные инструменты и программы. Некоторые из них включают графические редакторы, которые используются для проектирования шаблонов, верстки и графического наполнения. Они помогают создавать красивый и профессиональный дизайн веб–страниц.

Другой важный инструмент – система контроля версий, которая позволяет хранить разные состояния проекта и облегчает совместную работу. В системе контроля отслеживаются все изменения, вносимые в проект, и таким образом программисты могут легко делиться кодом, поддерживать его и минимизировать конфликты при совместной работе.

FTP клиент – это программа для передачи файлов на сервер (например, веб–сервер), их загрузки и редактирования. Он очень полезен при разработке веб–сайтов, так как позволяет легко обмениваться файлами между локальной машиной и сервером.

Наконец, фреймворки – это удобные инструменты, предлагающие готовые решения для различных задач. Они содержат наборы инструментов, библиотек и стандартных практик, которые значительно упрощают и ускоряют разработку веб–проектов. Фреймворки предлагают структуру и организацию проекта, что позволяет разработчикам сосредоточиться на реализации функциональности, а не на создании всего с нуля.

1.3 Виды и методы добавления эффектов наведения и нажатия

Псевдоклассы предоставляют возможность добавить особые классы к элементам. Они применяются к объектам, которые либо отсутствуют в структуре веб–страницы, либо не могут быть выбраны с помощью обычных селекторов. Например, псевдокласс можно использовать для стилизации первой буквы или первой строки в абзаце.

Вы, наверняка, замечали на различных сайтах, что при наведении курсора мыши на пункт меню происходят различные изменения. Фон меняет свой цвет, ссылки становятся другого цвета, а также изменяются шрифт и его размер. Все это возможно благодаря применению псевдоклассов. Давайте рассмотрим их синтаксис (см. Рис. 1).

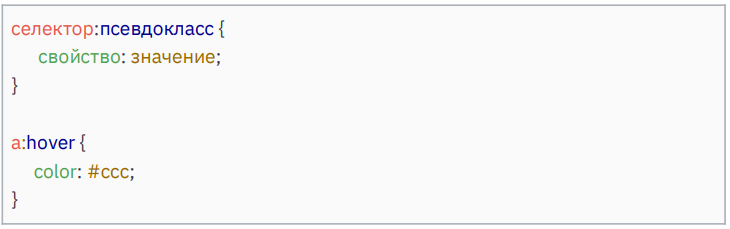


Рис. 1 Синтаксис псевдокласса

После селектора следует двоеточие, а затем указывается название псевдокласса без пробелов. Наиболее часто используемые псевдоклассы определяют состояние элемента и позволяют изменить его визуальное представление.

Ниже представлены несколько примеров популярных псевдоклассов:

1. a:link — применяется к ссылке, которая еще не была посещена пользователем.
2. a:visited — применяется к ссылке, которая уже была посещена пользователем.
3. a:hover — применяется к любому элементу, над которым находится курсор мыши.
4. a:focus — применяется к любому элементу, на котором установлен фокус с помощью мыши или клавиатуры.

Использование этих псевдоклассов позволяет создавать интерактивные и улучшенные пользовательские интерфейсы.

5. Псевдоэлемент "a:active" ссылается на активированный пользователем элемент. Псевдоэлементы позволяют ввести несуществующие элементы в веб–документ и придать им определенные стили. Они были введены в CSS1, но стали доступны только в CSS2.1. В начале использовалось одно двоеточие, но в CSS3 используется двойное двоеточие для отличия от псевдоклассов. Большинство современных браузеров поддерживают оба типа синтаксиса псевдоэлементов, за исключением Internet Explorer 8, который поддерживает только одно двоеточие. Поэтому рекомендуется использовать его для повышения совместимости. C помощью свойства "content" можно изменить внешний вид части элемента.

В этом разделе мы рассмотрим некоторые часто используемые псевдоэлементы в CSS:

1. :first–letter — этот псевдоэлемент позволяет выбирать первую букву каждого абзаца. Обратите внимание, что он применяется только к блочным элементам.
2. :first–line — этот псевдоэлемент выбирает первую строку текста внутри элемента. Он также применяется только к блочным элементам.
3. :before — этот псевдоэлемент вставляет генерируемое содержимое перед указанным элементом.
4. :after — этот псевдоэлемент добавляет генерируемое содержимое после указанного элемента.

Таким образом, эти псевдоэлементы могут быть полезными инструментами для стилизации и улучшения внешнего вида вашего веб–сайта.

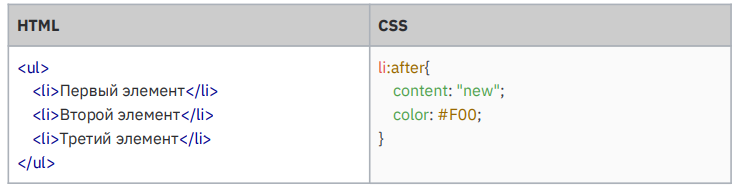


Рис. 2 Пример использования псевдоэлементов

После всех элементов списка li, изображенных на рисунке 2, появится текст "new" красного цвета. Если добавить плавные переходы для элементов, то для общего свойства transition будут использованы его составляющие, аналогично свойствам background, background–color, background–image и т.д. Давайте рассмотрим, из чего состоит общее свойство transition:

1. Свойство transition–property. Оно содержит название CSS–свойств, к которым будет применен эффект перехода. Это свойство не наследуется.
   1. Значение
      1. none – отсутствие свойства для перехода.
      2. all – значение по умолчанию. Это применяет эффект перехода ко всем свойствам элемента.
      3. свойство – определяет список css–свойств, перечисленных через запятую, которые участвуют в переходе.
2. Продолжительность перехода transition–duration задает промежуток времени, в течение которого должен осуществляться переход. Это свойство не наследуется.
   1. Значения
      1. Время перехода указывается в секундах или миллисекундах, например, 1s или 5ms, или 0.3s.
3. Функция перехода transition–timing–function. Это свойство задает временную функцию, которая определяет скорость перехода объекта от одного значения к другому. Если вы определяете более одного перехода для элемента, например, цвет фона элемента и его положение, вы
   1. Значения
      1. ease — это функция по умолчанию, которая обеспечивает плавный переход с медленным началом, быстрым ускорением и замедлением в конце. Она соответствует кривой cubic–bezier(0.25, 0.1, 0.25, 1).
      2. linear — переход происходит равномерно на протяжении всего времени, без колебаний в скорости. Он соответствует кривой cubic–bezier(0, 0, 1, 1).
      3. ease–in — переход начинается медленно и плавно ускоряется в конце. Он соответствует кривой cubic–bezier(0.42, 0, 1, 1).
      4. ease–out — переход начинается быстро и плавно замедляется в конце. Он соответствует кривой cubic–bezier(0, 0, 0.58, 1).
      5. ease–in–out — переход медленно начинается и медленно заканчивается. Этот стиль соответствует cubic–bezier(0.42,0,0.58,1).
      6. cubic–bezier(x1, y1, x2, y2) — позволяет вручную устанавливать значения от 0 до 1 для кривой ускорения.
4. Задержка перехода transition–delay. Это необязательное свойство позволяет изменять свойства с некоторой задержкой, а не моментально. Оно не наследуется.
   1. Время задержки перехода указывается в секундах или миллисекундах.
5. Краткая запись перехода transition. Все свойства, отвечающие за изменение внешнего вида элемента, можно объединить в одно свойство transition.

Для создания анимации в CSS3 используется свойство @keyframes. Это свойство представлено на рисунке 3 и является контейнером для различных свойств оформления.

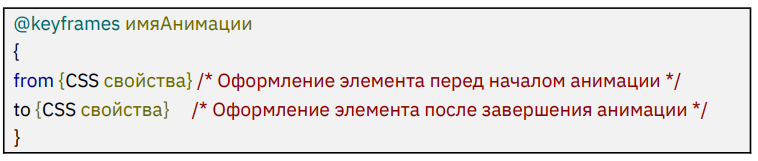


Рис.3 Пример записи свойства @keyframes

Для добавления анимации к элементу необходимо использовать CSS3 свойство "animation". В нем нужно указать имя анимации (1 значение) и время, в течение которого она будет выполняться (2 значение). Также можно задать количество повторов анимации (3 значение) (см. рис. 4).



Рис.4 Пример записи анимации

Длительность анимации animation–duration определяет, сколько времени занимает проигрывание анимации. Это свойство не наследуется от родительских элементов. Если не задано явно, по умолчанию устанавливается значение 0. (См. таблицу 1).

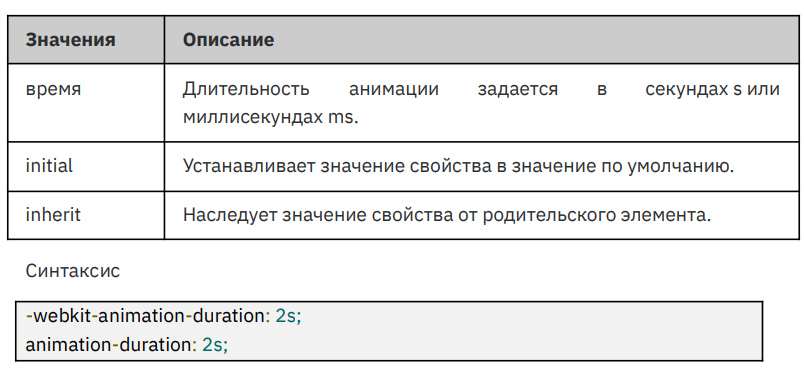


Табл.1 Значения свойств animation–duration

1.4 Что такое семантические элементы

Семантические HTML–элементы — это теги, которые служат для описания содержимого, находящегося внутри них. Они предоставляют информацию о структуре и значимости веб–страницы, что помогает поисковым системам лучше понимать ее содержание. Использование семантических HTML–элементов также сделает ваш код более понятным и легко поддающимся обслуживанию, так как каждый элемент будет иметь ясное назначение. В HTML5 появились новые семантические элементы, которые позволяют лучше структурировать веб–страницы и давать содержимому смысловую значимость. Отличительной особенностью всех семантических элементов является то, что они сами по себе ничего не делают. Например, элемент <video> используется для вставки полноценного видеоплеера на веб–страницу.

В данном случае возникает вопрос о том, стоит ли использовать новые элементы. Рассмотрим несколько причин, почему это целесообразно:

* + - 1. Более удобное редактирование и сопровождение. Используя новые элементы, можно с легкостью редактировать и обновлять веб–страницы, что облегчает работу разработчиков.

1. Оптимизация поисковых движков. Современные поисковые роботы уже проверяют наличие некоторых семантических элементов HTML5, чтобы получить больше информации о веб–страницах, которые они индексируют. Это может повысить видимость и рейтинг вашего сайта в поисковой выдаче.
2. Поддержка будущих возможностей. Новые браузеры и инструменты для редактирования веб–страниц используют всю мощь семантических элементов. Использование их сейчас гарантирует, что ваш сайт будет готов к будущим функциям и возможностям.

Внедрение новых элементов HTML5 не только улучшает функциональность и эффективность вашего сайта, но и подготавливает его для будущих инноваций в веб–разработке.

Рассмотрим назначение новых элементов более подробно:

Семантические элементы для работы со структурой страницы. Один из таких элементов – <header>. Он группирует вводные и навигационные элементы, хотя не является обязательным. В <header> можно разместить заголовки, заключить содержание раздела страницы, создать форму поиска или вставить логотипы. В html–документе может быть несколько элементов <header>, и они могут находиться в любой части страницы.



Рис. 5 Синтаксис семантического элемента <header>

Элемент <nav> предназначен для создания навигационных блоков на веб–странице или веб–сайте и может быть размещен как внутри <header>, так и в других частях страницы. Важно отметить, что на одной странице может быть несколько элементов <nav>. Этот элемент не заменяет теги <ul> или <ol>, а просто обрамляет их. Также не каждая группа ссылок на странице должна быть обернута в элемент <nav>. Он предназначен в первую очередь для разделов, содержащих основные блоки навигации.



Рис. 6 Синтаксис семантического элемента <nav>

В панели навигации можно использовать разнообразные элементы, а не ограничиваться только списками (см. Рис. 7):



Рис. 7 Пример использования элемента <nav>

Элемент <footer> представляет нижний колонтитул страницы или статьи. Обычно он содержит информацию об авторе, копирайте и дополнительные данные. Если <footer> используется для всей страницы, в нем также может присутствовать информация о авторских правах, условиях использования, контактной информации и ссылках на связанное содержимое. В одном веб–документе может быть несколько элементов <footer>. Интересно, что как каждая страница, так и каждая статья может иметь свой собственный <footer>, и его также можно поместить внутрь элемента <blockquote>, чтобы указать источник цитирования.



Рис. 8 Синтаксис семантического элемента <footer>

Элемент <article> используется для группировки различных записей, таких как публикации, статьи, записи блога или комментарии. Он представляет собой самостоятельный блок, который может быть использован несколько раз и обычно начинается с заголовка. Элемент <article> может также содержать другие вложенные элементы <article>, которые имеют схожее содержание с внешней статьей. Если на странице присутствует только одна статья, состоящая из заголовка и текстового содержания, то нет необходимости оборачивать ее в элемент <article>.

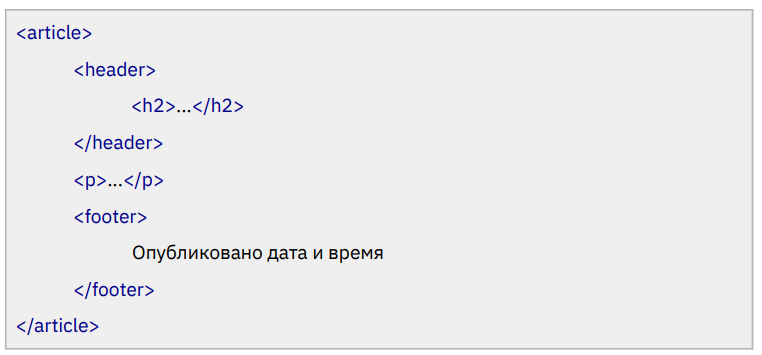


Рис. 9 Синтаксис семантического элемента <article>

Элемент <section> представляет собой многофункциональный компонент документа, который группирует тематически связанное содержимое и обычно содержит заголовок. В отличие от элемента <div>, <section> не используется для обертывания блоков. Он может содержать различные типы информации, такие как оглавления, разделы научных публикаций и программы мероприятий. На домашней странице сайта также можно использовать <section> для разделения вводной информации, новостей и контактов.



Рис. 10 Синтаксис семантического элемента <section>

Элемент <aside> предназначен для группировки содержимого, которое связано с основным контентом, но может быть рассмотрено как отдельный блок. Удаление этого элемента не должно повлиять на понимание основного содержания страницы. Обычно <aside> используется в качестве боковой колонки, например, в книгах, и может содержать такие элементы, как навигация, цифровые данные, цитаты, рекламные блоки или архивы. Однако, не следует использовать <aside> для простого позиционирования блоков в стороне.

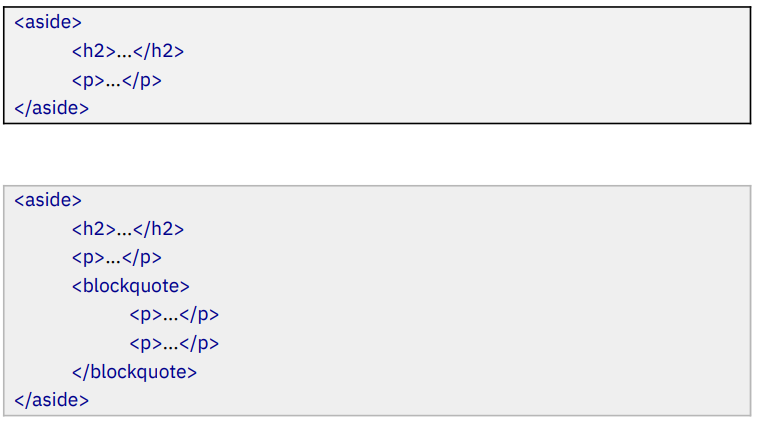


Рис. 11 Синтаксис семантического элемента <aside>

На рисунке 12 показан способ применения элемента <aside>:

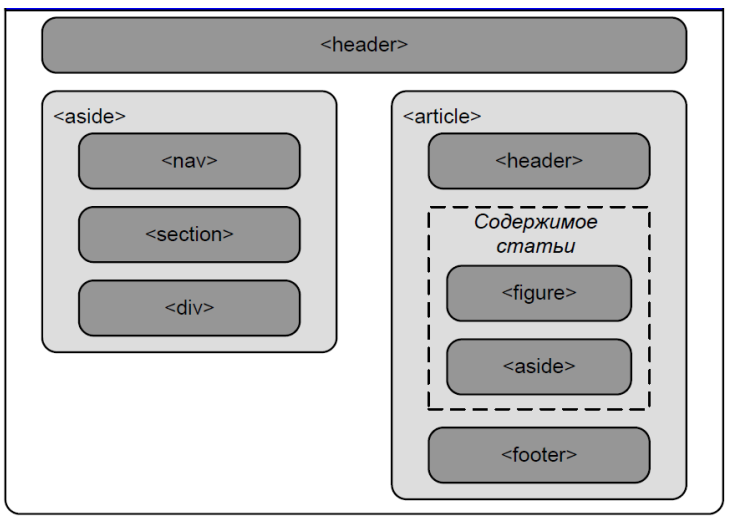


Рис. 12 Возможная структура элемента <aside>

Глава 2 Подготовка к написанию сайта

2.1 Создание макета в Figma

Figma (Фигма) – это удобный графический онлайн–редактор, предназначенный для коллективной работы. С помощью этого инструмента вы можете легко создавать прототипы веб–сайтов, интерфейсы приложений и обсуждать свои идеи с коллегами в режиме реального времени. В данной статье мы рассмотрим основные функции и возможности Фигмы, а также популярные плагины, которые помогут вам максимально эффективно использовать этот инструмент. Кроме того, мы расскажем, где можно бесплатно научиться работать с Фигмой.

В Фигме предоставляется возможность эффективной работы для различных специалистов, включая дизайнеров, маркетологов, менеджеров и разработчиков.

Для создания нового проекта в Figma, вам потребуется войти в свой аккаунт и нажать на кнопку «Создать новый файл». В открывшемся окне вы сможете выбрать размер и ориентацию холста, а также импортировать готовый макет в программу.

После выбора параметров создания нового проекта, вы будете автоматически перенаправлены в редактор Figma, где сможете начать работу над своим проектом. Здесь вы можете добавлять различные элементы, такие как текст, изображения, формы, создавать векторные объекты и многое другое.

Чтобы сохранить проект, регулярно нажимайте на кнопку «Сохранить изменения» в верхнем правом углу экрана. Также вы можете настроить автоматическое сохранение изменений через определенные промежутки времени.

По завершении работы над проектом, вы можете легко экспортировать его в нужном формате и поделиться им в Интернете или отправить клиенту для ознакомления. В программе Figma есть удобная функция экспорта, которую можно найти в меню "Экспорт".

В целом, создание нового проекта в Figma – это простой и интуитивно понятный процесс, который не требует особых навыков работы с графическими редакторами.

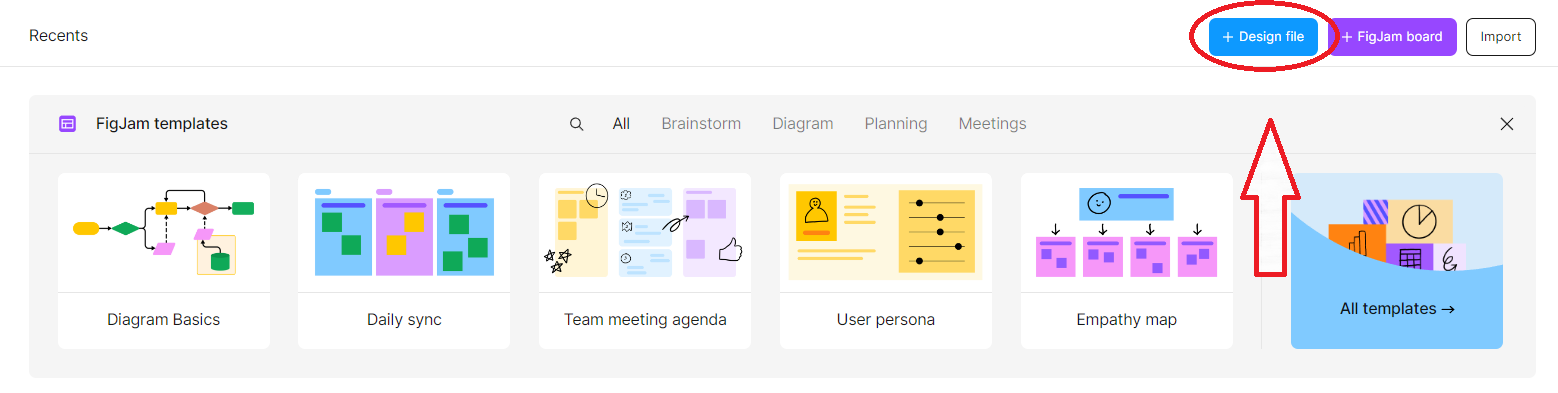


Рис. 13 Создание нового проекта

После нажатия на кнопку «+ Design file» (рис.13) открывается страница нового проекта (рис. 14), в котором мы и начнем проектировать наш сайт.

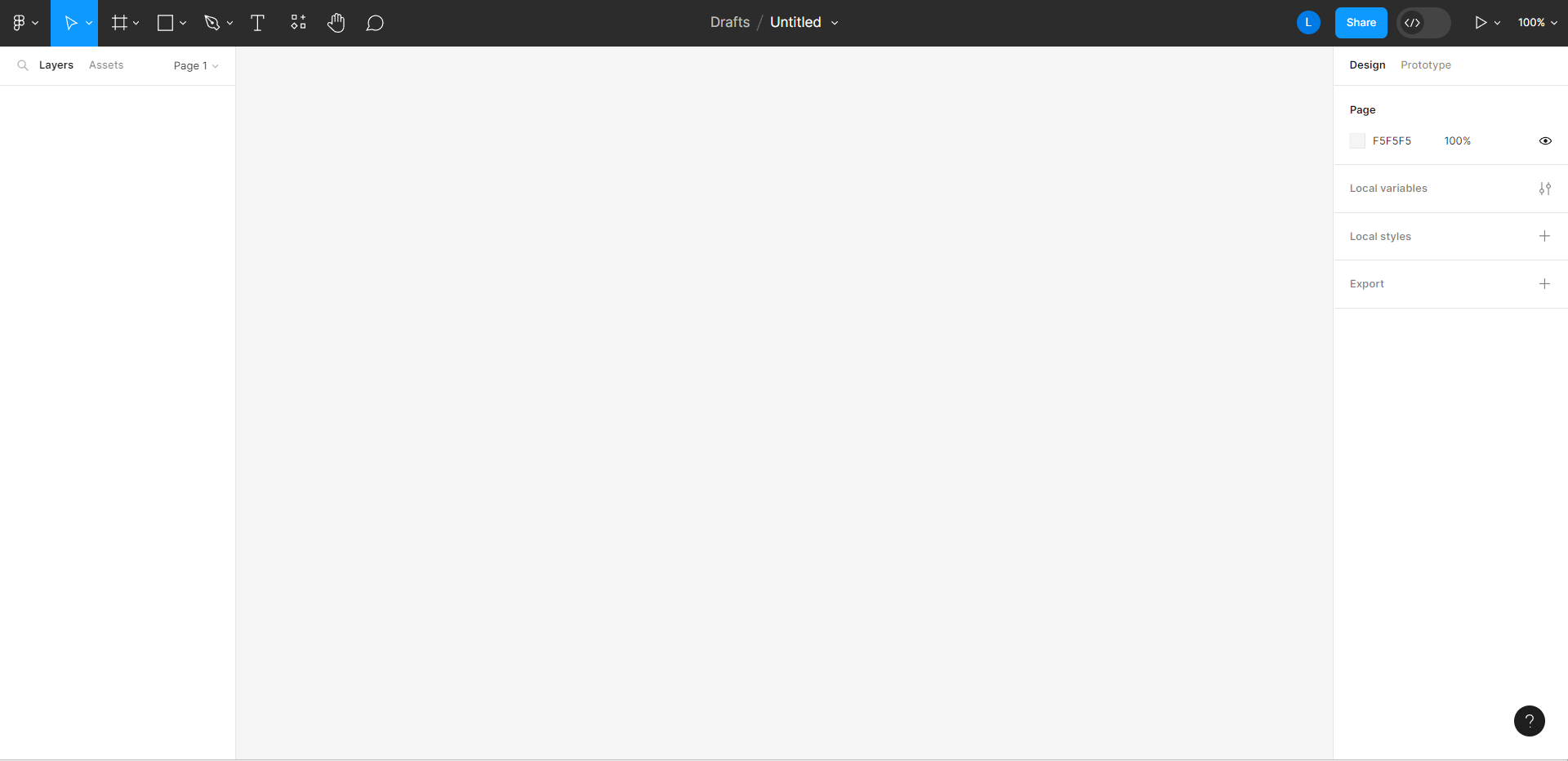


Рис. 14 Новая страница проекта

Теперь определимся с тем, какие страницы нам нужно спроектировать. Сайт для мебельной компании должен содержать в себе:

* главную страницу с логотипом и лозунгом компании;
* фотоальбом с уже выполненными заказами, подразделяемый на:
* кухни;
* шкафы и гардеробные;
* комоды и столы;
* страница с анкетой для заказа звонка;
* информацию об организации;
* контакты для связи и адреса офисов;

Название компании: PRODVITE

Начнем с главной страницы. Было придумано название компании и лозунг, где название: PRODVITE и лозунг: Ваш дом – Ваша индивидуальность. Цветовая гамма сайта будет состоять из нежных бежевых и тепло–коричневых оттенков, что будет визуально ассоциироваться у клиентов с теплотой, уютом и комфортом. Шрифт будет достаточно строгий, ровный без засечек, называется «Cormorant SC». Вдохновленная стилем с сайтом с коллекцией готовых css–анимаций «Animista» (рис. 16), я разработала страницу для своего сайта (рис. 17).



Рис. 15 Логотип мебельной компании

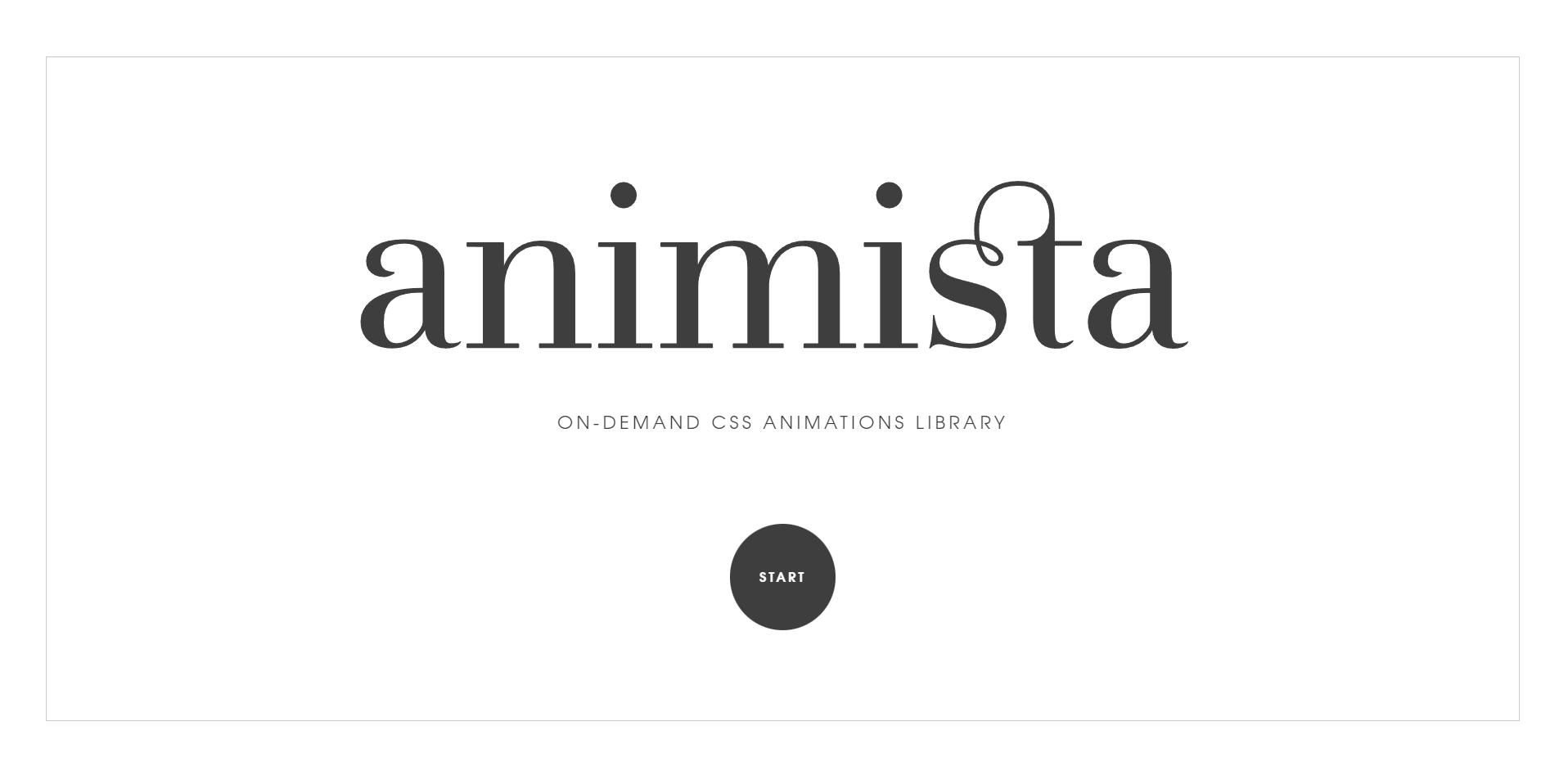


Рис. 16 Главная страница «Animista»



Рис. 17 Главная сайта мебельной компании

Теперь создаем навигационную страницу. В ней должно содержаться: Шапка профиля с логотипом, бюргер меню, текст, описывающий копанию, фото–ссылки на подразделы альбома и кнопка, переводящая на общую страницу альбома, а также подвал, с ссылками на социальные сети (рис. 18).

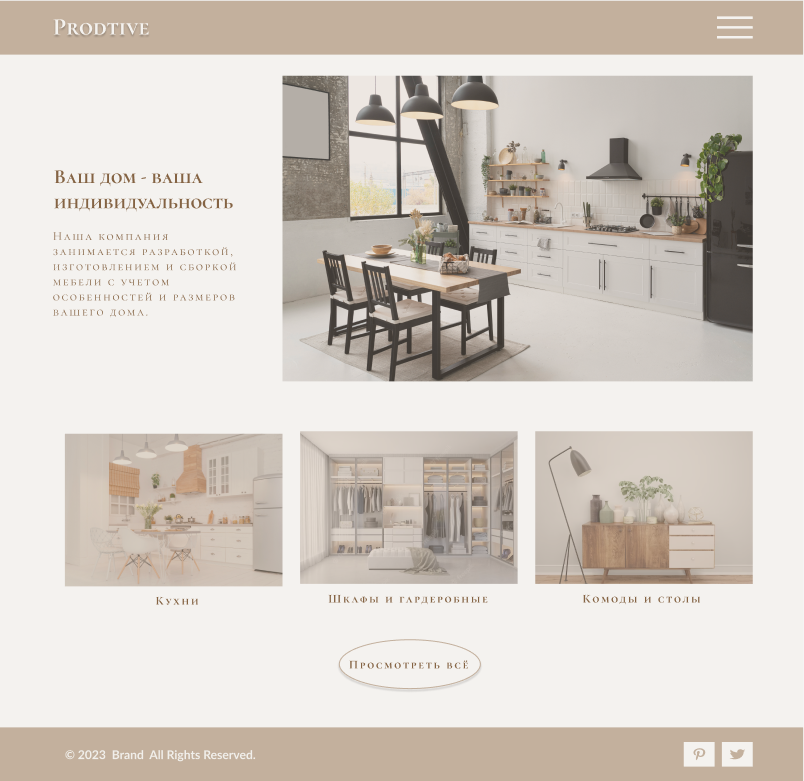


Рис. 18 Навигационная страница

Далее, создаем страницы фотоальбома. Поскольку компания абстрактная, фотоматериалы берем с сайта «Freepik». В принципе мы можем разработать макет только одной из страниц альбома, так как на этих страницах будут отличаться только фото и подраздел. Данная страница должна содержать:

* шапку сайта (логотип и бургер–меню, как на навигационной странице);
* раздел альбома;
* основная часть с портфолио организации;
* подвал (социальные сети компании, как на навигационной странице)

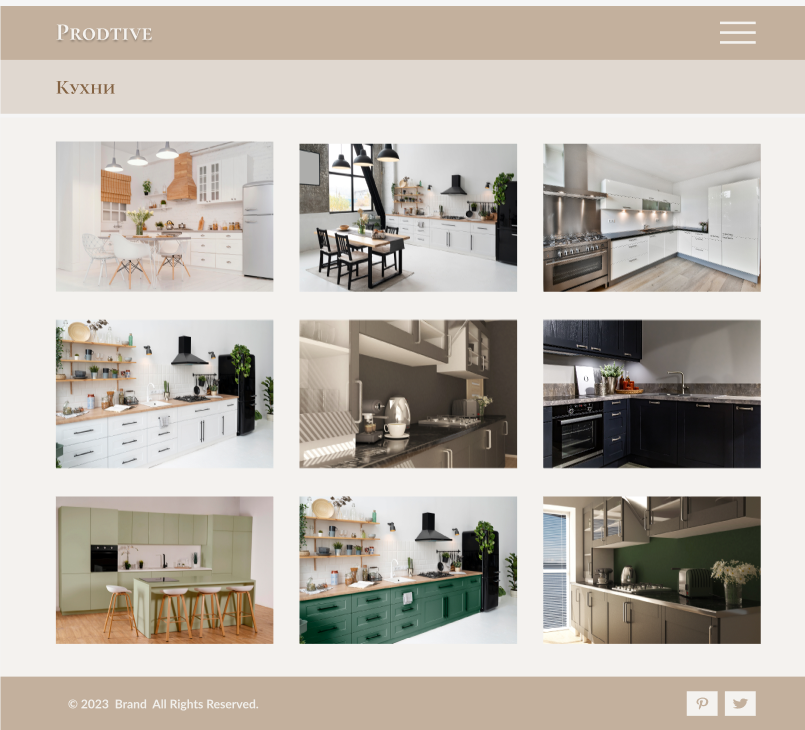


Рис. 19 Страница фотоальбома

Анкета для заказа звонка должна содержать в себе поля для ввода информации. Вводимой информацией будет ФИО и номер телефона. Подразумевается, что после заполнения и отправки анкеты, клиенту перезванивают на указанный номер и обращаются по имени из анкеты. Шапка и подвал остаются неизменными, отличие от страницы фотоальбома заключается в основной части и подразделе (рис. 20).



Рис. 20 Анкета

Страница с информации об организации повествует о философии производства мебели и порядке оформления заказа. Философия у организации следующая: У каждого человека – свои предпочтения, бюджет и потребности. Поэтому мы разрабатываем мебель, которая сочетает в себе эстетику, высокое качество и удобство использования, при этом даём гарантию и постгарантийный период! Самые опытных дизайнеры Краснодара и Московской области, а также мастера производства корпусной мебели Prodtive – обладают многолетним опытом, навыками и знаниями, чтобы создавать ту самую мебель, которая отразит Ваш индивидуальный стиль и соответствовать всем требованиям.

Работаем только с высококачественными материалами и используем проверенные годами технологии производства. Современное оборудование – позволяет нам создавать мебель любой сложности и воплощать самые смелые дизайнерские идеи. Каждый процесс выверен до миллиметра, а внимание уделяется – каждой детали, что обеспечивает высокую прочность и долговечность конечного продукта.

Порядок оформления и выполнения заказа:

1. Обработка заявки и формирование заказа
2. Создание и утверждение проекта
3. Подписание договора
4. Предоплата 70%
5. Срок производства: от 2х недель – до 45 рабочих дней
6. Бережная упаковка мебели
7. По готовности вносится доплата по договору
8. Сервисный центр отгружает вашу мебель
9. Последним этапом идёт сборка мебели у вас дома с подключением бытовой техники.

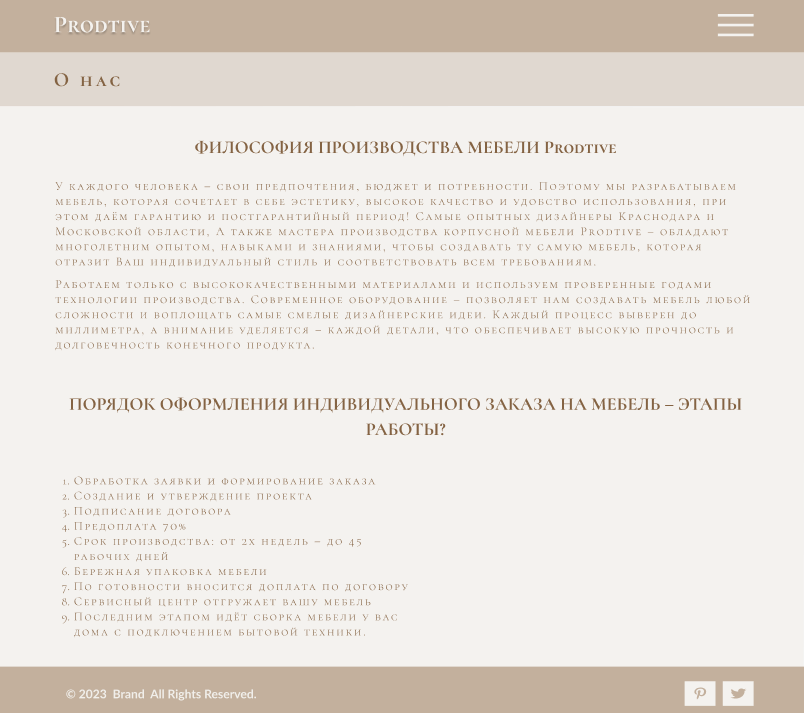


Рис. 21 Страница с информацией о компании

Страница с контактами обязательно включает в себя номер телефона для обращения по различным вопросам, юридический адрес. То есть адрес по которому компания зарегистрирована в ЕГРЮЛ, адрес отдела продаж (офиса), адрес цеха, для того чтобы клиент знал, где собирается его мебель. А также указывается электронная почта.



Рис. 22 Страница с контактной информацией и адресами

2.2 Разработка изменений сайта при нажатии и наведении

Реактивность объектов подразумевает их способность откликаться на наведение курсора или изменять свой вид в зависимости от действий пользователя. Несомненно, мы встречаемся с такими примерами повседневно: кнопки могут менять цвет или немного подрагивать при наведении курсора, ссылки могут изменять свой цвет, а картинки в галерее могут увеличиваться при клике.

CSS Animations – это модуль каскадных таблиц стилей, который позволяет создавать анимацию при наведении курсора на объекты. Этот функционал уже поддерживается всеми основными поисковыми системами. Несмотря на разногласия среди тех, кто предпочитает использовать Javascript для анимации, псевдокласс: hover широко используется в сообществе CSS для создания интерактивных эффектов.

И так, придумаем эффекты наведения на уже существующие в шаблоне страницы.

На рисунке 23 показано, какие эффекты наведения будут у главной страницы. Глядя на рисунок, мы видим, что с подобным эффектом у нас будет только одна кнопка, ведущая на навигационную страницу и эта кнопка при наведении будет становиться более темной. Так как главная страница вдохновлена сайтом «Animista», то и текст будет анимированным для этого, мы возьмем с сайта «Animista» анимацию под названием tracking–in–contract (animation: tracking–in–contract 4s cubic–bezier(0.215, 0.610, 0.355, 1.000) both;)

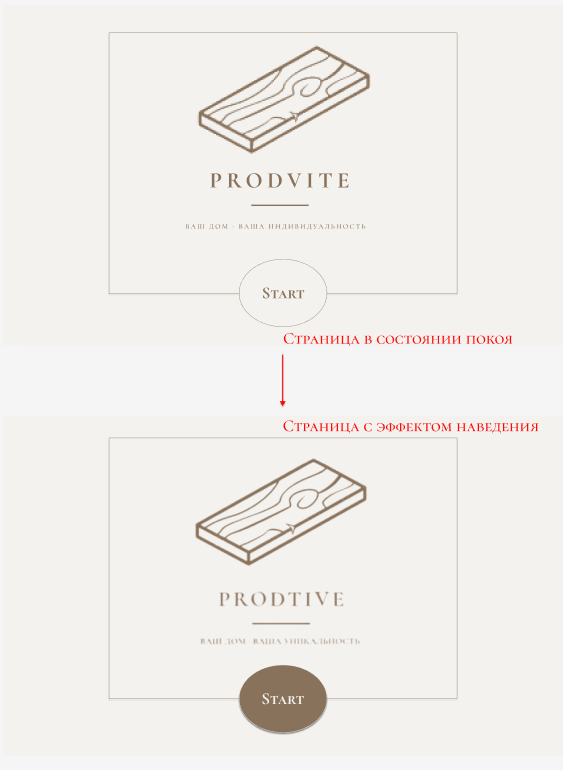


Рис. 23 Эффект наведения для главной страницы. До и после

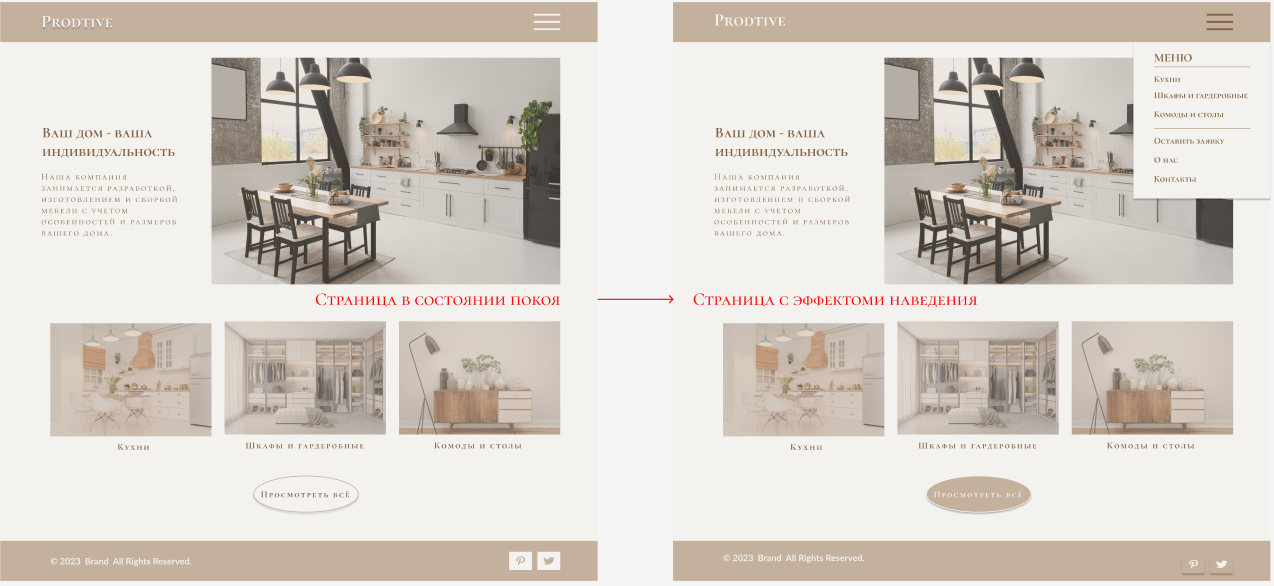


Рис. 24 Эффект наведения для навигационной страницы. До и после

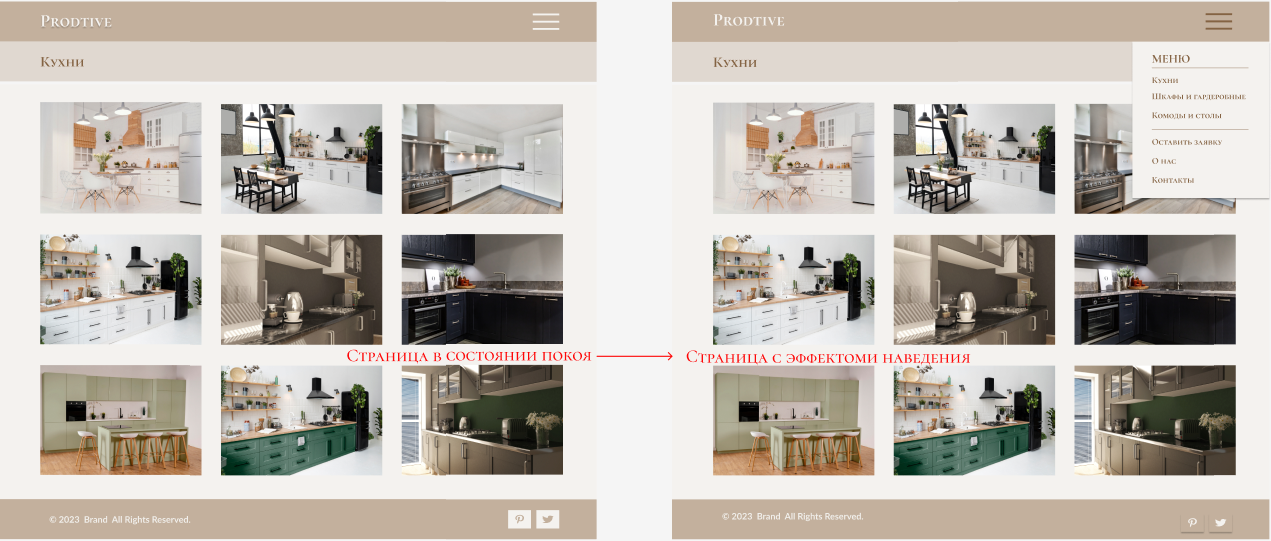


Рис. 25 Эффект наведения для страницы фотоальбома. До и после

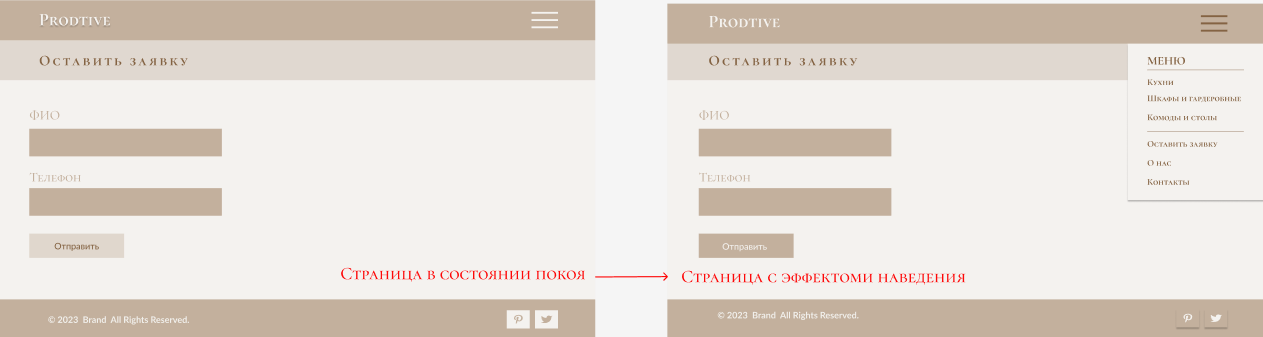


Рис. 26 Эффект наведения для страницы с анкетой. До и после

Глава 3 Разработка веб–сайта

3.1 Создание шапки сайта

Хедер, также известный как верхняя плашка или шапка сайта, является элементом, расположенным над остальными. Термин "header" также используется для обозначения "заголовка" или "колонтитула". В веб–дизайне и создании сайтов под хедером понимается область страницы, расположенная над контентной частью.

Обычно в шапке располагается логотип компании, ссылки на другие страницы, расположенные в бургер–меню, иногда добавляют строку поиска информации.

И так, начнем создавать шапку сайта. Для разбиения страницы на блоки мы будем использовать тег header, вместо div для расширения функционала нашего сайта.

Логотип компании обязательно должен ссылаться на главную страницу, в нашем случае на страницу index.html.

В записи это будет выглядеть следующим образом:

<header class="headeri">

<a href="index.html" class="headeri\_\_logo">Prodtive</a>

Теперь добавляем бургер–меню. Для этого нужно создать два отдельных блока div. В последнем блоке с id menu добавляем перечисление разделов сайтов со ссылкой на них, при помощи ненумерованного списка <ul> /<li>

<div id="burger–menu">

<span></span>

</div>

<div id="menu">

<ul>

<li class="menu\_\_bord menu\_\_center">Меню</li>

<li class="menu\_psition"><a href="kitchen.html">Кухни</a></li>

<li class="menu\_psition"><a href="wardrobe.html">Шкафы и гардеробные</a></li>

<li class="menu\_\_bord"><a href="dresser.html">Комоды и столы</a></li>

<li class="menu\_psition"><a href="form.html">Оставить заявку</a></li>

<li class="menu\_psition"><a href="about.html">О нас</a></li>

<li class="menu\_psition"><a href="contacts.html">Контакты</a></li>

</ul>

</div>

</header>

В JavaScript прописываем функционал меню.

<script>

var burgerMenu = document.getElementById('burger–menu');

var overlay = document.getElementById('menu');

burgerMenu.addEventListener('click', function ()

this.classList.toggle("close");

overlay.classList.toggle("overlay");

});

</script>

И обязательно прописываем стили в SCSS файле, который мы вместе со шрифтами предварительно связали с HTML страницей:

<head>

    <meta charset="UTF–8">

    <meta name="viewport" content="width=device–width, initial–scale=1.0">

    <title>Главная</title>

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link href="https://fonts.googleapis.com/css2?family=Cormorant+SC:wght@300;400;500;600;700&display=swap"

        rel="stylesheet">

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400&display=swap" rel="stylesheet">

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

</head>

Стили SCSS для шапки сайта:

.headeri {

    @extend %center;

    @extend %flex\_\_space\_between;

    background: $colorSite;

    padding–top: 23px;

    padding–bottom: 30px;

    &\_\_logo {

        color: $colorLightSelectSite;

        text–shadow: 0px 4px 4px rgba(0, 0, 0, 0.25);

        font–size: 48px;

        font–style: normal;

        font–weight: $weightFontBold;

        line–height: normal;

    }

    #burger–menu {

        cursor: pointer;

        height: 10px;

        width: 10px;

        margin–right: 50px;

        overflow: visible;

        position: relative;

        z–index: 2;

    }

    #burger–menu span,

    #burger–menu span:before,

    #burger–menu span:after {

        background: $colorLightSelectSite;

        display: block;

        height: 4px;

        opacity: 1;

        position: absolute;

        transition: 0.3s ease–in–out;

    }

    #burger–menu span:before,

    #burger–menu span:after {

        content: "";

    }

    #burger–menu span {

        right: 0px;

        top: 13px;

        width: 70px;

        display: block;

        transition: 0.7s;

    }

    #burger–menu span:hover {

        background–color: $colorSelectSite;

    }

    #burger–menu span:before {

        left: 0px;

        top: –20px;

        width: 70px;

        display: block;

        transition: 0.7s;

    }

    #burger–menu span:hover::before{

        background–color: $colorSelectSite;

    }

    #burger–menu span:after {

        left: 0px;

        top: 20px;

        width: 70px;

        display: block;

        transition: 0.7s;

    }

    #burger–menu span:hover::after{

        background–color: $colorSelectSite;

    }

    #burger–menu.close span {

        transform: rotate(–45deg);

        top: 13px;

        width: 70px;

        transition: 0.7s;

    }

    #burger–menu.close span:before {

        top: 0px;

        transform: rotate(90deg);

        width: 70px;

        transition: 0.7s;

    }

    #burger–menu.close span:after {

        top: 0px;

        left: 0;

        transform: rotate(90deg);

        opacity: 0;

        width: 0;

        transition: 0.7s;

    }

    #menu {

        filter: drop–shadow(0px 4px 4px rgba(0, 0, 0, 0.25));

        margin–left: 71%;

        z–index: 2;

        min–width: 370px;

        min–height: 420px;

        position: fixed;

        top: 112px;

        visibility: hidden;

        opacity: 0;

        text–align: left;

        padding–top: 20px;

        transition: all 0.3s ease–in–out;

    }

    #menu.overlay {

        visibility: visible;

        opacity: 1;

        padding–bottom: 30px;

        background: $colorLightSelectSite;

    }

    #menu li {

        padding–top: 17px;

        color: $colorSelectSite;

        font–size: 32px;

        font–weight: $weightFontBold;

        line–height: 50px;

        list–style: none;

    }

    #menu a {

        color: $colorText;

        font–size: 24px;

        font–weight: $weightFontBold;

        line–height: $lineHeight;

    }

    #menu a:hover{

        color: $colorSite;

    }

    .menu\_\_bord {

        margin–left: 55px;

        padding–bottom: 17px;

        padding–left: 0px;

        border–bottom: 1px solid $colorSelectSite;

        max–width: 260px;

    }

    .menu\_\_center {

        text–align: center;

    }

    .menu\_psition {

        padding–left: 55px;

    }

}

В итоге шапка на сайте выглядит следующим образом:



Рис 27. Итог написания шапки с закрытым меню

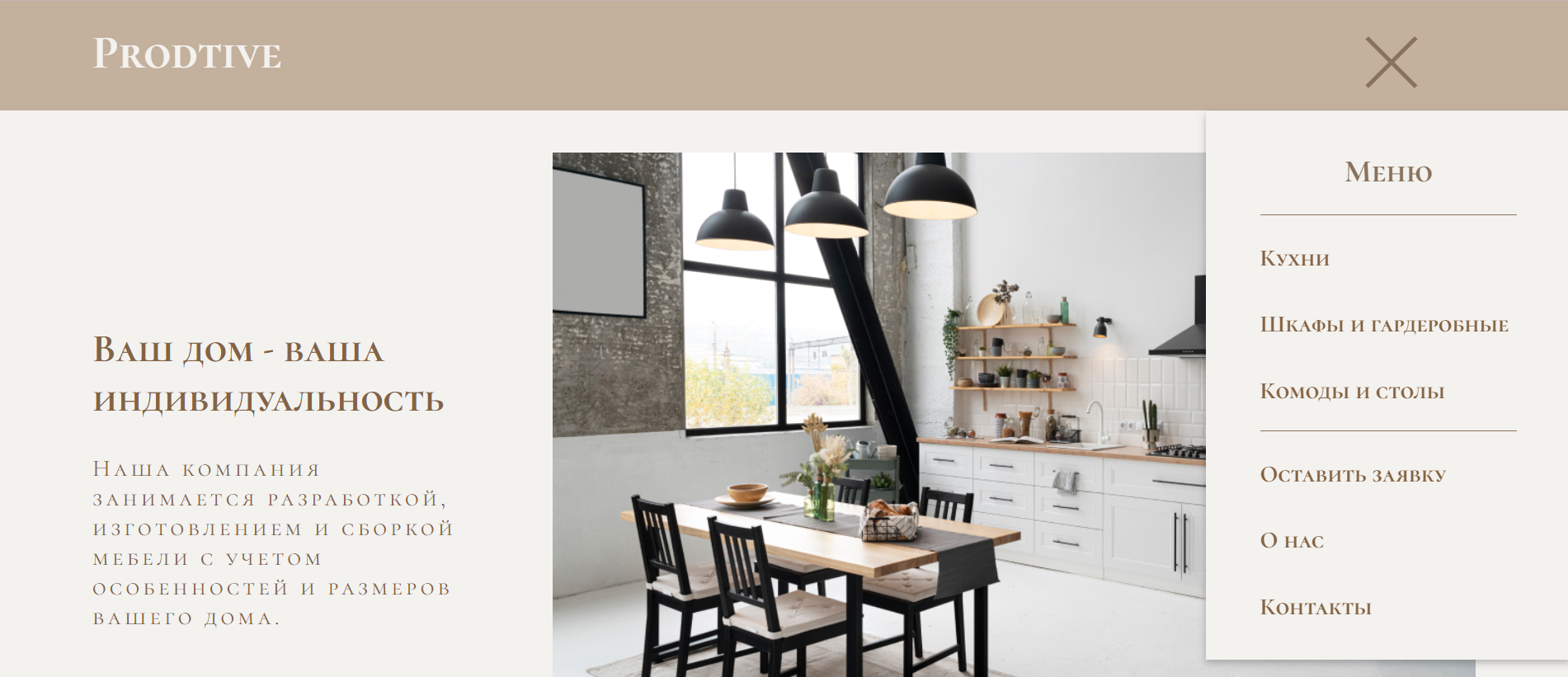


Рис. 28 Шапка с открытым меню

3.2 Создание контентной части

Контентная часть моего сайта состоит из тега section и div, которые делятся на часть со слоганом компании и на непосредственный контент главной страницы.

<section class="slog">

            <div class="slog\_\_content">

                <h1 class="slog\_\_title">Ваш дом – ваша индивидуальность </h1>

                <p class="slog\_\_text">Наша компания занимается разработкой, изготовлением и сборкой мебели с учетом

                    особенностей и размеров вашего дома.</p>

            </div>

            <div class="slog\_\_img">

                <img src="img/index/slog\_\_img.jpg" alt="photo">

            </div>

        </section>

        <div class="content">

            <div class="item content\_\_item1">

                <a href="kitchen.html" class="content\_\_link"><img src="img/index/link1.jpg" alt="photo"><span

                        class="content\_\_text">Кухни</span></a>

            </div>

            <div class="item content\_\_item2">

                <a href="wardrobe.html" class="content\_\_link"><img src="img/index/link2.jpg" alt="photo"><span

                        class="content\_\_text">Шкафы и гардеробные</span></a>

            </div>

            <div class="item content\_\_item3">

                <a href="dresser.html" class="content\_\_link"> <img src="img/index/link3.png" alt="photo"><span

                        class="content\_\_text">Комоды и столы </span></a>

            </div>

        </div>

Накладываем стили CSS:

.slog {

    @extend %center;

    display: flex;

    flex–direction: row;

    justify–content: space–between;

    &\_\_title {

        padding–top: 217px;

        padding–bottom: 30px;

        width: 398px;

        color: $colorText;

        font–size: $sizeTitle;

        font–style: normal;

        font–weight: $weightFontBold;

        line–height: 50px;

    }

    &\_\_text {

        width: 398px;

        color: $colorText;

        font–size: $sizeText;

        font–style: normal;

        font–weight: $weightFontRegular;

        line–height: 30px;

        letter–spacing: 3.6px;

    }

    &\_\_img {

        padding–top: 42px;

        padding–bottom: 55px;

    }

}

.content {

    @extend %center;

    display: grid;

    grid–template–columns: repeat(3, 432px);

    gap: 53px;

    &\_\_text {

        text–align: center;

        padding–bottom: 58px;

        color: $colorText;

        font–size: $sizeText;

        font–style: normal;

        font–weight: $weightFontBold;

        line–height: $lineHeight;

        letter–spacing: 3.6px;

    }

    &\_\_text {

        display: flex;

        flex–direction: column;

        gap: 13px;

    }

    .item {

        transition: 1s;

    }

    .item:hover {

        transform: scale(1.2);

    }

}

И получаем практически готовую страницу. Осталось сделать footer.

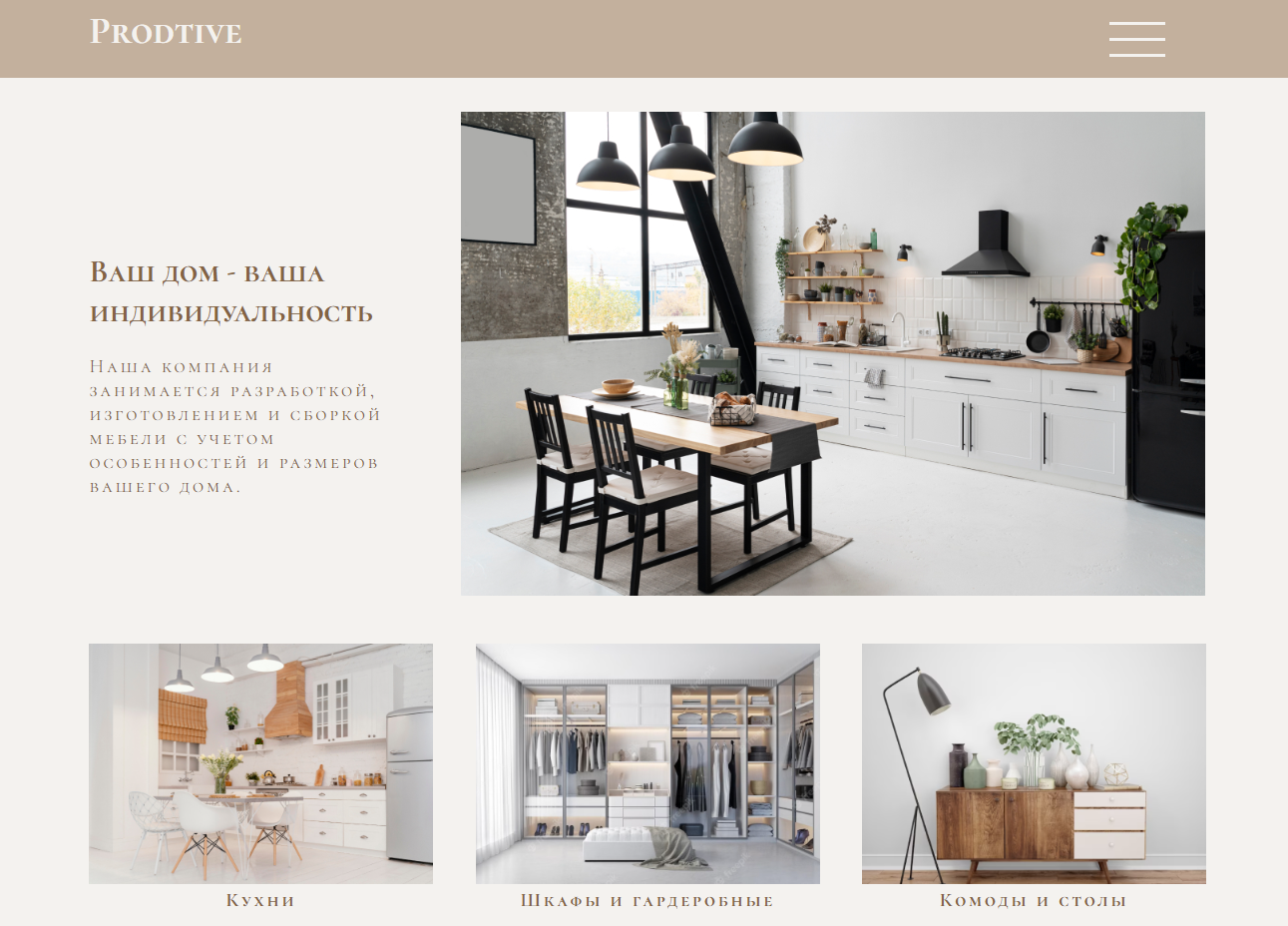


Рис 29. Контентная часть

3.3 Разработка подвала сайта

Футер сайта, также известный как подвал, является важной структурной частью, находящейся в нижней части веб–страницы. Он представляет собой противоположность хедеру или шапке сайта, которая располагается в верхней части.

Футер часто используется для размещения дополнительной информации и ссылок, а также может являться вспомогательным инструментом для сбора контактных данных.

В футере сайта обычно размещается информация, которую пользователи должны легко найти на любой странице. Поэтому часто в футере можно найти дублирующийся контент из хедера. Однако конкретный набор данных в футере зависит от типа сайта, его тематики, дизайна и содержания других блоков.

Типичные элементы контента, которые можно разместить в футере, включают контактную информацию, навигационные ссылки и правовую информацию.

Используя семантический тэг footer создаем блок для подвала, добавляем сведения об авторстве и ссылки на социальные сети. Я взяла одну социальную сеть для примера и две разные иконки.

<footer class="footer">

            <h3 class="footer\_\_text">© 2023 Brand All Rights Reserved.</h3>

            <div class="footer\_\_icon">

                <a href="https://ru.pinterest.com/" class="footer\_\_link"><svg width="62" height="49" viewBox="0 0 62 49"

                        fill="none" xmlns="http://www.w3.org/2000/svg">

                        <path class="link\_\_background" d="M61.3882 0H0V49H61.3882V0Z" fill="#F4F2EF" />

                        <g clip–path="url(#clip0\_2\_94)">

                            <path class="link\_\_color"

                                d="M31.7472 12.561C25.531 12.561 19.3876 15.8341 19.3876 21.1312C19.3876 24.5 21.7868 26.4141 23.2409 26.4141C23.8407 26.4141 24.186 25.0934 24.186 24.7201C24.186 24.2751 22.7501 23.3276 22.7501 21.4758C22.7501 17.6285 26.458 14.901 31.2564 14.901C35.3824 14.901 38.4359 16.7528 38.4359 20.1551C38.4359 22.696 37.1454 27.462 32.965 27.462C31.4564 27.462 30.1659 26.6007 30.1659 25.3661C30.1659 23.5573 31.7654 21.806 31.7654 19.9397C31.7654 16.772 26.0763 17.3462 26.0763 21.1743C26.0763 21.9782 26.2035 22.8683 26.6579 23.6004C25.8218 26.4428 24.1133 30.6776 24.1133 33.6061C24.1133 34.5105 24.2769 35.4006 24.3859 36.305C24.5919 36.4868 24.4889 36.4677 24.804 36.3768C27.8575 33.075 27.7485 32.429 29.1299 28.108C29.8751 29.2277 31.8017 29.8307 33.3285 29.8307C39.7628 29.8307 42.6527 24.878 42.6527 20.4135C42.6527 15.6618 37.4544 12.561 31.7472 12.561Z"

                                fill="#C3B09D" />

                        </g>

                        <defs>

                            <clipPath id="clip0\_2\_94">

                                <rect width="23.2652" height="24.5" fill="white" transform="translate(19.3876 12.25)" />

                            </clipPath>

                        </defs>

                    </svg>

                </a>

                <a href="https://ru.pinterest.com/" class="footer\_\_link"><svg width="62" height="49" viewBox="0 0 62 49"

                        fill="none" xmlns="http://www.w3.org/2000/svg">

                        <path class="link\_\_background" d="M62 0H0.611816V49H62V0Z" fill="#F4F2EF" />

                        <g clip–path="url(#clip0\_2\_90)">

                            <path class="link\_\_color"

                                d="M43.9537 19.5098C43.9734 19.7274 43.9734 19.9451 43.9734 20.1627C43.9734 26.8007 37.5765 34.4492 25.8848 34.4492C22.2829 34.4492 18.9368 33.6252 16.1221 32.1951C16.6339 32.2417 17.1259 32.2572 17.6574 32.2572C20.6295 32.2572 23.3654 31.4644 25.5502 30.112C22.7552 30.0653 20.413 28.6196 19.606 26.6297C19.9997 26.6763 20.3933 26.7074 20.8067 26.7074C21.3775 26.7074 21.9483 26.6452 22.4797 26.5364C19.5666 26.07 17.3818 24.0491 17.3818 21.6084V21.5463C18.2281 21.9194 19.2123 22.1526 20.2555 22.1836C18.543 21.282 17.4212 19.7429 17.4212 18.0018C17.4212 17.0691 17.736 16.2141 18.2872 15.4679C21.4168 18.5148 26.121 20.5047 31.396 20.7223C31.2976 20.3492 31.2385 19.9606 31.2385 19.572C31.2385 16.8048 34.0728 14.5507 37.5961 14.5507C39.4266 14.5507 41.0799 15.157 42.2412 16.1364C43.678 15.9187 45.0558 15.499 46.2762 14.9238C45.8038 16.0898 44.8 17.0691 43.4812 17.6909C44.7606 17.5822 46.0007 17.3023 47.1422 16.9137C46.2763 17.9086 45.1937 18.7946 43.9537 19.5098Z"

                                fill="#C3B09D" />

                        </g>

                        <defs>

                            <clipPath id="clip0\_2\_90">

                                <rect width="31.0202" height="24.5" fill="white" transform="translate(16.1221 12.25)" />

                            </clipPath>

                        </defs>

                    </svg>

                </a>

            </div>

        </footer>

Для того что, чтобы наш подвал смотрелся красиво, добавляем ему стили:

.footer {

    @extend %center;

    display: flex;

    flex–direction: row;

    justify–content: space–between;

    align–items: center;

    background: $colorSite;

    &\_\_text {

        padding–top: 38px;

        padding–bottom: 38px;

        font–family: 'Roboto', sans–serif;

        color: rgba(244, 242, 239, 0.95);

        font–family: Roboto;

        font–size: $sizeText;

        font–style: normal;

        font–weight: $weightFontRegular;

        line–height: $lineHeight

    }

    &\_\_icon {

        display: flex;

        gap: 15px;

    }

    &\_\_link {

        filter: drop–shadow(0px 4px 4px rgba(0, 0, 0, 0.25));

    }

    .link\_\_background {

        transition: 0.7s;

    }

    .link\_\_color {

        transition: 0.7s;

    }

    svg:hover .link\_\_background {

        fill: #C3B09D;

    }

    svg:hover .link\_\_color {

        fill: #F4F2EF;

    }

}

3.4 Добавление адаптива

Адаптив максимально очень важен для сайта. Дело в том, что, создав красивый, стильный сайт на своем компьютере мы видим отображение аккуратным, таким как его нам обрисовал дизайнер, но на других экранах эта красивая модель сайта может сильно съехать. Например, когда пользователь смотрит на наш сайт с телефона, все блоки могут перемешаться, встать не в презентабельном порядке, а шрифт может быть очень большим, для маленького экрана. Здесь и приходит на помощь адаптив.

Создадим адаптацию страницы index.html для разрешения 1280px. Для начала в код CSS добавляем строку:

@media (max–width: 1280px) {

}

В эту строку мы будем вписывать дополнительные стили для блоков и элементов блоков. В процессе написания диплома я переписала код с CSS в SCSS для удобства, так как там есть вложенность.

Напишем стили для блока headeri. Важно, не нужно повторять стили, которые уже были прописаны для компьютера, нужно дополнить или перезаписать те позиции, которые нужно исправить.

@media (max–width: 1280px) {

    .headeri {

        padding–left: calc(50% – 960px/2);

        padding–right: calc(50% – 960px/2);

        #menu {

            margin–left: 62%;

        }

        #menu li {

            font–size: 28px;

        }

        #burger–menu span {

            width: 60px;

        }

        #burger–menu span::after {

            width: 60px;

        }

        #burger–menu span:before {

            width: 60px;

        }

        #burger–menu.close span {

            width: 60px;

        }

        #burger–menu.close span:before {

            width: 60px;

        }

        &\_\_logo {

            font–size: 46px;

        }

    }

}

Таким же образом пишем код и для остальных блоков. Все блоки длолжны находиться внутри @media (max–width: 1280px):

.slog {

        padding–left: calc(50% – 960px/2);

        padding–right: calc(50% – 960px/2);

        &\_\_title {

            padding–top: 50px;

            font–size: 36px;

        }

        &\_\_text {

            font–size: 26px;

        }

        &\_\_img\_photo {

            max–width: 500px;

        }

    }

    .content {

        padding–left: calc(50% – 960px/2);

        padding–right: calc(50% – 960px/2);

        grid–template–columns: repeat(3, 300px);

        gap: 30px;

        &\_\_img {

            max–width: 300px;

        }

        &\_\_text {

            font–size: 20px;

        }

    }

    .button {

        &\_\_border {

            width: 230px;

        }

        &\_\_link {

            font–size: 20px;

        }

    }

    .footer {

        padding–left: calc(50% – 960px/2);

        padding–right: calc(50% – 960px/2);

        &\_\_text {

            font–size: 20px;

        }

        &\_\_link\_ico {

            width: 50px;

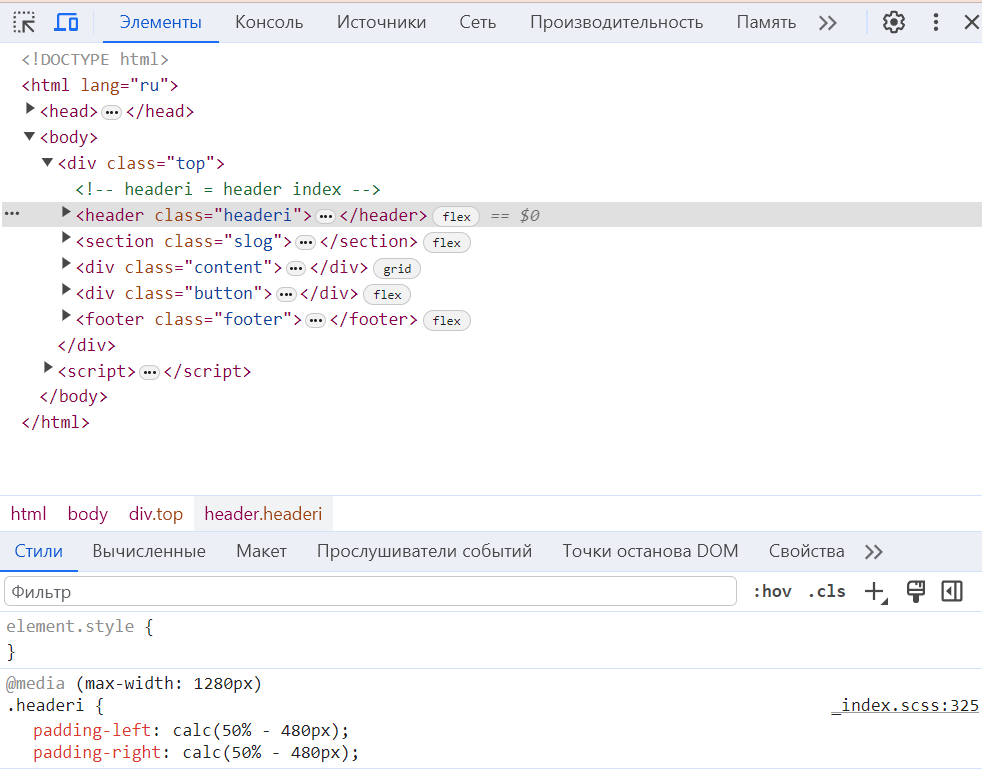
        }

    }

По этой аналогии делаем адаптацию для планшета, среднего и маленького телефона

3.5 Проверка внешнего вида на разных разрешениях экрана

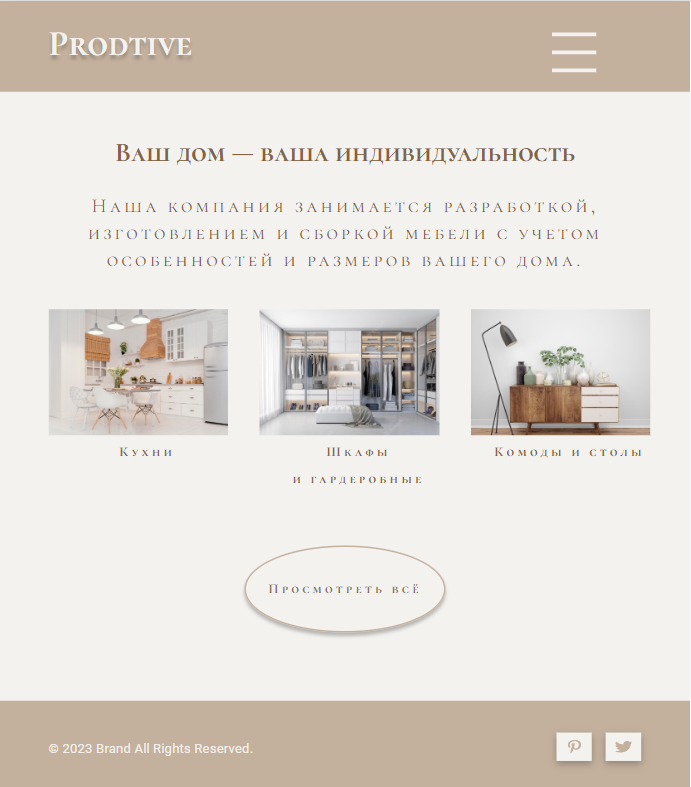
Внешний вид на разных экранах можно проверетьь через инструменты разработчика в браузере. Нажав на кнопку  можно посмотреть как наш сайт будет выглядеть на разных разрешениях экрана.



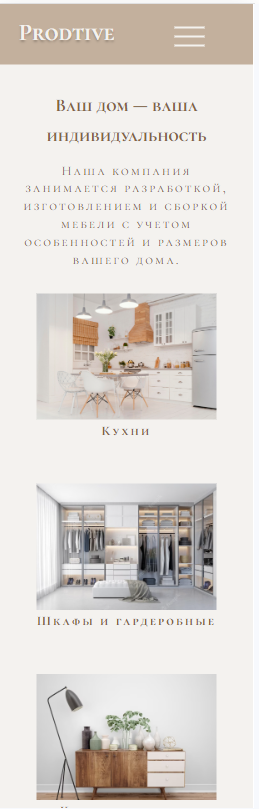
Страница на разрешении 1024 px:



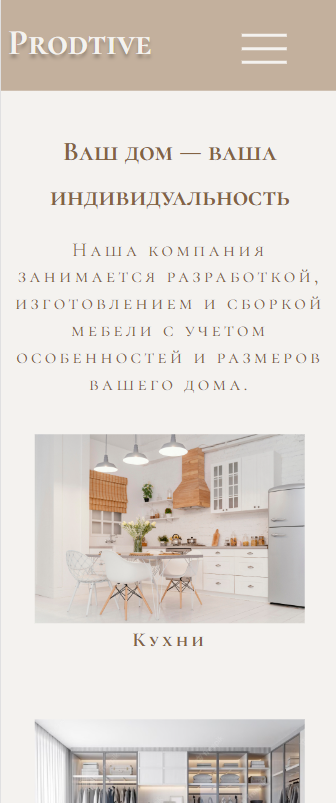
Среднестатистическое разрешение планшета 768px:



Разрешение 425px:



Разрешение 375px:



3.6 Проверка на валидность

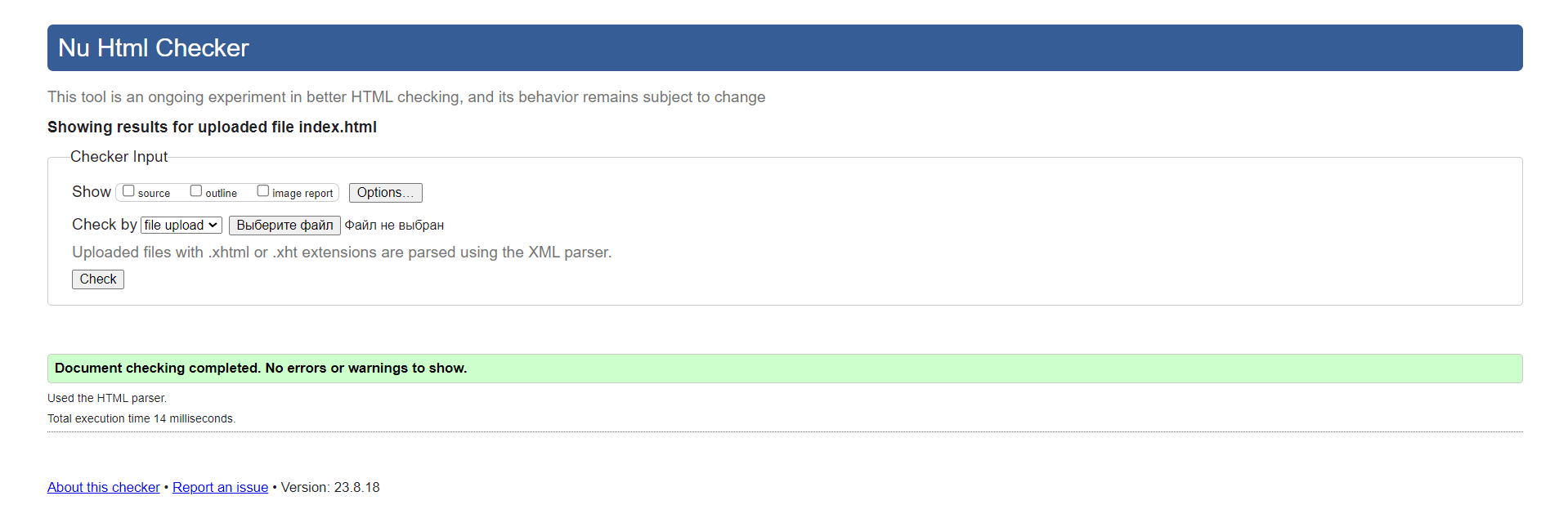
Правильный, валидный html–код — это код, написанный по спецификации W3C, в которой собраны стандарты и рекомендации по удобству и универсальности Всемирной сети.

При написании кода стоит придерживаться этих правил. Они в целом довольно похожи на правила обычного, привычного нам русского языка. Например, если вы не закроете тег (в русском языке — не закончите правильно абзац текста) — будет нарушена структура и смысловая составляющая. Проверка кода на валидность позволяет увидеть все подобные ошибки и исправить их.

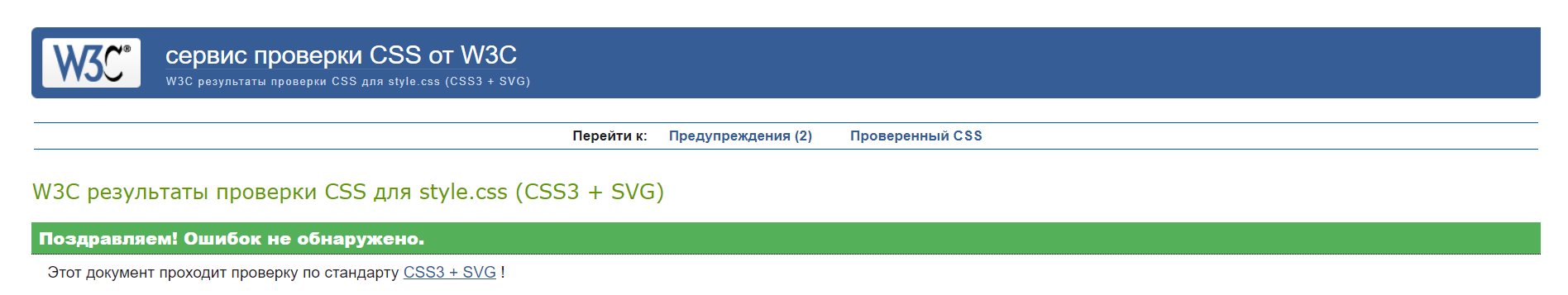
Но конечно, дело не только в структуре. Ваш код могут смотреть как другие разработчики, так и браузер, а также поисковые машины. И чтобы страница быстрее загружалась, правильнее обрабатывалась, а поисковые машины верно понимали смысл всех тегов, важно писать валидный код.

Проверяем HTML с помощью сайта: <https://validator.w3.org/nu/#file>

Результат:



CSS код проверяем с помощью сайта [https://jigsaw.w3.org/css–validator/validator](https://jigsaw.w3.org/css-validator/validator)



3.7 Проверка типографики

Изначально типографика относилась только к печатным изданиям и графическому дизайну. Но чем больше материалов попадало в сеть — тем быстрее понятие распространялось в сфере создания сайтов. Сейчас большинство под «Типографикой» понимают совокупность правил для создания страниц. Визуальное оформление отвечает за общее впечатление от текста и вложенных смыслов. Понятие «вёрстка» тоже прошло долгий путь и переросло из «создание макета на печать» в «создание сайта».

Типографика — это свод правил по оформлению текста для удобства пользователя. Чтобы каждый нашёл нужную идею, быстро её прочёл, при желании задержался и насладился визуальной концепцией ресурса.

Наш проект после проверки Типографа:

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF–8">

    <meta name="viewport" content="width=device–width, initial–scale=1.0">

    <title>Главная</title>

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link href="https://fonts.googleapis.com/css2?family=Cormorant+SC:wght@300;400;500;600;700&display=swap"

        rel="stylesheet">

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400&display=swap" rel="stylesheet">

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

</head>

<body>

    <div class="top">

        <!–– headeri = header index ––>

        <header class="headeri">

            <a href="index.html" class="headeri\_\_logo">Prodtive</a>

            <div id="burger–menu">

                <span></span>

            </div>

            <div id="menu">

                <ul>

                    <li class="menu\_\_bord menu\_\_center">Меню</li>

                    <li class="menu\_psition"><a href="kitchen.html">Кухни</a></li>

                    <li class="menu\_psition"><a href="wardrobe.html">Шкафы и&nbsp;гардеробные</a></li>

                    <li class="menu\_\_bord"><a href="dresser.html">Комоды и&nbsp;столы</a></li>

                    <li class="menu\_psition"><a href="form.html">Оставить заявку</a></li>

                    <li class="menu\_psition"><a href="about.html">О&nbsp;нас</a></li>

                    <li class="menu\_psition"><a href="contacts.html">Контакты</a></li>

                </ul>

            </div>

        </header>

        <section class="slog">

            <div class="slog\_\_content">

                <h1 class="slog\_\_title">Ваш дом&nbsp;&mdash; ваша индивидуальность </h1>

                <p class="slog\_\_text">Наша компания занимается разработкой, изготовлением и&nbsp;сборкой мебели

                    с&nbsp;учетом

                    особенностей и&nbsp;размеров вашего дома.</p>

            </div>

            <div class="slog\_\_img">

                <img src="img/index/slog\_\_img.jpg" alt="photo">

            </div>

        </section>

        <div class="content">

            <div class="item content\_\_item1">

                <a href="kitchen.html" class="content\_\_link"><img src="img/index/link1.jpg" alt="photo"><span

                        class="content\_\_text">Кухни</span></a>

            </div>

            <div class="item content\_\_item2">

                <a href="wardrobe.html" class="content\_\_link"><img src="img/index/link2.jpg" alt="photo"><span

                        class="content\_\_text">Шкафы и&nbsp;гардеробные</span></a>

            </div>

            <div class="item content\_\_item3">

                <a href="dresser.html" class="content\_\_link"> <img src="img/index/link3.png" alt="photo"><span

                        class="content\_\_text">Комоды и&nbsp;столы </span></a>

            </div>

        </div>

        <div class="button">

            <div class="button\_\_border">

                <a href="#" class="button\_\_link">Просмотреть всё</a>

            </div>

        </div>

        <footer class="footer">

            <h3 class="footer\_\_text">&copy;&nbsp;2023 Brand All Rights Reserved.</h3>

            <div class="footer\_\_icon">

                <a href="https://ru.pinterest.com/" class="footer\_\_link"><svg width="62" height="49" viewBox="0 0 62 49"

                        fill="none" xmlns="http://www.w3.org/2000/svg">

                        <path class="link\_\_background" d="M61.3882 0H0V49H61.3882V0Z" fill="#F4F2EF" />

                        <g clip–path="url(#clip0\_2\_94)">

                            <path class="link\_\_color"

                                d="M31.7472 12.561C25.531 12.561 19.3876 15.8341 19.3876 21.1312C19.3876 24.5 21.7868 26.4141 23.2409 26.4141C23.8407 26.4141 24.186 25.0934 24.186 24.7201C24.186 24.2751 22.7501 23.3276 22.7501 21.4758C22.7501 17.6285 26.458 14.901 31.2564 14.901C35.3824 14.901 38.4359 16.7528 38.4359 20.1551C38.4359 22.696 37.1454 27.462 32.965 27.462C31.4564 27.462 30.1659 26.6007 30.1659 25.3661C30.1659 23.5573 31.7654 21.806 31.7654 19.9397C31.7654 16.772 26.0763 17.3462 26.0763 21.1743C26.0763 21.9782 26.2035 22.8683 26.6579 23.6004C25.8218 26.4428 24.1133 30.6776 24.1133 33.6061C24.1133 34.5105 24.2769 35.4006 24.3859 36.305C24.5919 36.4868 24.4889 36.4677 24.804 36.3768C27.8575 33.075 27.7485 32.429 29.1299 28.108C29.8751 29.2277 31.8017 29.8307 33.3285 29.8307C39.7628 29.8307 42.6527 24.878 42.6527 20.4135C42.6527 15.6618 37.4544 12.561 31.7472 12.561Z"

                                fill="#C3B09D" />

                        </g>

                        <defs>

                            <clipPath id="clip0\_2\_94">

                                <rect width="23.2652" height="24.5" fill="white" transform="translate(19.3876 12.25)" />

                            </clipPath>

                        </defs>

                    </svg>

                </a>

                <a href="https://ru.pinterest.com/" class="footer\_\_link"><svg width="62" height="49" viewBox="0 0 62 49"

                        fill="none" xmlns="http://www.w3.org/2000/svg">

                        <path class="link\_\_background" d="M62 0H0.611816V49H62V0Z" fill="#F4F2EF" />

                        <g clip–path="url(#clip0\_2\_90)">

                            <path class="link\_\_color"

                                d="M43.9537 19.5098C43.9734 19.7274 43.9734 19.9451 43.9734 20.1627C43.9734 26.8007 37.5765 34.4492 25.8848 34.4492C22.2829 34.4492 18.9368 33.6252 16.1221 32.1951C16.6339 32.2417 17.1259 32.2572 17.6574 32.2572C20.6295 32.2572 23.3654 31.4644 25.5502 30.112C22.7552 30.0653 20.413 28.6196 19.606 26.6297C19.9997 26.6763 20.3933 26.7074 20.8067 26.7074C21.3775 26.7074 21.9483 26.6452 22.4797 26.5364C19.5666 26.07 17.3818 24.0491 17.3818 21.6084V21.5463C18.2281 21.9194 19.2123 22.1526 20.2555 22.1836C18.543 21.282 17.4212 19.7429 17.4212 18.0018C17.4212 17.0691 17.736 16.2141 18.2872 15.4679C21.4168 18.5148 26.121 20.5047 31.396 20.7223C31.2976 20.3492 31.2385 19.9606 31.2385 19.572C31.2385 16.8048 34.0728 14.5507 37.5961 14.5507C39.4266 14.5507 41.0799 15.157 42.2412 16.1364C43.678 15.9187 45.0558 15.499 46.2762 14.9238C45.8038 16.0898 44.8 17.0691 43.4812 17.6909C44.7606 17.5822 46.0007 17.3023 47.1422 16.9137C46.2763 17.9086 45.1937 18.7946 43.9537 19.5098Z"

                                fill="#C3B09D" />

                        </g>

                        <defs>

                            <clipPath id="clip0\_2\_90">

                                <rect width="31.0202" height="24.5" fill="white" transform="translate(16.1221 12.25)" />

                            </clipPath>

                        </defs>

                    </svg>

                </a>

            </div>

        </footer>

    </div>

    <script>

        var burgerMenu = document.getElementById('burger–menu');

        var overlay = document.getElementById('menu');

        burgerMenu.addEventListener('click', function () {

            this.classList.toggle("close");

            overlay.classList.toggle("overlay");

        });

    </script>

</body>

</html>

Заключение

В ходе выполнения данного проекта были рассмотрены теоретически аспекты разработки сайта, в последствии примененные на практики для создания собственного дизайна, по которому непосредственно мы писали сайт для мебельной компании.

Также были поэтапно выполнены все поставленные перед работой задачи.

1. Мы изучили всю литературу, касающеюся нашей темы;

2. Рассмотрели основные виды и методы добавления эффектов наведения и нажатия;

3. Изучили работу с семантическими элементами;

4. Создали дизайн проекта в дизайнерской программе Figma;

5. Написали сайт в соответствии с макетом. В разработке участвовали различные языки, такие как:

* стандартизированный язык гипертекстовой разметки документов для просмотра веб–страниц в браузере – HTML;
* код, который вы используется для стилизации нашей веб–страницы – CSS;
* SCSS – препроцессор CSS, который позволяет вкладывать правила CSS друг в друга, работать с переменными, писать собственные функции и использовать миксины (примеси) для создания отдельных тем одного макета
* JS – ультипарадигменный язык программирования, который поддерживает объектно–ориентированный, императивный и функциональный стили ;

6. А также добавили адаптив к разработанному сайту и добавили эффекты различного вида.

Для проверки кода я использовала Валидатор и Типограф в открытых источниках.

Список используемой литературы

1. «Что такое вебсайт? И для чего он нужен» – [https://pumpkinstudio.ru/blog/development/veb–sayt/](https://pumpkinstudio.ru/blog/development/veb-sayt/)
2. «Немного интересного о первом в мире сайте» – <https://pikabu.ru/story/nemnogo_interesnogo_o_pervom_v_mire_sayte_435664>
3. «Гид по Figma для начинающих веб–дизайнеров» – [https://tilda.education/articles–figma](https://tilda.education/articles-figma)
4. «Справочник HTML» – [https://hcdev–ru.pages.dev/html/](https://hcdev-ru.pages.dev/html/)
5. «CSS Справочник примеры и элементы» – <https://html5css.ru/cssref/default.php>
6. «Документация SCSS» – [https://sass–lang.com/documentation/](https://sass-lang.com/documentation/)
7. «Документация JS» – [https://js–documentation.netlify.app/](https://js-documentation.netlify.app/)
8. «Валидатор HTML» – <https://validator.w3.org/>
9. «Валидатор CSS» – [https://jigsaw.w3.org/css–validator/](https://jigsaw.w3.org/css-validator/)
10. «Типограф» – <https://www.artlebedev.ru/typograf/>
11. 47 готовых решений для создания Web–сайта / А.Г. Богданов и др. – Москва: Гостехиздат, 2021. – 272 c.
12. Гаевский, А.Ю. 100% самоучитель. Создание Web–страниц и Web–сайтов. HTML и JavaScript / А.Ю. Гаевский, В.А. Романовский. – М.: Триумф, 2022. – 464 c.
13. Дакетт, Д. HTML и CSS. Разработка и создание веб–сайтов / Д. Дакетт. – М.: Эксмо, 2020. – 756 c.
14. Ищенко, В.А. 47 готовых решений для создания Web–сайтов. / В.А. Ищенко. – М.: Триумф, 2019. – 721 c.
15. Снелл Абсолютно ясно о создании Web–страниц и Web–сайтов / Снелл, Нэд. – М.: Триумф, 2020. – 224 c.