МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФГАОУ ВО  
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

Институт радиоэлектроники и информационной безопасности

Кафедра «Радиоэлектроника и телекоммуникации»

**ОТЧЁТ**

по лабораторной работе №2

«ИСПОЛЬЗОВАНИЕ CSS ПРИ РАЗРАБОТКЕ WEB -САЙТА»

по дисциплине

«Коммуникации в сети Интернет и принципы построения сайтов»

Выполнил:

студент гр. ИС/б-19-2-о

Марковская Д.В.

Защитил с оценкой: \_\_\_\_\_\_

Принял: ассистент Табакаев Д.И.

Севастополь

2020

**СОДЕРЖАНИЕ**

[1. Описание лабораторной работы 3](#_Toc38196757)

[1.1. Цель работы 3](#_Toc38196758)

[1.2. Индивидуальное задание 3](#_Toc38196759)

[1.3. Теоретические сведения 3](#_Toc38196760)

[2. Выполнение лабораторной работы 9](#_Toc38196761)

[2.1. Текст программы 9](#_Toc38196762)

[2.2. Таблица имя и назначение тегов 15](#_Toc38196763)

[2.3. Результат выполнения работы 17](#_Toc38196764)

# 1. Описание лабораторной работы

Цель работы

Научиться работать с каскадными таблицами стилей, а также изучить основные методы работы с селекторами и построения сайта блочной верстки.

Индивидуальное задание

Создать сайт, состоящий и нескольких веб – страниц, для офиса по продаже недвижимости.

Теоретические сведения

* + 1. Назначение стилевых таблиц

Собственные средства HTML (теги и их атрибуты) выполняют две основные роли: поддержку структуры документа (состав и взаимосвязи элементов) и определение внешнего вида визуальных элементов. Идея разделения описания внешнего вида документа от элементов, определяющих его структуру, воплотилась в технологии, называемой каскадными таблицами стилей (*Cascading Style Sheets, CSS*). Таблица стилей, действует подобно шаблону форматирования, может быть разработана отдельно от HTML-документа, а затем применена к нему. Изменяя содержимое таблицы стилей, можно изменять внешний вид HTML-документов, не затрагивая их структуры информационного содержания. Одна и та же таблица стилей может применяться к нескольким документам, и, наоборот, к одному и тому же документу может быть применено несколько таблиц стилей. В последнем случае браузер учитывает приоритеты таблиц и по определенным правилам разрешает возникающие конфликты, в результате чего таблицы выстраиваются неким каскадом.

Кроме технологичности стилизации HTML-документов, CSS обеспечивают еще две важные вещи: произвольное позиционирование элементов и создание визуальных эффектов, таких как полупрозрачность и трансформации графических изображений и текстов.

* + 1. Встраивание таблиц стилей в HTML-документ

Для применения каскадной таблицы стилей к HTML-документу необходимо ее связать с ним или встроить в него. Это можно сделать четырьмя способами:

1. Вставка непосредственно в заголовок HTML-документа. Правила таблицы стилей заключаются в контейнерный тег ***<style>***;

**<head>**

**<style>**

**H1{color: blue; font-size: 24pt}**

**B {font-style: italic}**

**</style>**

**</head>**

1. Вставка непосредственно тег виде строки описания в атрибуте *Style*; **<h1 style="color: blue; font-size:24pt">**
2. Импорт - вставка таблицы стилей из внешнего файла. Файл таблицы стилей является текстовым файлом с расширением ***css***. Оператор **@import** используется перед другими правилами таблицы стилей в контейнере ***<style>*** или в ***css***-файле.

**<style>**

**@import: url(**[**http://www.myserver.ru/css/mystyle.css**](http://www.myserver.ru/css/mystyle.css)**)**

**</style>**

1. Связывание с таблицей стилей в внешнем файле с помощью ссылки задаваемой тегом ***<link>***, который помещаемого в контейнер ***<head>***. Общий вид:

**href="http:// www.myserver.ru/css/mystyle.css"> </head>**

Таблица стилей, вставленная с помощью тега ***<style>***, действует на элементы только текущего HTML-документа, в котором этот тег находится. Если ту же таблицу необходимо применить и к другим документам, то ее код придется повторить в соответствующих HTML-страницах. При этом возрастает общий объем файлов сайта, а также трудозатраты в случае необходимости изменить его стиль. Чтобы избежать этого, используют импорт или связывание таблиц из внешних ***css***-файлов. Когда требуется изменить параметры стилей для отдельных элементов (например, их оформление), используют атрибут style. Возможно также комбинирование всех способов встраивания таблиц.

В записях таблиц стилей можно задавать комментарии, которые задаются символами /\* и \*/.

В тег ***<style>*** имеет следующие атрибуты:

***type*** - для каскадных таблиц стилей всегда имеет значение ***text***/***css***;

В ***media*** - определяет тип устройства вывода. Браузеры обычно используют следующие значения: ***screen*** (экран), ***print*** (печать) и ***all*** (все). Можно создать стили отдельно для отображения документа на экране монитора и для вывода на печать.

* + 1. Типы селекторов

CSS состоит из **правил**, а каждое правило – из **селектора** и **блока объявлений**. Блок объявлений содержит CSS-свойства, определяющие отображение элемента веб-страницы в браузере. Селектор отвечает за выбор этого самого элемента. Селектор служит для однозначной идентификации HTML элемента средствами CSS. Он позволяет выбирать именно тот элемент (или группу элементов), который нужен. С помощью **простых** селекторов можно выбирать:

* все объекты – универсальный селектор;
* объекты определенного типа;
* объекты с заданным классом;
* объект с определенным идентификатором;
* объекты с определенными характеристиками – селекторы атрибутов.

Объединяя простые селекторы можно выбирать объекты по более сложным правилам:

* объекты, находящиеся внутри какого-то объекта – селектор потомка;
* объекты, непосредственно вложенные в какой-то объект – дочерний селектор;
* объект, расположенный после другого объекта – сестринский селектор.

Также существуют селекторы псевдоклассов и псевдоэлементов. Они позволяют назначать стили элементам, которые зависят не только от разметки, но и от состояния документа.

* + 1. Универсальный селектор

Предназначен для выбора всех элементов. Стили, указанные для универсального селектора применяются ко всем элементам сразу.

Обычно применяется для сброса зависящих от браузера начальных значений стилей (в частности, отступов).

**\* {**

**padding: 0;**

**margin: 0;**

**}**

* + 1. Селектор типа

Предназначен для выбора всех элементов определенного типа. Стили применяются ко всем элементам указанного типа не зависимо от уровня вложенности. Применяется для задания общих, для всех элементов определенного типа, стилей. Общий вид:

**Имя тега { описание }**

**Пример**. Для всех абзацев (тег p) установить размер шрифта 12px.

**p {font-size: 12px;}**

* + 1. Селектор класса

Предназначен для выбора всех элементов по имени класса (по значению атрибута *class*). Стили применяются к любым тегам с

соответствующим классом. Важно учитывать, что в отличие от названий HTML-тегов, в названиях классов различаются большие и маленькие буквы. То есть *class="active" и class="Active"* – это совсем разные классы. Общий вид:

**.Имя класса { описание }**

**Пример**. Задать зеленый цвет текста в любых тегах с классом ***active***. **.active {color: #0f0; }**

Комбинируя селектор класса и селектор типа, можно объединить их свойства.

**Пример**. Задать зеленый цвет текста только для абзацев с классом ***active***.

**p.active {color: #0f0; }**

HTML позволяет задавать в качестве значения атрибута ***class*** список разделенных пробелами названий (порядок следования не имеет значения). Другими словами, один элемент может иметь сразу несколько классов: **<div class="panel hint active"></div>**

Стили, относящиеся к каждому из перечисленных классов, будут, объединяясь, применяться к этому элементу. IE6 не понимает такой записи.

Пример. Правило распространяется только на элементы, в списке классов которых встречаются (в любом порядке) и класс *popup* и класс *active*.

**.popup.active { color: #0f0;}**

Концепция классов, наиболее часто применяется при верстке. Например, верстать страницу, используя в качестве контейнеров теги *div*, задавая им определенные классы (в соответствии с функциональным назначением):

**<div class="header">**

**<!--элементы шапки сайта-->**

**</div>**

**<div class="sideBar">**

**<!--элементы панели меню-->**

**</div>**

**<div class="content">**

**<!--основное содержимое-->**

**</div>**

* + 1. Селектор идентификатора

Предназначен для выбора элемента по уникальному идентификатору (значению атрибута *id*). Позволяет задать стили конкретному HTML-элементу. Общий вид:

**#Имя идентификатора { описание }**

**Пример**. Задать зеленый цвет текста в любом теге с *id="active"*

**#active { color: #0f0; }**

Комбинируя селектор идентификатора и селектор типа, можно объединить их свойства. Учитывая, что id на странице должен быть уникальным, смысла в таком комбинировании не много. Как правило, идентификаторы применяются там, где предполагается работа скриптов. Например, в формах или в каких-то динамических элементах. Так же с помощью *id* можно подчеркнуть уникальность элемента, его присутствие в единственном экземпляре.

Пример. Задать зеленый цвет только для элемента списка с данным *id*.

**li#active { color: #0f0; }**

* + 1. Единицы измерения

Единицы измерения CSS используются для указания размеров различных элементов. Есть абсолютные и относительные единицы измерения. Абсолютные единицы не зависят от устройства вывода, а относительные единицы определяют размер элемента относительно значения размера, используемого в родительском элементе. Ниже представлена таблица и подробное описание каждой единицы измерения CSS.

Абсолютные единицы:

**Миллиметр** - *mm*, сантиметр - *cm* и дюйм – *in*. Само собой разумеется, что это абсолютные единицы измерения. Один *cm* = 0.39370*in*, *1in* = 2.54*cm* и 10*mm* = 1*cm*. Компьютерные дисплеи плохо вычисляют данные единицы измерения, таким образом, у этих величин ограниченное применение и обычно их используют при указании размера для вывода страниц на печать.

**Пиксели** – *px*. Пиксель – это маленькая точка на экране. Пиксели определяют размер элемента. Использование пикселей дает вам точный контроль над размером элемента, позволяя вам точно вычислить его ширину и высоту, это будет полезным для точной разметки дизайна страницы. Однако, есть несколько минусов использования пикселей, вы должны быть осведомлены о них:

Во-первых, установка размера шрифта с помощью пиксельных единиц не позволяет пользователю изменять размер шрифта с помощью настроек в браузере. Если размер шрифта 12 пикселей, он всегда будет иметь высоту 12 точек, независимо от того, что пользователь установил размер шрифта по умолчанию в браузере. Так что, если вы решили указать размер шрифтов, следует подумать об использовании другой единицы измерения.

Во-вторых, когда речь заходит о печатных средствах массовой информации, пиксели не имеют реального значения. При разработке

документа для печати, устройство печати должно будет само догадываться о том, что вы имели ввиду с точки зрения физических размеров. Хотя обычно можно просмотреть документ и внести изменения перед печатью

**Точки - *pt* и пики – *pc***. Эти единицы измерения чаще всего используются для указания именно размера шрифта, например, в обычном блокноте, мы часто указываем размер текста равный 14, это значение как раз и есть единицы измерения *pt*.

Точка (*1pt*) равна 1/72 дюйма, в то время как пика (*1pc*) равна 1/6 дюйма (*1pc = 12pt*). Документы, предназначенные для печати, будут иметь возможность сообщить устройству именно тот размер шрифта, который следует использовать при печати. Цифровым дисплеям, однако, придется самим догадываться, как конвертировать эти единицы в пиксели, и нет никаких реальных универсальных способов узнать, как это будет сделано. Поскольку точки (pt) были использованы с первых дней Интернета, большинство браузеров автоматически могут установить соотношения между пикселями и точками, но это по сути неправильно. Помните, что небольшие дисплеи сегодня могут иметь высокое разрешение, так что определить то, насколько большой «дюйм на экране» будет практически невозможно на устройствах. Для указания размера элементов на веб-страницах следует избегать использования этих единиц.

**Относительные единицы:**

* **Процент - %.** Самая простая единица измерения, это процент (%), она не имеет напрямую никакого отношения к размеру шрифта или элемента в целом и может быть использована в комбинации с другими единицами измерения, указывающими величину. Размер установленный в процентах, напрямую зависит от размера родительского элемента, например, размер шрифта задается относительно размера шрифта родительского элемента, также высота и ширина выражается в процентах относительно высоты и ширины родительского элемента;
* **Вычисляемая x-высота – *ex***. Единица измерения ex используется достаточно редко. В качестве основы для размера 1ex используется высота символа "x" в нижнем регистре указанного шрифта. Большинство браузеров не поддерживает эту единицу измерения должным образом, и она не рекомендуется для использования в документах, предназначенных для браузеров;
* **Вычисляемая единица – *em***. *Em* является относительной единицей измерения. Один *em* равен *16px*. Если *em* используется для определенного элемента, то за 1em принимается размер шрифта его родителя.

Выполнение лабораторной работы

Текст программы\

Styles.css

body {  
margin: 0;  
font-family: 'Amatic SC', cursive;  
font-size: 15px;  
color: [#4a0000](https://vk.com/im?sel=67774049&st=%234a0000)  
line-height: 1.6;  
}  
\*,  
\*:after {  
box-sizing: border-box;  
}  
.container {  
width: 100%;  
max-width: 1000px;  
margin: 0 auto;  
}  
.intro {  
display: flex;  
flex-direction: column;  
justify-content: center;  
width:100%;  
height: 100vh;  
background: url("../css/images/fon.jpg") center;  
background-size: cover;  
}  
.intro\_2 {  
width: 100%;  
max-width: 700px;  
margin: 0 auto;  
text-align: center;  
}  
.intro\_3 {  
width: 100%;  
margin: 0 auto;  
}  
.intro\_title {  
color: [#ff0004](https://vk.com/im?sel=67774049&st=%23ff0004);  
font-size: 100px;  
font-weight: bold;  
line-height: 1.1;  
}  
.intro\_title:after {  
content:"";  
display: block;  
width: 80px;  
height: 3px;  
background-color: [#ff0004](https://vk.com/im?sel=67774049&st=%23ff0004);  
margin:70px auto;  
}  
.intro\_m {  
width: 100%;  
margin: 0 auto;  
text-align: center;  
color: #333;  
font-size: 18px;  
font-weight: bold;  
line-height: 1;  
padding-top: 70px;  
}  
.knp {  
display: inline-block;  
vertical-align: top;  
padding: 8px 19px;  
border: 3px solid [#ff0004](https://vk.com/im?sel=67774049&st=%23ff0004);  
color: [#ff0004](https://vk.com/im?sel=67774049&st=%23ff0004);  
font-size: 18px;  
font-weight: bold;  
text-decoration: none;  
transition: background 0.1s linear, color 0.1s linear;  
}  
.knp:hover {  
background-color: [#ff0004](https://vk.com/im?sel=67774049&st=%23ff0004);  
color: [#fff](https://vk.com/im?sel=67774049&st=%23fff);  
  
}  
.header {  
padding-top: 25px;  
width: 100%;  
position: absolute;  
top: 0;  
left: 0;  
right: 0;  
z-index: 1000;  
}  
.header\_2 {  
display: flex;  
justify-content: space-between;  
align-items: center;  
}  
.header\_logo {  
font-size: 40px;  
font-weight: 700;  
color: [#ff0004](https://vk.com/im?sel=67774049&st=%23ff0004);  
font-family: 'Lobster', cursive;  
}  
nav {  
font-size: 30px;  
  
}  
.nav\_link {  
color: [#ff0004](https://vk.com/im?sel=67774049&st=%23ff0004);  
text-decoration: none;  
display: inline-block;  
vertical-align: top;  
margin: 0 10px;  
}  
.nav\_link:hover {  
color: #700000;  
transition: color 0.1s linear;  
}  
h1, h2, h3, h4, h5, h6 {  
margin: 0;  
}  
.table {  
width:100%;  
padding-top: 55px;  
}  
table {  
text-align: left;  
border-collapse: collapse;  
border: 3px;  
margin: 0 auto ;  
border-style: solid;  
border-color: #333;  
}  
th {  
font-size: 40px;  
font-family: 'Pattaya', sans-serif;  
font-weight: 400;  
}  
td {  
font-size: 30px;  
}

1.html

<!DOCTYPE html>  
<html>  
 <head>  
 <meta charset=”utf-8”>  
 <link rel=”stylesheet” href=”css/style.css”>  
 <link href=”<https://fonts.googleapis.com/css2?family=Amatic+CS:wght@700&family=Lobster&display=swap>” rel=”stylesheet”>

<title>Ресторан</title>  
 </head>

<body>

<header class=”header”>

<div class=”container”>

<div class=”header\_2”>

<div class=”header\_logo”>Chili Pizza</div>

<nav>

<a class=”nav\_link” href=”../www/1.html”>ГЛАВНОЕ</a>

<a class=”nav\_link” href=”../www/2.html”>МЕНЮ</a>

<a class=”nav\_link” href=”../www/3.html”>ДОСТАВКА</a>

<a class=”nav\_link” href=”#”>КОНТАКТЫ</a>

</nav>

</div>

</div>

</header>

<div class=”intro”>

<div class=”container”>

<div class=”intro\_2”>

<h1 class=”intro\_title”>ДОБРО ПОЖАЛОВАТЬ В РЕСТОРАН “CHILI PIZZA”</h1>

<a class=”knp” href=”#”>УЗНАТЬ БОЛЬШЕ</a>

</div>

</div>

</div>

</body>

</html>

2.html

<!DOCTYPE html>  
<html>  
 <head>  
 <meta charset=”utf-8”>  
 <link rel=”stylesheet” href=”css/style.css”>  
 <link href=”<https://fonts.googleapis.com/css2?family=Amatic+CS:wght@700&family=Lobster&display=swap>” rel=”stylesheet”>

<title>Меню</title>  
 </head>

<body>

<header class=”header”>

<div class=”container”>

<div class=”header\_2”>

<div class=”header\_logo”>МЕНЮ</div>

<nav>

<a class=”nav\_link” href=”../www/1.html”>ГЛАВНОЕ</a>

<a class=”nav\_link” href=”../www/2.html”>МЕНЮ</a>

<a class=”nav\_link” href=”../www/3.html”>ДОСТАВКА</a>

<a class=”nav\_link” href=”#”>КОНТАКТЫ</a>

</nav>

</div>

</div>

</header

<div class=”intro”>

<div class=”intro\_3”>

<div class=”table”>

<table border=”1” cellpadding=”2” align=”cehter”>

<tr><th>Наименование</th><th>Цена</th><th>Вес</th><th>Наименование</th><th>

Цена</th><th>Вес</th></tr>

<tr><td>Маргарита</td><td>245 руб.</td><td>350 г.</td><td>Капрезе</td><td>399 руб.</td><td>470 г.</td></tr>

<tr><td>Детская</td><td>260 руб.</td><td>450 г.</td><td>Фаджоли</td><td>255 руб.</td><td>470 г.</td></tr>

<tr><td>Мафия</td><td>405 руб.</td><td>500 г.</td><td>Салями</td><td>315 руб.</td><td>450 г.</td></tr>

<tr><td>Кальцоне</td><td>285 руб.</td><td>535 г.</td><td>Квадро Формаджи</td><td>365 руб.</td><td>440 г.</td></tr>

<tr><td>Гавайская</td><td>305 руб.</td><td>500 г.</td><td>Техас</td><td>260 руб.</td><td>470 г.</td></tr>

<tr><td>Кантри</td><td>299 руб.</td><td>440 г.</td><td>Палермо</td><td>490 руб.</td><td>605 г.</td></tr>

<tr><td>Ранчо</td><td>355 руб.</td><td>630 г.</td><td>Сполетто</td><td>315 руб.</td><td>535 г.</td></tr>

<tr><td>Венеция</td><td>520 руб.</td><td>650 г.</td><td>Карбонара</td><td>315 руб.</td><td>450 г.</td></tr>

<tr><td>Барбекью</td><td>299 руб.</td><td>470 г.</td><td>Чили</td><td>365 руб.</td><td>450 г.</td></tr>

<tr><td>Наполи</td><td>515 руб.</td><td>710 г.</td><td>Нью-Йорк</td><td>330 руб.</td><td>490 г.</td></tr>

</table>

</div>

</div>

</div>

</body>

</html>

3.html

<!DOCTYPE html>

<html>

<head>

<meta charset=”utf-8”>

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

<link href=”<https://fonts.googleapis.com/css2?family=Amatic+SC:wght@700&family=Lobster&display=swap>” rel=”stylesheet”>

<title>ДОСТАВКА</title>

</head>

<body>  
 <header class=”header”>

<div class=”container>

<div class=”header\_2”>

<div class=”header\_logo>ДОСТАВКА</div>

<nav>

<a class=”nav\_link” href=”../www/1.html”>ГЛАВНОЕ</a>

<a class=”nav\_link” href=”../www/2.html”>МЕНЮ</a>

<a class=”nav\_link” href=”../www/3.html”>ДОСТАВКА</a>

<a class=”nav\_link” href=”#”>КОНТАКТЫ</a>

</nav>

</div>

</div>

</header>

<div class=”intro”>

<div class=”container”>

<div class=”intro\_m”>Бесплатная доставка еды по Симферополю при заказе от 500 руб. (с учетом всех акций и скидок), если сумма заказа меньше, доставка 200 руб.

<p>Бесплатная доставка по пригороду при заказе от 600р, если сумма заказа меньше, доставка 200р.</p>

АЭРОПОРТ, АЭРОФЛОТСКОЕ, АГРАРНОЕ, БЕЛОГЛИНКА, БОГДАНОВКА, БЕЛОЕ, ГРЭС, ДАВЫДОВКА, КОМСОМОЛЬСКОЕ, МИРНОЕ, МОЛОДЕЖНОЕ, СТОРОГОНОВКА

<p>От 1000р. Или доставка – 200р. </p>

АК-МЕЧЕТЬ, ДУБКИ, КАМЕНКА, ПЕРОВО, ФОНТАНЫ

<p>От 1500р. Или доставка 200р.</p>

Нов. АЭРОПОРТ, ДОНСКОЕ, КОЛЛЕКТИВНЫЕ САДЫ, ЛОЗОВОЕ, ТРУДОВОЕ, УКРОМНОЕ, ЧИСТЕНЬКОЕ  
 <p>Заказ еды на дом с доставкой – не роскошь, а средство быстро и недорого организовать стол внезапных или, наоборот, долгожданных гостей, заскочивших на чай старых друзей, с которыми вы давно не виделись, а также это прекрасная альтернатива походу в кафе или ресторан со своей половинкой. Кстати, одно из преимуществ нашей услуги состоит в том, что в этом случае вам уж точно никто не скажет: «Будьте как дома, но не забывайте, что вы в гостях!». </p>

<p>Наша доставка еды по Симферополю - это также оптимальный выход из ситуации, если некогда или неохота готовить. Позвольте себе иногда полениться, а наша служба заказа еды все сделает за вас: приготовит, надежно упакует, сервирует и доставит все в лучшем виде! </p>

<p>Наслаждайтесь едой – ведь это одно из самых больших удовольствий в жизни!</p>

<p>Бесплатная доставка еды по Симферополю при заказе от 500 руб. (с учетом всех акций и скидок), если сумма заказа меньше, доставка 200 руб.</p>

<p>Бесплатная доставка по пригороду при заказе от 600 руб., если сумма заказа меньше, доставка – 200 руб. (Давыдовка, Белоглинка, Богдановка, Мирное, Аграрное, Молодежное, Белое, ГРЭС, Аэрофлотский, Каменка, Аэропорт, Строгановка).</p>

</div>

</div>

</div>

</body>

</html>

**Таблица имя и назначение тегов**

Таблица 1 — Используемые свойства

|  |  |
| --- | --- |
| **Имя свойства** | **Назначение свойства** |
| Margin | Свойство margin задает все свойства отступов в одной строке. |
| padding | Свойство padding задает все свойства полей в одной строке. |
| Border | Свойство border объединяет в себе все свойства, которые используются для определения границы (рамки) элемента. |
| Font-size | Свойство font-size задает размер шрифта. |
| Flex-direction | CSS свойство , указывающее на то,как flex-элементы располагаются во flex-контейнере по главной оси и направлению. |
| vertical-align | Свойство vertical-align задает вертикальное выравнивание элемента. |
| transition | CSS свойство transition позволяет делать плавные переходы между двумя значениями какого-либо CSS свойства, по существу это означает простейшую анимацию. |
| height | Свойство height определяет высоту элемента. |
| width | Свойство width определяет ширину элемента. |
| color | Свойство color позволяет задать цвет текста. |
| background | Свойство background объединяет в себе все свойства, которые используются для определения фона страницы. |
| font-family | Свойство font-family задает шрифт для элемента. |
| text-decoration | Свойство text-decoration определяет оформление текста. |
| display | Свойство CSS display определяет, как элемент должен быть показан в документе. |
| float | Свойство CSS float определяет по какой стороне будет выравниваться объект. Остальные элементы будут обтекать его с других сторон. |
| position | CSS свойство position указывает какой тип позиционирования используется для элемента (статический, относительный, абсолютный или фиксированный). |
| max-width | Свойство CSS max-width устанавливает максимальную ширину элемента |
| Box-sizing | Применяется для изменения алгоритма расчета ширины и высоты элемента. |
| text-align | Свойство text-align определяет горизонтальное выравнивание текста в элементе. |
| font-weight | Свойство font-weight определяет насыщенность (толщину) символов в тексте. |
| top | Свойство CSS top указывает направление смещения позиционированного элемента от верхнего края. |

Результат выполнения работы

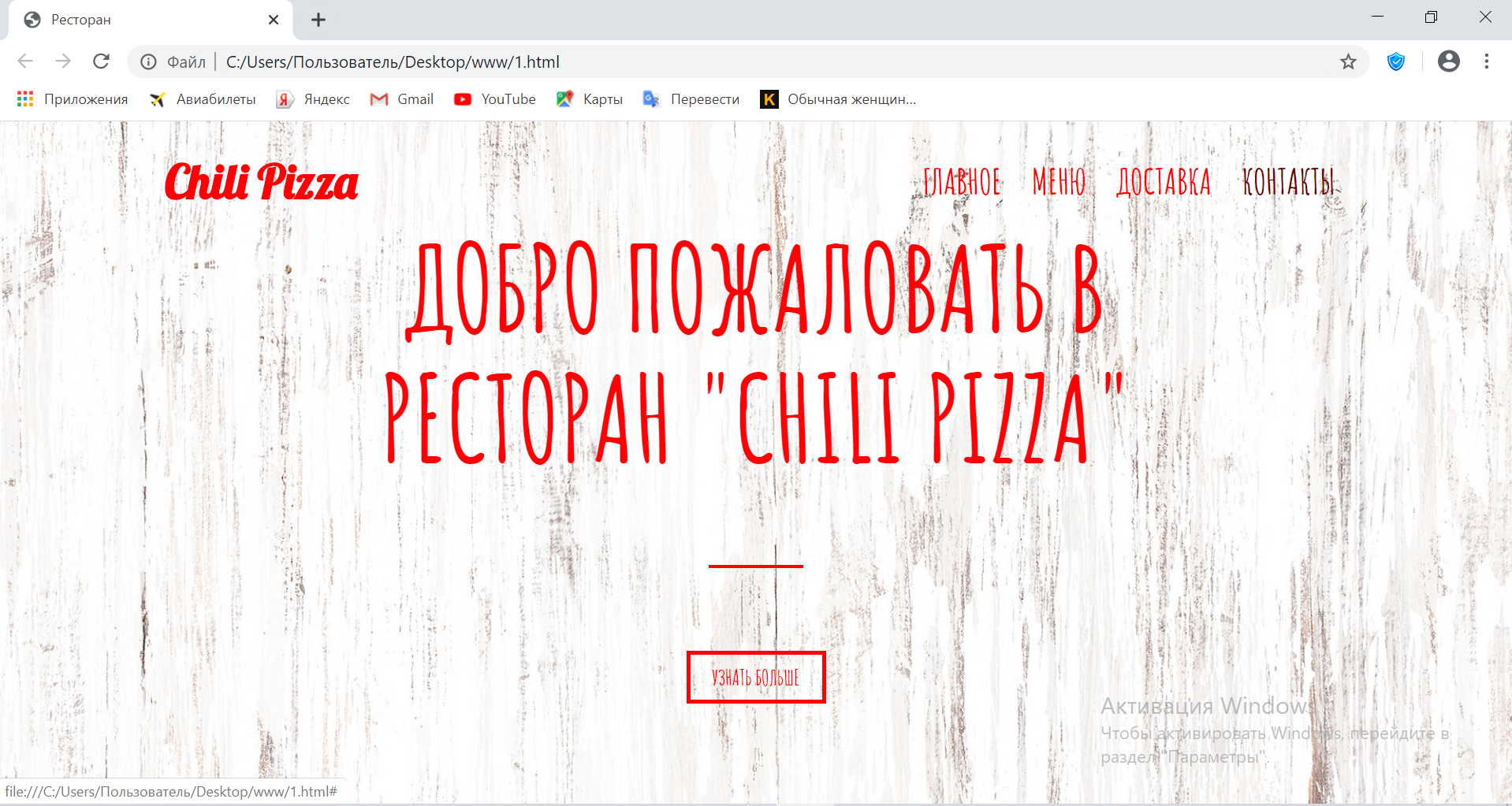


Рис. 1.1. — Первая страница сайта



Рис. 1.2. — Вторая страница сайта

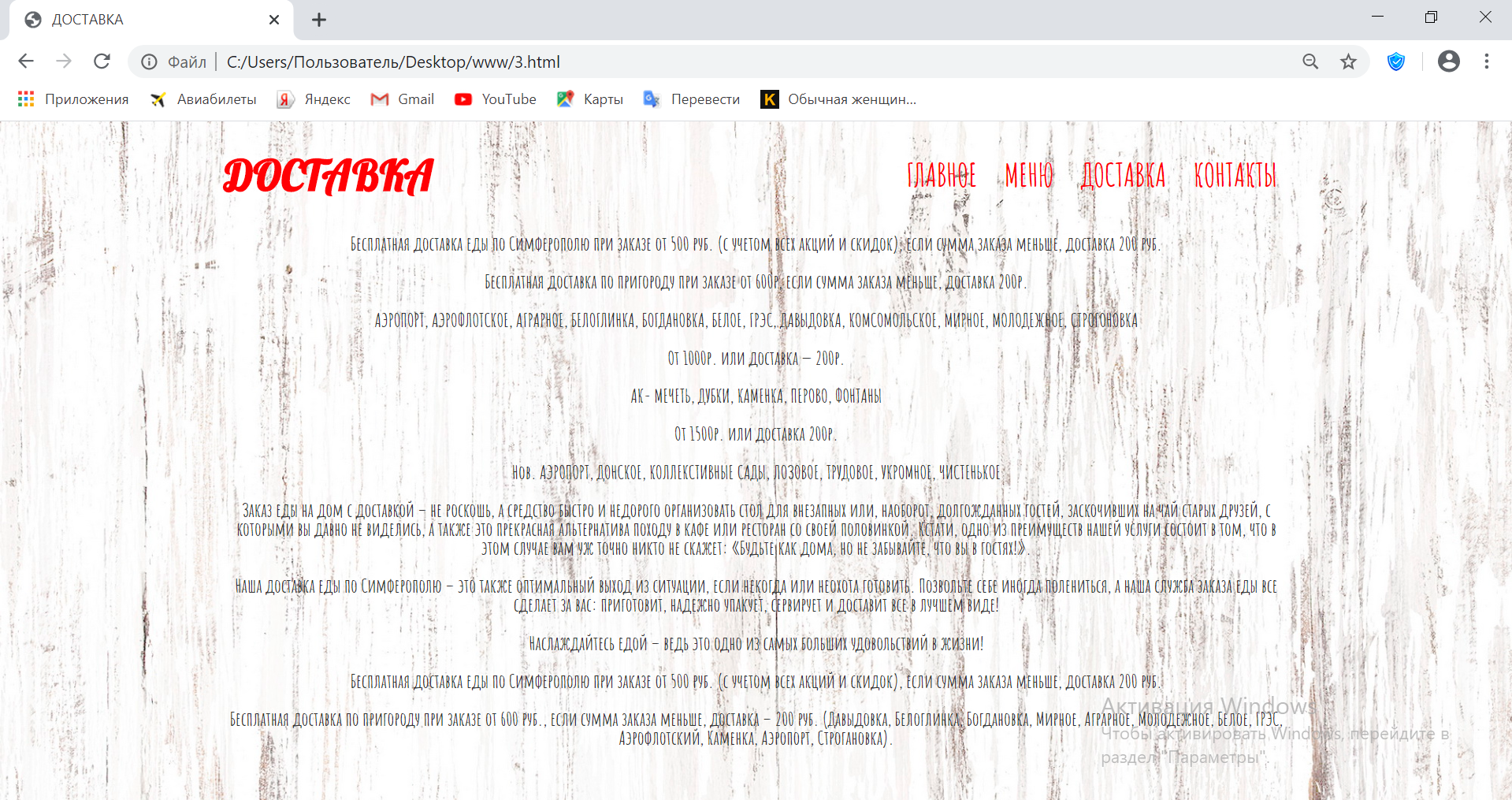


Рис. 1.3. — Третья страница сайта



Рис. 1.4. — Макет разработанного сайта

**Выводы**

В ходе данной лабароторной работы были изучены основые методы работы с каскадными таблицами стилей и селекторами. Также, ознакомились с методом блочной вёрстки сайта. В ходе данной работы был разработан сайт для ресторана, состоящий из трех страниц.