**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**МЕХАНИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ**

**Кафедра веб-технологий и компьютерного моделирования**

**СОЗДАНИЕ АДАПТИВНОГО И ОТЗЫВЧИВОГО САЙТА**

Курсовая работа

Мисюкевич Дарьи Андреевны

студентки 2 курса, специальность «математика и информационные технологии»

Научный руководитель:

старший преподаватель

Ю.И. Пономаренко

Минск, 2020

# ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 2](#_Toc40900866)

ГЛАВА 1. [ОБЗОР ТЕХНОЛОГИЙ 3](#_Toc40900868)

[1.1 Виды вёрстки 3](#_Toc40900869)

[1.2 Преимущества адаптивного дизайна 5](#_Toc40900870)

[1.3 Основные техники реализации сайтов под любое разрешение 6](#_Toc40900871)

ГЛАВА 2. [РЕАЛИЗАЦИЯ ЭЛЕМЕНТОВ ВЕБ-СТРАНИЦЫ 7](#_Toc40900873)

[2.1 Меню 7](#_Toc40900874)

[2.2 Слайдер 8](#_Toc40900875)

[2.3 Секция, посвящённая предоставляемым услугам 9](#_Toc40900876)

[2.4 Галерея с примерами работ 9](#_Toc40900877)

[2.5 Секция с информацией о сотрудниках веб-студии 9](#_Toc40900878)

[2.6 Форма для заказа 10](#_Toc40900879)

[2.7 Модальноe окно с подтверждением заказа. 10](#_Toc40900880)

[ЗАКЛЮЧЕНИЕ 11](#_Toc40900881)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 12](#_Toc40900882)

# 

# ВВЕДЕНИЕ

Развитие интернет-технологий привело к тому, что посещение сайтов стало неотъемлемой частью жизни большинства людей. Сайты предоставляют пользователям много возможностей: чтение новостей, поиск информации, просмотр фильмов, запись на услуги и заказ товаров. С распространением мобильных устройств и ростом мобильного трафика важным требованием при разработке сайтов стали адаптивная и отзывчивая вёрстка. Эти виды вёрстки позволяют избежать необходимости создания и поддержки двух вариантов сайта – для настольных компьютеров и для мобильных устройств. Таким образом, создание адаптивного и отзывчивого веб-сайта является приоритетным направлением в веб-дизайне.

Задачей данной работы является создание веб-сайта для студии веб-дизайна с использованием адаптивной и отзывчивой вёрстки. На сайте требуется реализовать мобильное меню для навигации по веб-странице, адаптивный слайдер, секцию, посвящённую предоставляемым услугам, галерею с примерами работ, секцию с информацией о сотрудниках веб-студии, форму для заказа, модальноe окно с подтверждением заказа.

# ГЛАВА 1

# ОБЗОР ТЕХНОЛОГИЙ

## 1.1 Виды вёрстки

**Фиксированная вёрстка.** Фиксированная вёрстка — это вид вёрстки, при котором размеры элементов веб-страницы задаются с помощью абсолютных единиц измерения, наиболее распространёнными из которых являются пиксели. При изменении размеров окна браузера размеры элементов веб-страницы не меняются. На данный момент такой тип вёрстки считается устаревшим. Это связано с появлением устройств с абсолютно разными разрешениями экранов.

**Плюсы.** Наиболее быстрая и простая реализация. Результат всегда выглядит так, как было задумано, на мониторах с шириной больше ширины макета.

**Минусы.** На разрешениях, которые довольно сильно отличаются от целевого, сайт может отображаться некорректно. На мониторах с шириной больше ширины макета будет появляться много пустого пространства по краям, а на мониторах с шириной меньше ширины макета — горизонтальная прокрутка.

**Резиновая верстка.** Резиновая вёрстка — это вид вёрстки, отличающийся от фиксированной вёрстки заданием размеров элементов веб-страницы с помощью относительных единиц измерения. При изменении размеров окна браузера размеры элементов веб-страницы меняются. В результате получается, что при увеличении ширины окна браузера происходит увеличение ширины элементов страницы, а при уменьшении ширины окна браузера — уменьшение ширины элементов страницы.

**Плюсы.** Позволяет сайту достаточно хорошо отображаться на мониторах с любым разрешением.

**Минусы.** Более сложная реализация по сравнению с фиксированной вёрсткой. Тяжело добиться удобного дизайна при всех возможных разрешениях экранов. Резиновая вёрстка подходит не для всех элементов веб-страницы. Чрезмерное растяжение текстового блока приведёт к тому, что текст станет неудобным для чтения. Изменение размеров растровых графических изображений исказит их пропорции и ухудшит качество. [3]

**Адаптивная верстка.** Адаптивная вёрстка — это вид вёрстки, основанный на использовании медиа-запросов. Медиа-запросы содержат стили, описывающие вид элементов веб-страницы в заданных контрольных точках. При достижении контрольной точки стили элементов меняются в соответствии с правилами для данной контрольной точки. Таким образом, медиа-запросы позволяют подстраивать контент под различные устройства и параметры экранов.

**Плюсы.** Сайт корректно отображается на разных разрешениях экрана.

**Минусы.** Сложная реализация. Сайт может отображаться не очень красиво на разрешениях экрана, отличных от контрольных точек, так как расстояния между заданными точками могут быть достаточно большими. Нет плавного перехода между контрольными точками.

**Отзывчивая вёрстка.** Отзывчивая вёрстка — это вид вёрстки, сочетающий в себе использование относительных единиц измерения и медиа-запросов. В отличие от адаптивной вёрстки, при отзывчивой вёрстке веб-страница плавно изменяется между контрольными точками.

**Плюсы.** Хорошо адаптируется под любое разрешение экрана.

**Минусы.** Сложная реализация.

## 1.2 Преимущества адаптивного дизайна

Среди преимуществ адаптивного дизайна можно выделить следующие:

* Сайтом удобно пользоваться на любом устройстве.
* «Пользователям удобнее делиться контентом, если для него используется единый URL». [2]
* «Алгоритмы Google точнее индексируют ресурсы страницы, а не регистрируют разные ее версии». [2]
* «Создание одной адаптивной страницы занимает меньше времени, чем создание нескольких вариантов одного контента». [2]
* «Не требуется переадресация, что ускоряет загрузку». [2]

## 1.3 Основные техники реализации сайтов под любое разрешение

### Относительные значения для размеров и отступов. Относительные значения можно задавать, например, для width, height, margin, padding. Самый известный способ задания относительного размера элементов – указание в процентах (%). При этом значение высчитывается относительно родительского элемента. [1]

Также существуют значения относительно размера экрана:

* vw – 1% ширины окна.
* vh – 1% высоты окна.
* vmin – наименьшее из vw и vh;
* vmax – наибольшее из vw и vh.

### Относительные значения для размера шрифта. Существуют следующие относительные значения для шрифтов:

* em задаёт размер относительно шрифта родителя;
* rem задаёт размер относительно размера шрифта элемента *html*.

Также относительные размеры шрифта можно задавать с помощью %, vw, vh.

### Максимальные и минимальные размеры компонентов. Для задания максимальных и минимальных значений ширины и высоты компонента используются свойства max-width, min-width и max-height, min-height соответственно.

### Использование медиа-запросов. Медиа-запросы позволяют задать стили для определённого размера экрана и типа устройства на основе характеристик устройства. Все медиа-запросы начинаются с ключевого слова @media, за которым следуют условия.

Наиболее распространённые типы устройств, для которых можно задать стили с помощью медиа-запросов:

* all – все типы устройств. Является значением по умолчанию.
* print – принтеры и другие печатающие устройства.
* screen – экраны цветных мониторов.
* speech – речевые синтезаторы, а также программы для воспроизведения текста.

При написании медиа-запросов используются характеристики носителя. Значения, используемые при задании характеристик, являются контрольными точками. [5]

Наиболее распространённые параметры:

* width. Проверяет ширину экрана. Значение *min-wid*th применяет правило, если ширина экрана больше ширины, указанной в медиа-запросе, *max-width* – если меньше ширины, указанной в медиа-запросе.
* height. Проверяет высоту экрана. Используются значения *min-*height и *max-* height*.*
* orientation. Проверяет ориентацию экрана. Может принимать два значения: *portrait* и *landspace*.
* resolution. Проверяет разрешение экрана. Значение *min-*resolution проверяет минимальное разрешение экрана*, max-*resolution – максимальное.

# ГЛАВА 2

# РЕАЛИЗАЦИЯ ЭЛЕМЕНТОВ ВЕБ-СТРАНИЦЫ

## 2.1 Меню

Для реализации навигации по странице я использовала неупорядоченный список *ul*, содержащий ссылки (рисунок 2.1). Каждая ссылка имеет атрибут *href,* указывающий с помощью идентификатора *id* блок на странице, на который надо перейти при нажатии на ссылку.

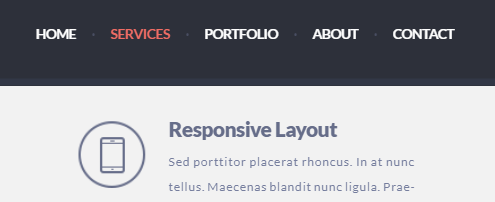


рисунок 2.1

Для выравнивания элементов списка по горизонтальной оси были использованы свойства *display: flex* и [*justify-content*](https://developer.mozilla.org/ru/docs/Web/CSS/justify-content)*: space-between.*

Элемент страницы, содержащий логотип и навигацию, зафиксирован с помощью свойства *position: fixed.* Это сделано для того, чтобы посетители сайта с помощью навигации могли перемещаться по странице, находясь в любой её части. При прокрутке страницы подсвечивается ссылка, соответствующая просматриваемому блоку.

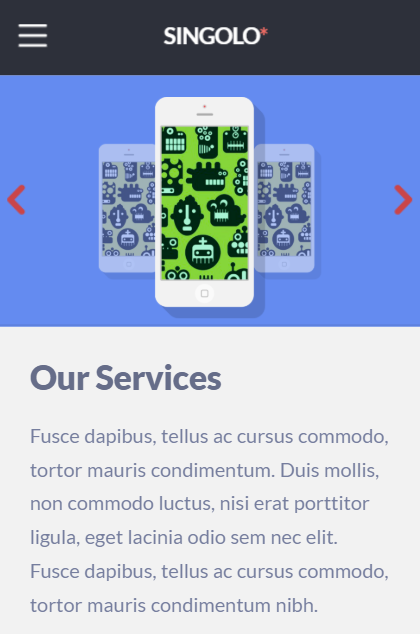
При ширине экрана меньше 768 пикселей логотип располагается по центру, а элементы навигации скрываются в «гамбургер-меню» (рисунок 2.2). Меню получило такое название, так как оно представляет собой три параллельные горизонтальные линии.

Чтобы показывать и скрывать меню при нажатии на «гамбургер», я использовала JavaScript. При открытом меню иконка «гамбургера» повёрнута на 90 градусов с помощью свойства *transform: rotate(90deg),* логотип перемещён влево (рисунок 2.3). Плавность перехода между состояниями открытого и закрытого меню реализована с помощью свойства *transition: 0.6s ease-in-out*.



рисунок 2.3

рисунок 2.2



Ссылки в меню поменяли своё расположение на вертикальное благодаря свойству *flex-direction: column.*

## 2.2 Слайдер

Для реализации смены слайдов я использовала JavaScript. В коде отслеживаются события нажатия на стрелки слайдера, и в зависимости от того, какая стрелка была нажата, осуществляется соответствующая анимация скрытия текущего слайда и показа нового (рисунок 2.4).

Плавного переключения слайдов удалось достичь с помощью CSS свойств *transition: 0.6s ease-in-out* и *animation: 0.6s ease-in-out*. Отступы и размеры телефонов задаются в процентах. Это позволяет слайдеру корректно отображаться на экранах любой ширины. Высота слайдера фиксированная.

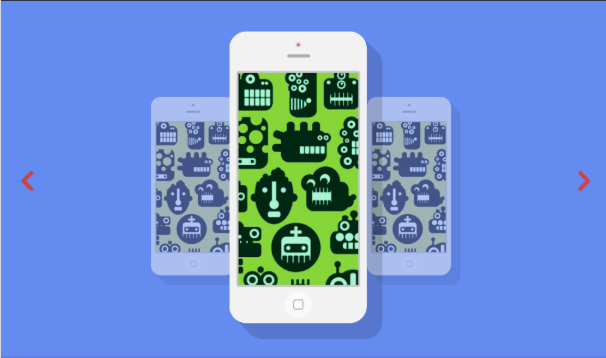


рисунок 2.4

При ширине экрана меньше 1020 пикселей высота слайдера задаётся с помощью функции *calc()*, позволяющей рассчитывать значения свойств CSS во время их определения: height: *calc(100vw \* 0.584).*

## 2.3 Секция, посвящённая предоставляемым услугам

Для представления услуг в виде трёхколоночной сетки использованы ранее упомянутые свойства *display: flex*, [*justify-content*](https://developer.mozilla.org/ru/docs/Web/CSS/justify-content)и свойство flex-wrap: wrap, которое позволяет элементам переноситься на новую строку (рисунок 2.5).

При ширине экрана от 768 пикселей до 1019 пикселей сетка двухколоночная, меняются расстояния как от элементов сетки до краёв блока, так и расстояния между соседними элементами (рисунок 2.6).

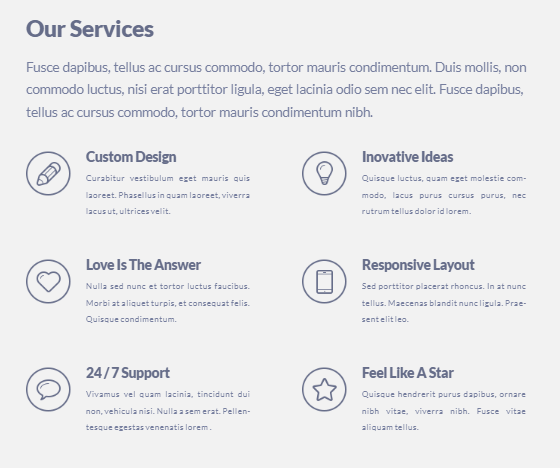


рисунок 2.6



рисунок 2.5

При ширине экрана меньше 768 пикселей элементы сетки выстраиваются в колонку свойством *flex-direction: column* (рисунок 2.7).



рисунок 2.7

## 2.4 Галерея с примерами работ

В данной секции я реализовала теги, при выборе каждого из которых в галерее перемешиваются изображения.

Для перехода от четырёхколоночной сетки (рисунок 2.8) к трёхколоночной (рисунок 2.9) при ширине экрана от 768 пикселей до 1019 пикселей, а затем к двухколоночной (рисунок 2.10) при ширине экрана менее 768 пикселей для изображений использовано свойство flex-basis, задающее их базовый размер по горизонтальной оси.

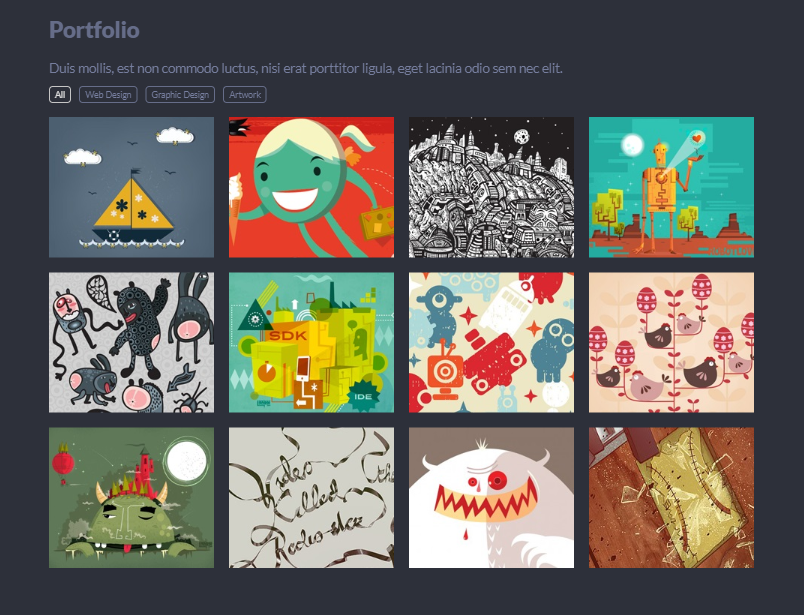


рисунок 2.8

## 2.5 Секция с информацией о сотрудниках веб-студии

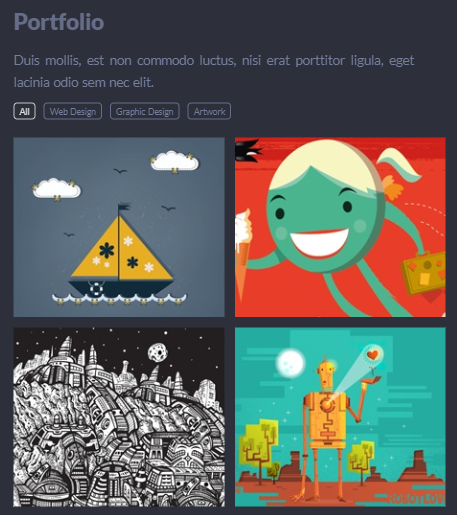


рисунок 2.10

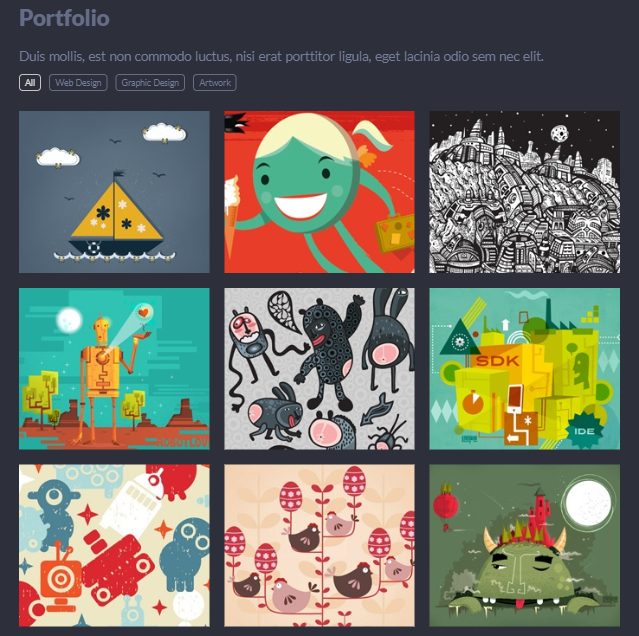


рисунок 2.9

Трёхколоночная сетка (рисунок 2.11) в данной секции также реализована с помощью *flexbox*. Ширина каждого из блоков, содержащих информацию о сотрудниках, задана в процентах. Ширина изображений равна 100%. Эти свойства позволяют блокам подстраиваться под изменение ширины экрана, а изображениям — растягиваться на всю ширину родительского блока.



рисунок 2.11

Заголовки с именами сотрудников имеют свойства *overflow: hidden, text- overflow: ellipsis,* *white-space: nowrap*. Это позволяет даже очень длинным именам занимать ровно одну строку. В результате заголовок не «вылезает» за пределы блока, а в конце него появляется многоточие (рисунок 2.12).

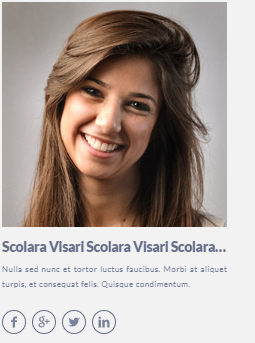


рисунок 2.12

При ширине экрана меньше 768 пикселей сетка становится одноколоночной (рисунок 2.13).

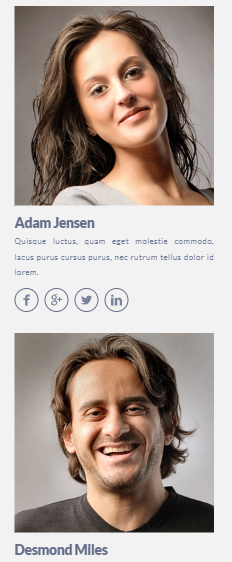


рисунок 2.13

## 2.6 Форма для заказа

Для реализации формы заказа я использовала элемент *form* и несколько полей разных типов для отправки имени, адреса электронной почты, темы сообщения и детального описания проекта (рисунок 2.14). Поле для ввода имени имеет тип «text» и базовую валидацию, представленную в виде атрибута pattern="[A-ZA-Яa-zа-яёЁ]+". Это регулярное выражение позволяет проверить, есть ли во введённом имени символы, отличные от букв русского и английского алфавитов, и, если есть, показать предупреждение о некорректных данных и запретить отправку формы. Поле для ввода адреса электронной почты имеет тип «email», что позволяет проверить корректность формата введённого адреса. Эти поля отмечены как обязательные для заполнения с помощью атрибута «required».

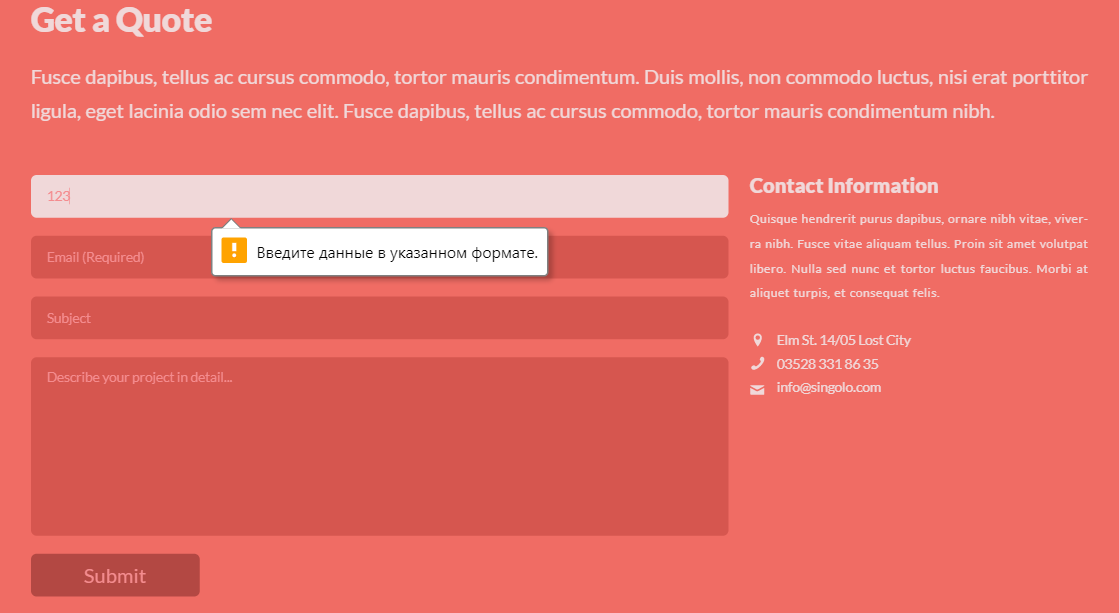


рисунок 2.14

Ширина формы и блока с контактной информацией задана в процентах. Это позволяет элементам корректно подстраиваться под изменение ширины экрана.

При ширине экрана меньше 768 пикселей контактная информация располагается под формой, а кнопка для отправки заказа растягивается на ширину формы для того, чтобы и на мобильных телефонах не возникало трудностей с попаданием по ней (рисунок 2.15).

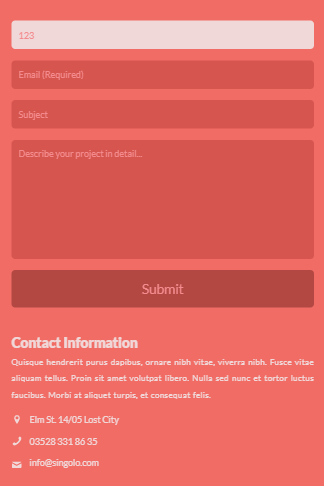


рисунок 2.15

2.7 Модальноe окно с подтверждением заказа

Для отображения модального окна (рисунок 2.16) я использовала JavaScript. В коде отслеживается событие нажатия на кнопку отправки заказа и проверяется корректность введённых в форму данных.

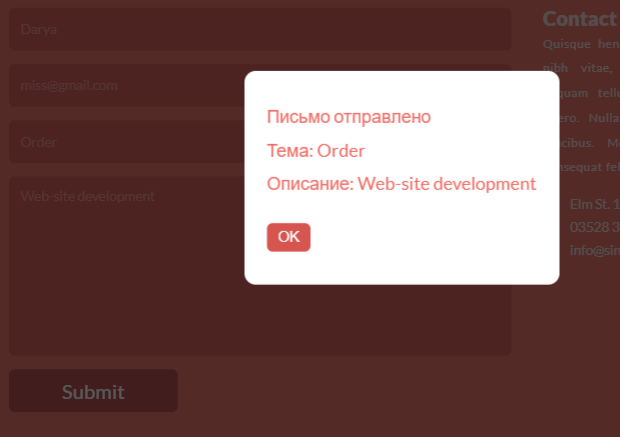


рисунок 2.16

Если форма прошла валидацию, данные из полей считываются и отображаются в модальном окне, ранее скрытом с помощью свойства display: none. Под модальным окном появляется тёмный блок, полностью перекрывающий окно браузера. Этот эффект достигается с помощью фиксированного позиционирования блока: *position: fixed, left: 0, right: 0, top: 0, bottom: 0*, а также свойства *z-index*, имитирующего перпендикулярное экрану третье измерение и позволяющего накладывать элементы друг на друга. [6] Чем больше значение этого свойства, тем ближе располагается элемент на странице. Таким образом, значения этого свойства подобраны так, чтобы блок перекрывал все элементы на странице, кроме модального окна с подтверждением заказа.

На модальном окне расположена кнопка, при нажатии на которую окно и тёмный фон скрываются, а содержимое полей формы очищается.

# ЗАКЛЮЧЕНИЕ

Результатом выполнения мной данной курсовой работы является создание адаптивного и отзывчивого веб-сайта с использованием HTML, CSS, JavaScript.

На веб-сайте реализованы мобильное меню для навигации по веб-странице, адаптивный слайдер, секция, посвящённая предоставляемым услугам, галерея с примерами работ, секция с информацией о сотрудниках веб-студии, форма для заказа, модальноe окно с подтверждением заказа.

При создании веб-сайта я использовала технологию flexbox, предоставляющую удобные инструменты, с помощью которых можно контролировать выравнивание относительно горизонтальной и вертикальной осей и размеры элементов, а также распределение свободного места между элементами. Это позволило мне создать гибкие макеты, состоящие из разного количества колонок при разной ширине экрана.

Изменения элементов веб-страницы в зависимости от ширины экрана удалось достичь с помощью мощной технологии — медиа-запросов. Так, например, благодаря медиа-запросам, при уменьшении ширины до определённого значения я скрыла панель навигации сверху и реализовала «гамбургер-меню».

Разработанный веб-сайт корректно отображается на устройствах с шириной экрана более 320 пикселей.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Адаптивная вёрстка сайтов: обзор подходов и CSS фреймворков / JazzTeam Software Development Company [Электронный ресурс]. — 2017. — Режим доступа: https://jazzteam.org/ru/technical-articles/overview-of-approaches-and-css-frameworks-for-adaptive-web-page-layout. — Дата доступа: 15.05.2020.

2. Адаптивный дизайн / Поиск для разработчиков / Google Developers [Электронный ресурс]. — 2020. — Режим доступа: https://developers.google.com/search/mobile-sites/mobile-seo/responsive-design. – Дата доступа: 21.05.2020.

3. Резиновая верстка сайтов: плюсы и минусы [Электронный ресурс]. — 2018. — Режим доступа: https://webstudio2u.net/ru/programming/386-unfixed-layout.html. — Дата доступа: 19.05.2020.

4. 4 степени гибкости Вашего сайта Оцениваем адаптивность HTML-верстки / Веб студия Флаг. Разработка сайтов в Екатеринбурге [Электронный ресурс]. — 2015. — Режим доступа: https://flagstudio.ru/blog/4-stepeni-adaptivnosti-html-verstki. — Дата доступа: 12.05.2020.2

5. CSS3-медиазапросы [Электронный ресурс]. — 2019. — Режим доступа: https://html5book.ru/css3-mediazaprosy. – Дата доступа: 15.05.2020.

6. z-index / htmlbook.ru [Электронный ресурс]. — 2002—2020. — Режим доступа: http://htmlbook.ru/css/z-index. — Дата доступа: 20.05.2020.