



Знакомство с HTML и CSS. Формирование веб-страниц.

Overview

Изучение HTML и CSS – это первый шаг, который необходимо сделать, если вы хотите научиться верстать сайты или работать контент-менеджером. Это относительно простые технологии, которые можно выучить самостоятельно, чтобы затем изучать более сложные темы.

В данной практической работе мы познакомимся на практике с данными языками разметки.

Что такое HTML и CSS и зачем нужно их знать?

HTML – это язык разметки, который указывает браузерам (Google Chrome, Яндекс.Браузер и другим подобным программам), где и какие элементы выводить на странице сайта. Например, где находится заголовок, основной текст, ссылки на другие страницы, меню, списки, таблицы и так далее.

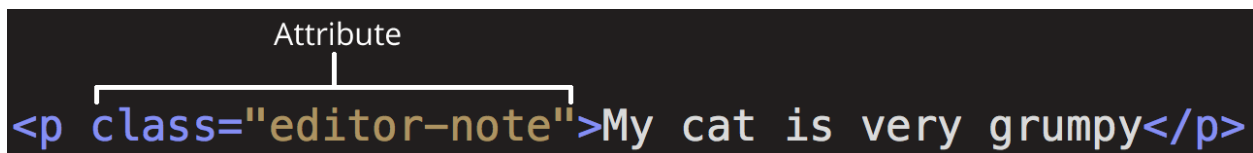
CSS – это каскадные таблицы стилей. С их помощью задают оформление различных элементов. Например, при помощи CSS можно менять цвет шрифта у текста, задавать фон страницы или отдельных элементов, красиво оформлять списки и таблицы и даже создавать интерактивные элементы (анимацию).

Из чего состоит HTML?

Язык разметки HTML состоит из тегов. Условно *теги* – это элементы, которые указывают браузеру, что должно выводиться на странице. Например, есть теги, которые обозначают вставку картинки или фотографии, видео, таблицы. Есть теги, которые обозначают начало и завершение абзаца.



Внутри тегов могут прописываться *атрибуты*, в которых указываются различные характеристики. Например, внутри тега, обозначающего ссылку, указывается атрибут с адресом страницы или сайта, куда эта ссылка ведет.





Для получения большей информации о HTML, используйте статьи на MDN.

https://developer.mozilla.org/ru/docs/Learn/HTML/Introduction_to_HTML/Getting_started

Чтобы быстрее разобраться в тегах, используйте справочники по HTML и CSS.

Например: <https://htmlbase.ru/>

Стилизация HTML-элементов

Для того, чтобы представить элементы на странице в том виде, который требуется - используются правила отображения, которые устанавливаются к элементам или группам элементов на вашей веб-страницей.

Пример правила CSS для тега <h1>:

```
h1 {  
  color: red;  
  font-size: 5em;  
}
```



Подробнее о CSS можно узнать на том же MDN.

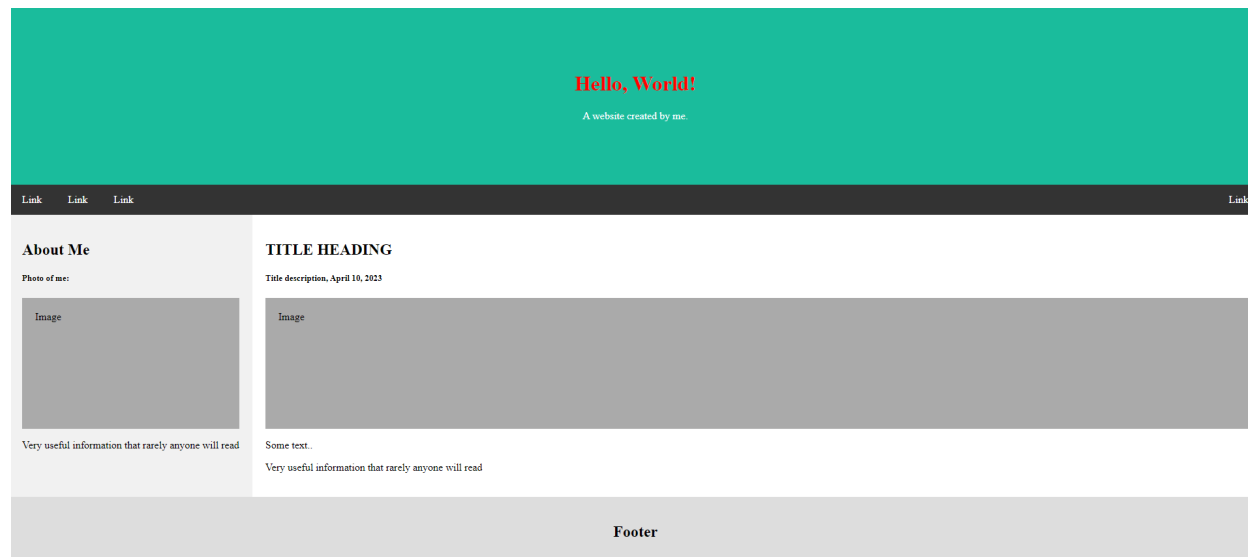
https://developer.mozilla.org/ru/docs/Learn/CSS/First_steps/How_CSS_is_structured

О всех стилях CSS можно также узнать в данном справочнике.

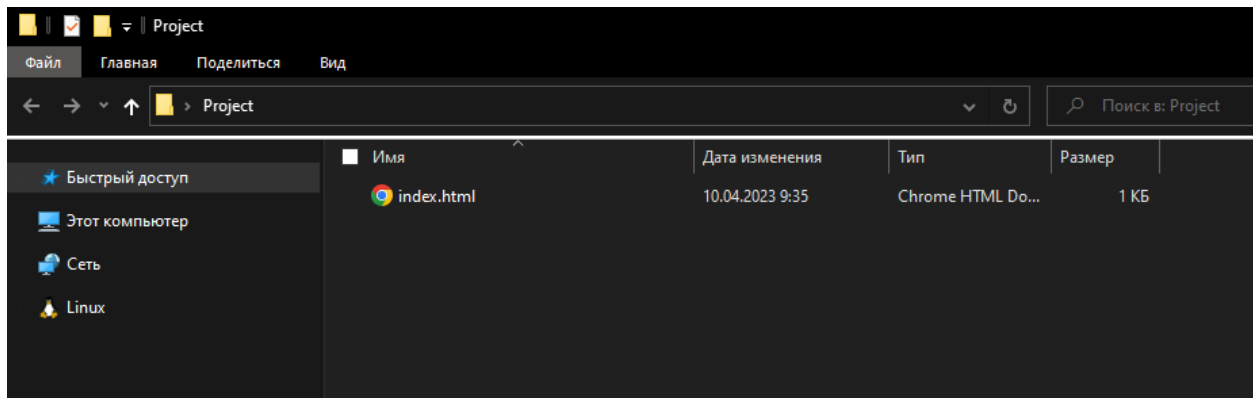
<https://htmlbase.ru/>

Построение макета страницы на HTML

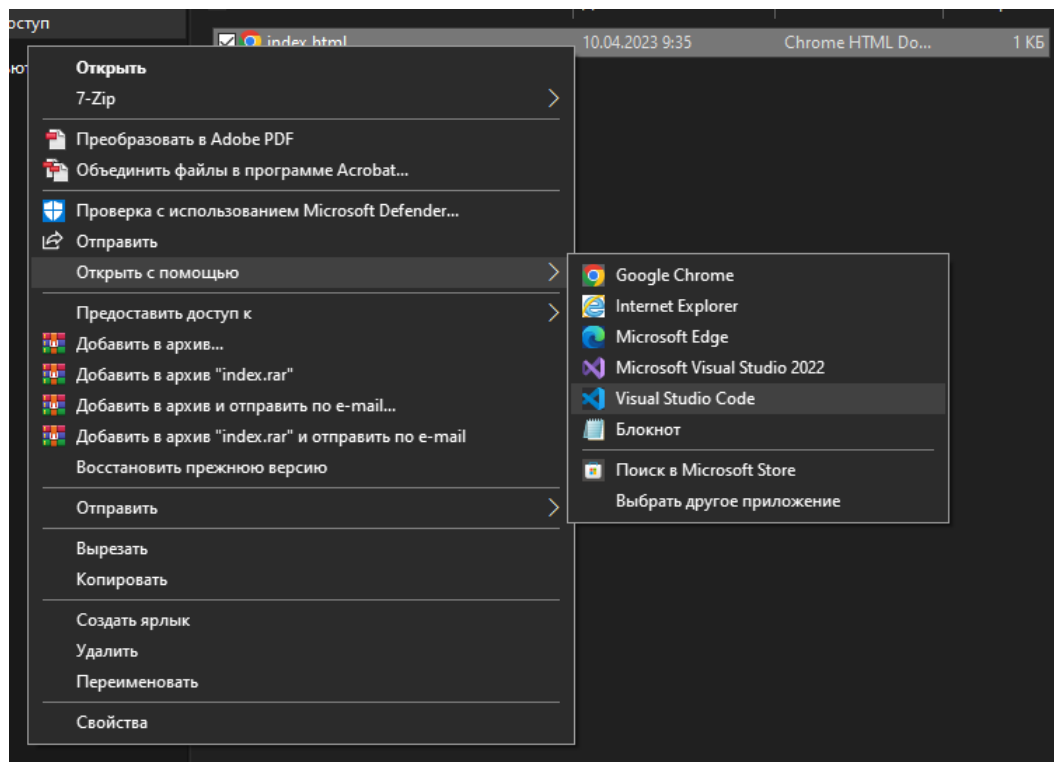
Для того, чтобы получить первый опыт построения веб-страниц, построим следующий макет страницы:



Создайте папку в которой вы будете выполнять данную работу. Внутри данной папки создайте файл с расширением .html



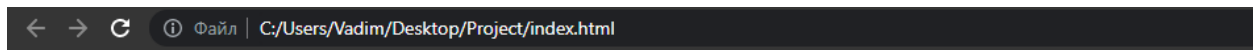
Откройте созданный файл в любом редакторе. (Даже блокнот подойдет)



Построим базовую структуру, которая есть у каждого HTML-документа. Опишите созданный вами файл следующим образом.

```
index.html X
C: > Users > Vadim > Desktop > Project > index.html > html
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <title>Title</title>
6      <meta charset="UTF-8">
7  </head>
8
9  <body>
10     <h1>Hello, World!</h1>
11 </body>
12
13 </html>
```

Чтобы посмотреть на результат - откройте данный файл при помощи браузера:



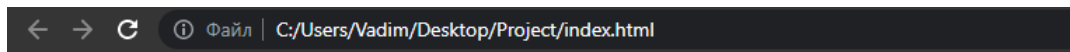
Hello, World!

На данный момент мы только описали основную структуру страницы. Для того, чтобы добавить другие элементы на страницу - добавляйте их внутри тега `<body>`

Добавьте после заголовка первого уровня следующий элемент.

```
8
9  <body>
10     <h1>Hello, World!</h1>
11     <p>A website created by me.</p>
12 </body>
13
```

Сохраните документ и обновите страницу в браузере:

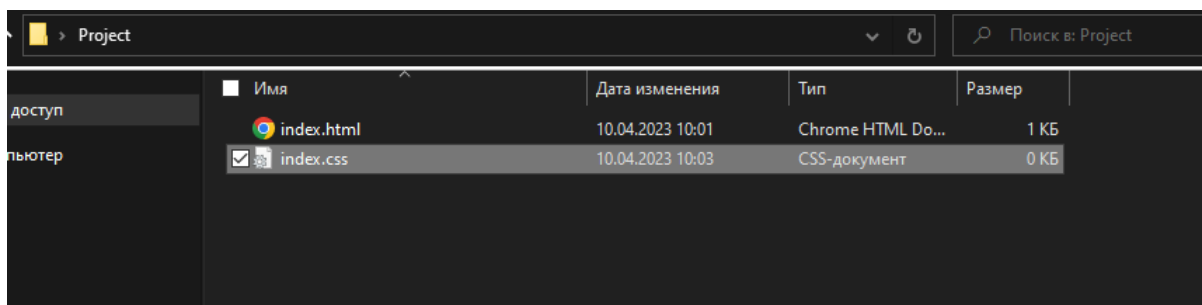


Hello, World!

A website created by me.

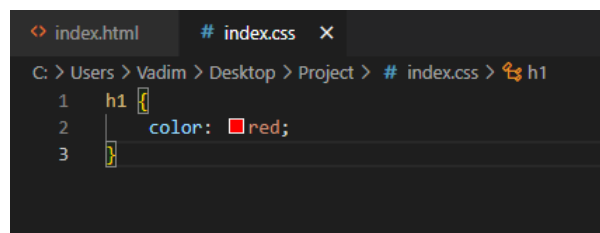
Для того, чтобы отображать элементы html по-другому (Задать: цвет, размер текста, отступы или др.) - используется таблица стилей CSS.

Создайте файл index.css рядом с index.html



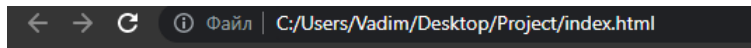
Поменяем цвет текста для заголовка первого уровня <h1>

Откройте файл index.css и добавьте стиль отображения для тега h1. (Этот стиль будет применяться для всех элементов с тегом <h1>)



Сохраните файл и обновите страницу в браузере.

Как вы могли заметить, текст не стал красным. Причина по которой цвет не изменился заключается в том, что страница html не знает про существование таблицы стилей css и поэтому не применяет описанный нами стиль.



Hello, World!

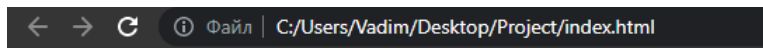
A website created by me.

Для того, чтобы связать html-страницу с таблицей стилей - нам потребуется написать внутри тега `<head>` следующую строку кода:

```
4 <head>
5   <title>Title</title>
6   <meta charset="UTF-8">
7   <link rel="stylesheet" href="index.css">
8 </head>
```

Тег `<head>` служит для конфигурации страницы. Например: тег `<title>` позволяет устанавливать отображаемый текст на вкладке вашей страницы, а тег `<meta>` с атрибутом `charset` устанавливает кодировку отображаемых символов на странице.

Сохраните документ и обновите страницу браузера, чтобы увидеть изменения. Теперь цвет текста для тега `h1` изменился на красный.



Hello, World!

A website created by me.

Давайте будем отображать данный текст в качестве шапки.

Оберните текст тегом `<div>`, чтобы сгруппировать элементы.

```
10 <body>
11   <div>
12     <h1>Hello, World!</h1>
13     <p>A website created by me.</p>
14   </div>
15 </body>
```

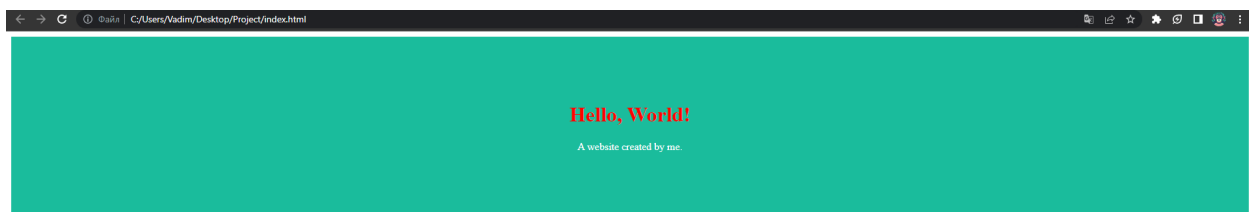
Теперь, сделаем так, чтобы тег `<div>` был в виде рамки.

```
6  div {  
7    padding: 80px;  
8    text-align: center;  
9    background: #1abc9c;  
10   color: white;  
11 }
```

Разберем следующие свойства:

- padding позволяет установить отступы внутри элементы
- text-align установить по горизонтали расположение текстовых элементов
- background определяет фон для элемента
- color определяет цвет для текста внутри данного элемента.

Сохраните изменения и посмотрите на полученный результат:

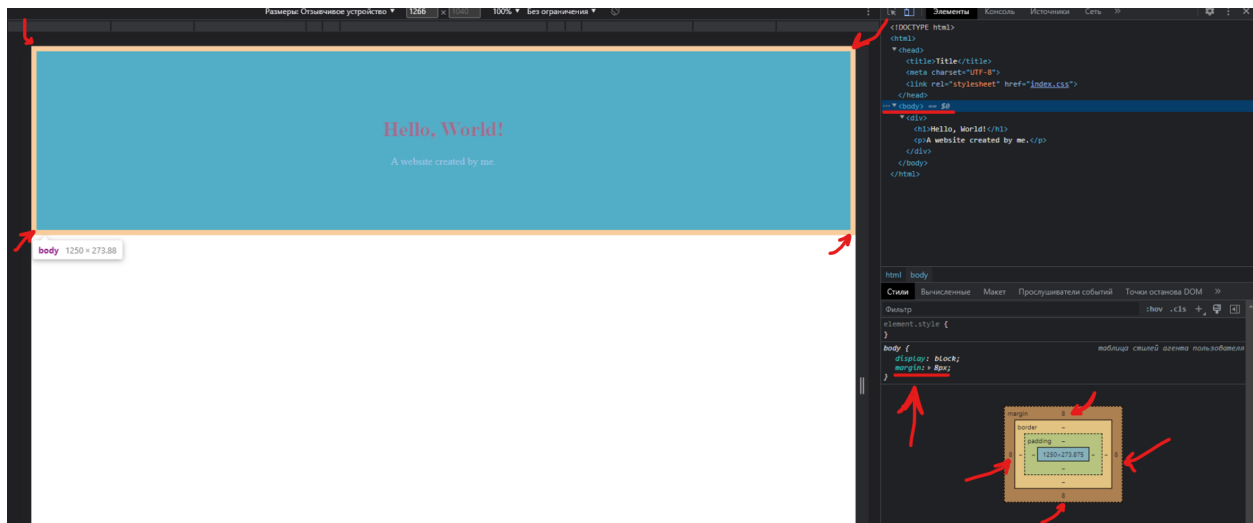


Немного про панель разработчика в браузере

Многие из вас слышали забавные истории про то, как какой-нибудь школьник менял свои отметки в журнале при помощи “Кода элемента”. Данный инструмент называется “Панель разработчика” и очень полезна во frontend-разработке.

Нам нужно выявить причину появления белых полос на краях страницы. Для этого откроем консоль разработчика (F12).

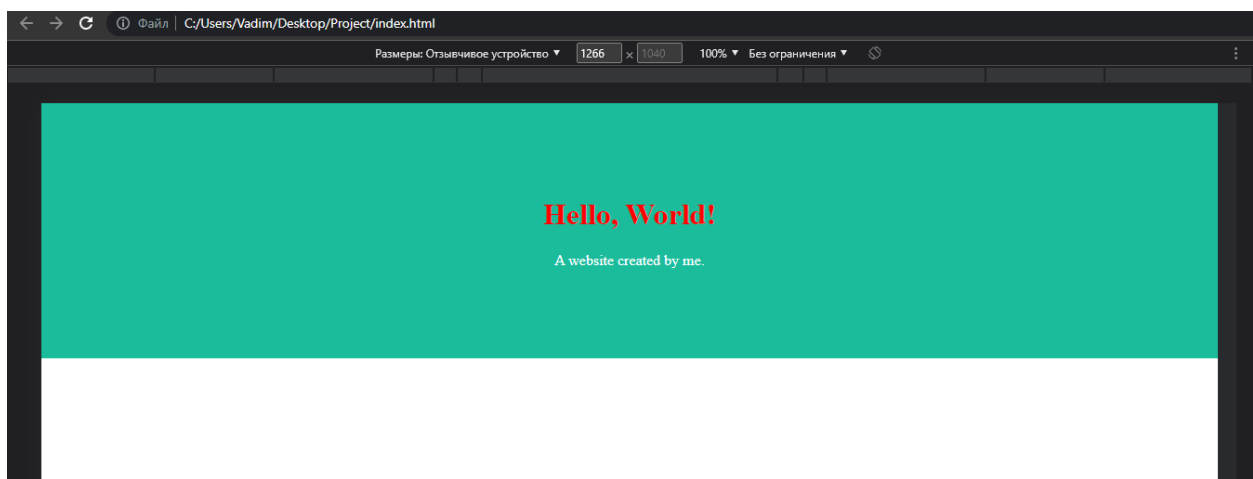
Если во вкладке Elements мы посмотрим на тег `<body>` мы можем заметить, что у данного элемента по умолчанию имеется свойство `margin`, которое по умолчанию устанавливает внешние отступы по краям. В данном случае это 8 пикселей по всем сторонам.



В таком случае перезапишем свойство margin, которое уберет отступы для тега <body>

```
1  h1 {  
2    color: red;  
3  }  
4  
5  body {  
6    margin: 0;  
7  }  
8  
9  div {  
10   padding: 80px;  
11   text-align: center;  
12   background: #1abc9c;  
13   color: white;  
14 }
```

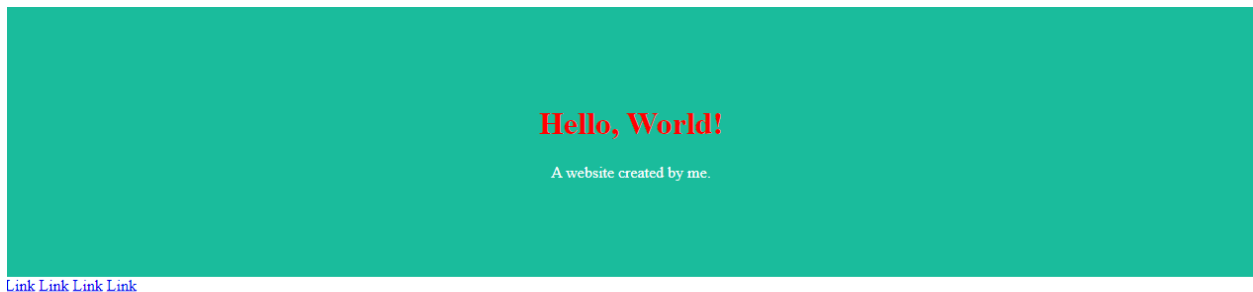
Теперь “шапка” страницы выглядит должным образом.



Теперь добавим маленькую панель для навигации по страницам. Для того, чтобы обеспечивать переход по ссылке - используется тег `<a>`. Чтобы указывать ресурс на который будет осуществляться переход - используется атрибут `href`.

```
10 <body>
11   <div>
12     <h1>Hello, World!</h1>
13     <p>A website created by me.</p>
14   </div>
15
16   <a href="#">Link</a>
17   <a href="#">Link</a>
18   <a href="#">Link</a>
19   <a href="#">Link</a>
20
21 </body>
22
```

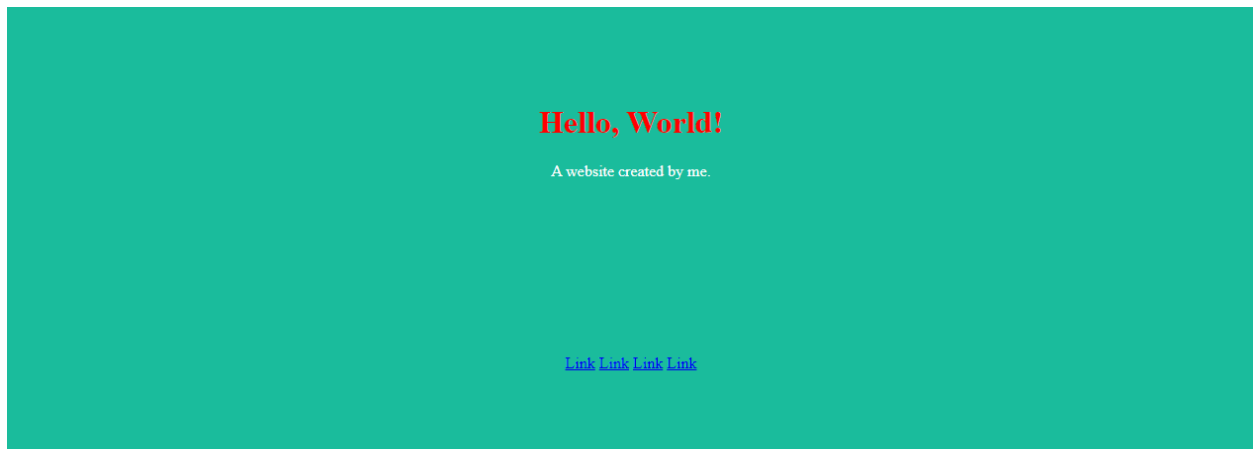
После добавления появились синие ссылки для перехода.



Чтобы сделать панель навигации - объединим их внутри блока `<div>`

```
10 <body>
11   <div>
12     <h1>Hello, World!</h1>
13     <p>A website created by me.</p>
14   </div>
15
16   <div>
17     <a href="#">Link</a>
18     <a href="#">Link</a>
19     <a href="#">Link</a>
20     <a href="#">Link</a>
21   </div>
22 </body>
```

В результате данного действия, страница будет выглядеть следующим образом:



Связано это с тем, что мы определили стиль для тега `<div>` и теперь все элементы с данным тегом будут отображаться с таким фоном и располагать элементы по центру.

Однако мы хотим, чтобы панель навигации выглядела по другому. Для этого вернемся в таблицу стилей и преобразуем правила для отображения.

Чтобы исправить данную проблему, создадим классы в CSS и используем описанные в них свойства только для тех элементов, которым потребуется.

Замените `div` на `.header` чтобы создать класс, который будет описывать шапку нашей страницы

```
9  .header {
10    padding: 80px;
11    text-align: center;
12    background: #1abc9c;
13    color: white;
14 }
```

Обновите страницу и посмотрите на результат

Hello, World!

A website created by me.

[Link Link Link Link](#)

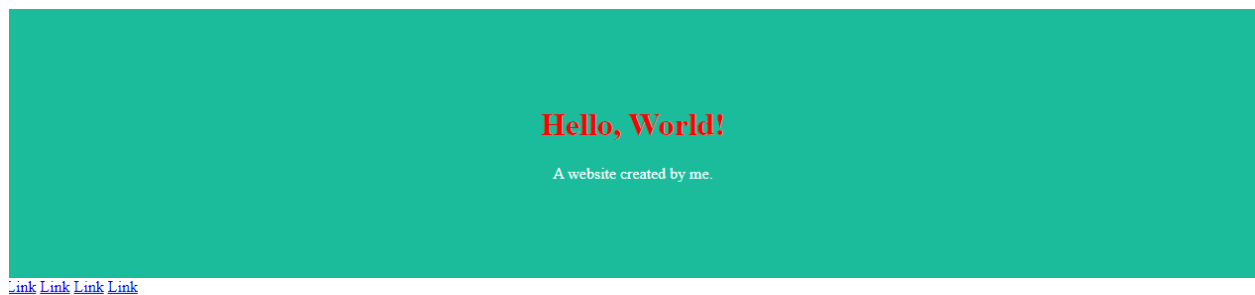
Для того, чтобы шапка вернула свои правила для отображения - добавьте атрибут `class` в котором вы укажете название применяемого стиля.

```

10 <body>
11   <div class="header">
12     <h1>Hello, World!</h1>
13     <p>A website created by me.</p>
14   </div>
15
16   <div>
17     <a href="#">Link</a>
18     <a href="#">Link</a>
19     <a href="#">Link</a>
20     <a href="#">Link</a>
21   </div>
22 </body>
23

```

Теперь мы можем создать отдельный стиль для отображения навигационной панели



Давайте сделаем панель для ссылок темно-серого цвета.

```

16 .navbar {
17   background-color: #333;
18 }
19

```

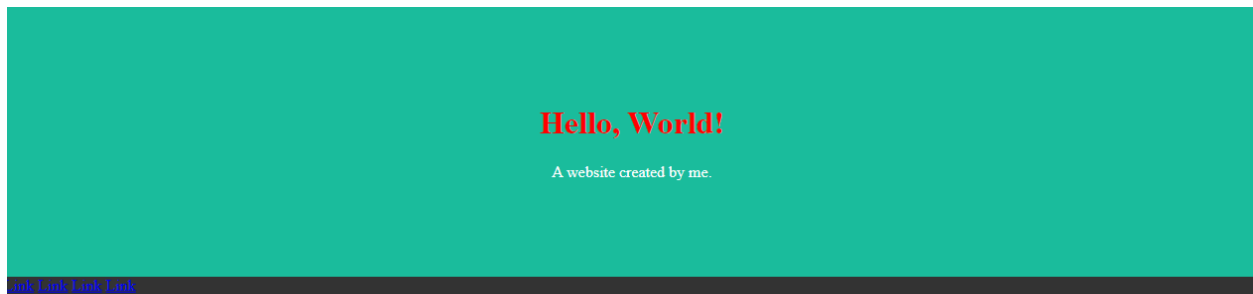
И применим стиль для группы ссылок.

```

<div class="navbar">
  <a href="#">Link</a>
  <a href="#">Link</a>
  <a href="#">Link</a>
  <a href="#">Link</a>
</div>

```

В данный момент ссылки имеют не совсем подходящий цвет. Поэтому потребуется изменить стиль для ссылок внутри навигационной панели.



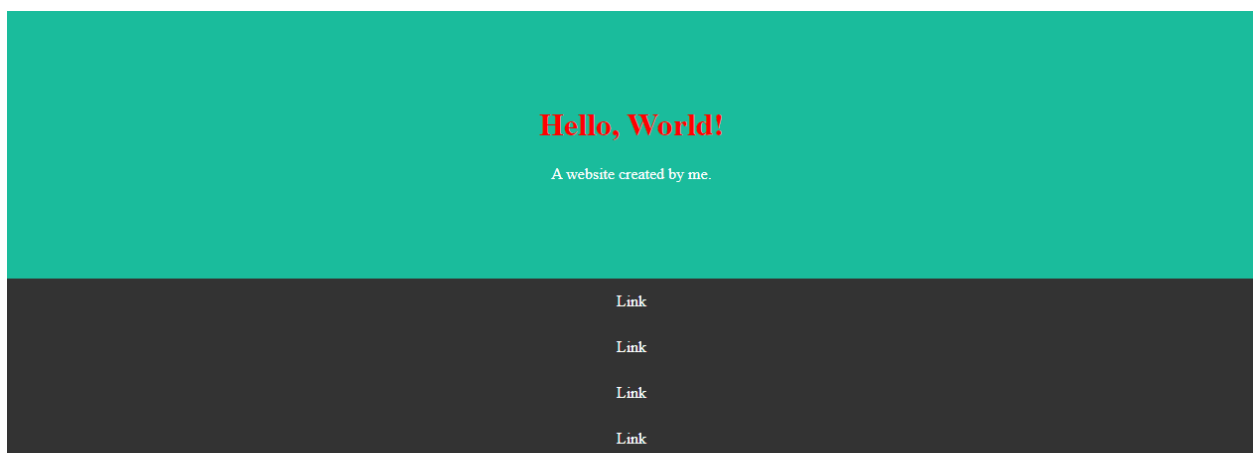
Добавим следующее правило, которое будет затрагивать все элементы с тегом `<a>`, расположенные внутри элемента с классом `navbar`.

```
.navbar a {  
  display: block;  
  color: white;  
  text-align: center;  
  padding: 14px 20px;  
  text-decoration: none;  
}
```

Разберем новые свойства, которые были применены для ссылок:

- `display` определяет как будет отображаться элемент. В нашем случае ссылкам был установлен блочный тип. Таким образом каждый элемент начинается с новой строки и занимает 100% ширины по умолчанию.
- `text-decoration` задает подчеркивание элементу. Для тега `<a>` он по умолчанию включен, в данном случае он имеет свойство `none` (выключен)

В результате, панель навигации преобразилась в следующий вид:

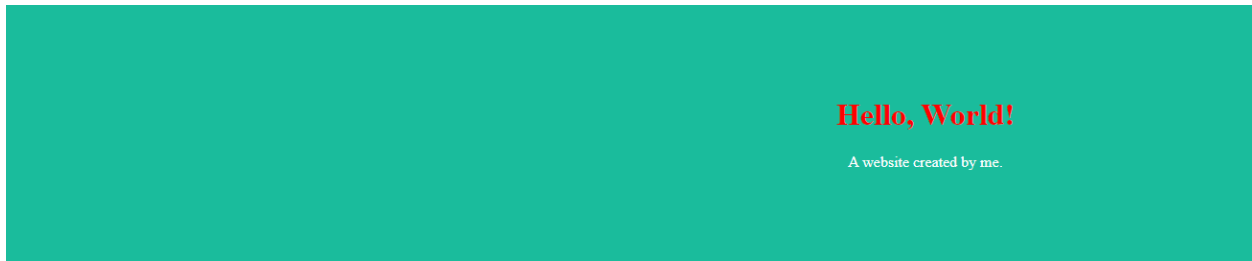


Для того, чтобы ссылки в панели навигации располагались горизонтально, подправим стили.

Добавим свойство `float` для ссылок, чтобы текст обтекал по левой части

```
.navbar a {  
  float: left;  
  display: block;  
  color: white;  
  text-align: center;  
  padding: 14px 20px;  
  text-decoration: none;  
}
```

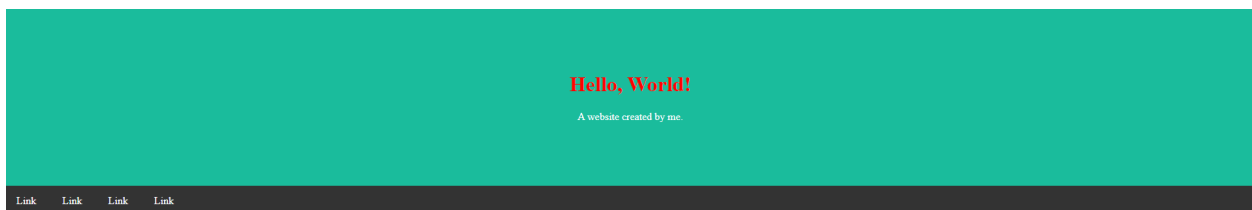
Если оставить это как есть - то панель навигации просто пропадёт



Чтобы исправить это, добавим свойство `overflow` для класса `navbar`

```
.navbar {  
  overflow: hidden;  
  background-color: #333;  
}
```

Панель навигации готова



Углубление в CSS

Разберем также некоторые особенности CSS, которые вы можете использовать при разработке.

Возможность применять несколько стилей

Первой из них является возможность применять сразу несколько стилей для одного элемента, то есть накладывать стили друг на друга.

В качестве примера попробуем сделать так, чтобы 4-я ссылка в навигационной панели была расположена в справа.

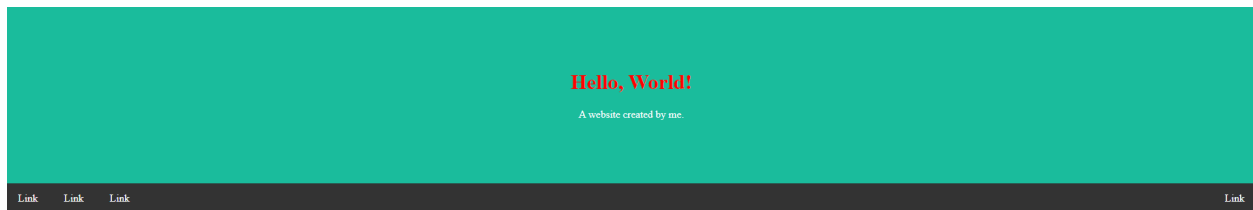
Для этого создадим правило, которое будет работать только для элементов внутри класса navbar и которые являются ссылкой (тегом <a>). Установим свойство обтекания текста по правой части:

```
29
30 .navbar a.right {
31     float: right;
32 }
```

И наделим данным свойством последнюю ссылку в панели навигации

```
<div class="navbar">
  <a href="#">Link</a>
  <a href="#">Link</a>
  <a href="#">Link</a>
  <a class="right" href="#">Link</a>
</div>
```

Теперь последняя ссылка расположена справа и, если потребуется, можно добавить данное свойство и другим ссылкам:



Использование псевдоклассов

Рассмотрим следующую особенность, а именно: задавать псевдоклассы для различных элементов.

Давайте сделаем так, чтобы при наведении на ссылку - элемент изменял свой фон. Таким образом пользователю будет понятно, что он навелся на элемент который можно будет нажать.

Чтобы сделать это, добавим псевдокласс **:hover** для всех ссылок (тега <a>), которые находятся на навигационной панели navbar.

```
.navbar a:hover {
    background-color: #ddd;
    color: black;
}
```

Таким образом, при наведении на элемент (ссылку) меняются свойства на те, которые были определены.



Добавление содержимого на странице

Давайте добавим в качестве контента на странице что-то в виде “блока новостей”.

Блок новостей имеет какую-нибудь общую структуру, на котором имеется следующее:

- Заголовок
- Информация о новости (Категория, дата)
- Картинка
- Основной текст
- Дополнительный текст

Давайте сформируем данную структуру на html и расположим его под навигационной панелью:

```
<body>
  <div class="header">
    <h1>Hello, World!</h1>
    <p>A website created by me.</p>
  </div>

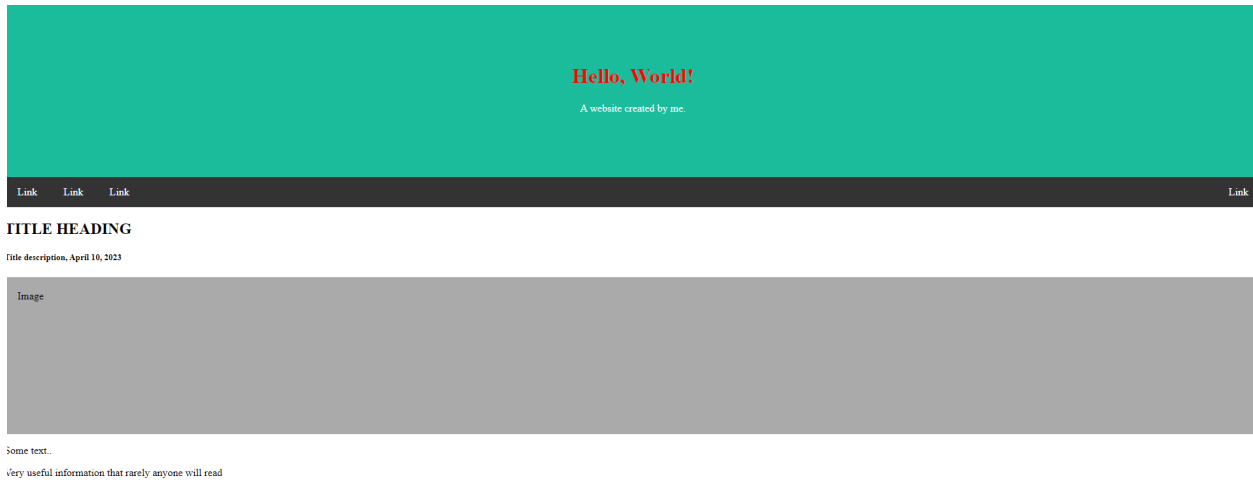
  <div class="navbar">
    <a href="#">Link</a>
    <a href="#">Link</a>
    <a href="#">Link</a>
    <a class="right" href="#">Link</a>
  </div>

  <div>
    <h2>TITLE HEADING</h2>
    <h5>Title description, April 10, 2023</h5>
    <div class="fakeimg" style="height:200px;">Image</div>
    <p>Some text..</p>
    <p>Very useful information that rarely anyone will read</p>
  </div>
</body>
```

И в качестве “заглушки” для изображений добавим следующий стиль:

```
.fakeimg {
  background-color: #aaa;
  width: 100%;
  padding: 20px;
}
```

Теперь на странице данная структура имеет следующий вид:



Представим, что заказчик захотел размещать в левой части какой-нибудь дополнительный контент, например информацию из “About Me”.

Данный контент будет иметь следующую структуру:

- Заголовок
- Подпись
- Основная картинка
- Основной текст

Опишем структуру на html


```

<div class="navbar">
  <a href="#">Link</a>
  <a href="#">Link</a>
  <a href="#">Link</a>
  <a class="right" href="#">Link</a>
</div>

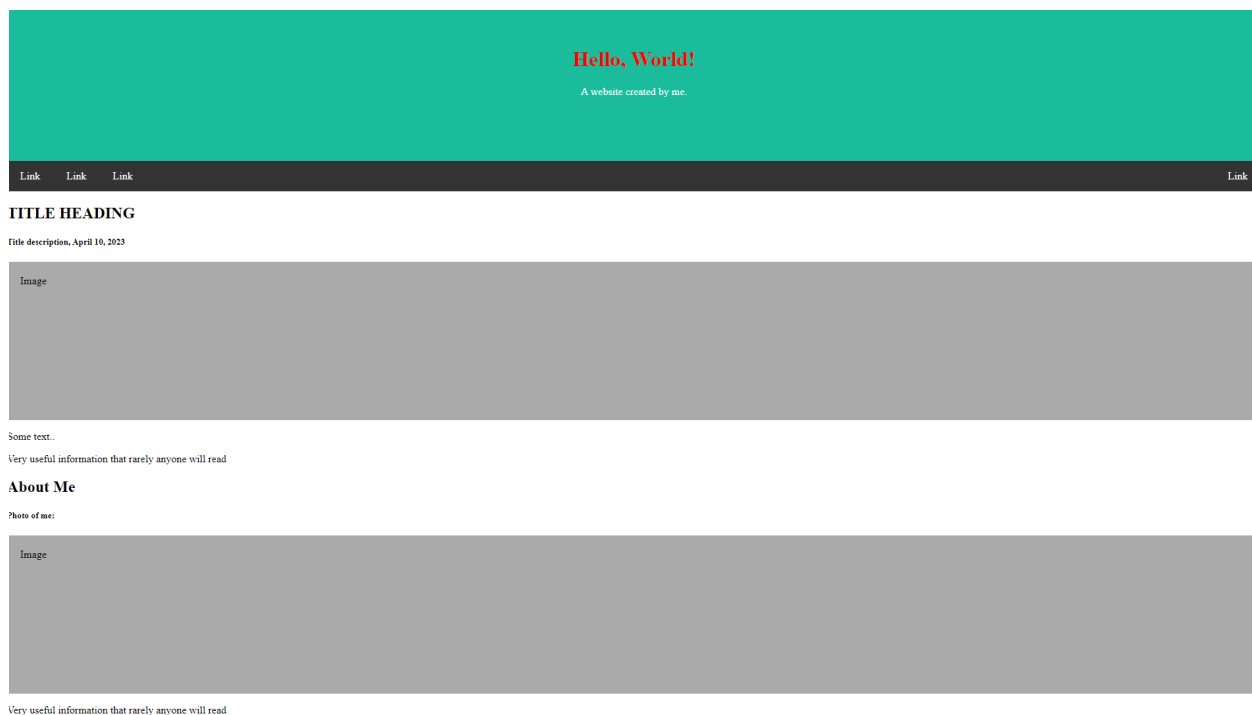
<div>
  <h2>TITLE HEADING</h2>
  <h5>Title description, April 10, 2023</h5>
  <div class="fakeimg" style="height:200px;">Image</div>
  <p>Some text..</p>
  <p>Very useful information that rarely anyone will read</p>
</div>

<div>
  <h2>About Me</h2>
  <h5>Photo of me:</h5>
  <div class="fakeimg" style="height:200px;">Image</div>
  <p>Very useful information that rarely anyone will read</p>
</div>

</body>

```

Если расположить данный блок после основного контента - получится следующее:



Чтобы расположить данные блоки в ряд, сделаем следующие действия:

1. Заранее присвоим блокам подходящие для них классы (main и side), а также обернём данные блоки в ещё один `<div>` и данному элементу присвоим класс `row`.

```

<div class="row">
  <div class="main">
    <h2>TITLE HEADING</h2>
    <h5>Title description, April 10, 2023</h5>
    <div class="fakeimg" style="height:200px;">Image</div>
    <p>Some text..</p>
    <p>Very useful information that rarely anyone will read</p>
  </div>

  <div class="side">
    <h2>About Me</h2>
    <h5>Photo of me:</h5>
    <div class="fakeimg" style="height:200px;">Image</div>
    <p>Very useful information that rarely anyone will read</p>
  </div>
</div>

```

- Опишем следующие правила для классов row, side и main. Сделаем так, чтобы все дочерние элементы располагались внутри элемента с классом row, также, чтобы элемент класса main имел ширину как 70% от ширины окна.

```

.row {
  display: flex;
}

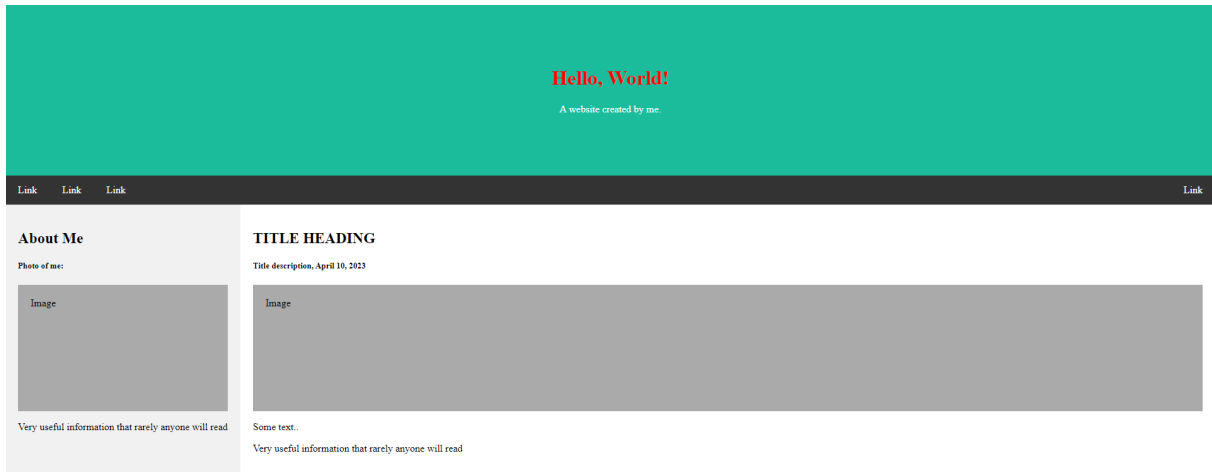
.side {
  background-color: #f1f1f1;
  padding: 20px;
}

.main {
  flex: 70%;
  background-color: white;
  padding: 20px;
}

* {
  box-sizing: border-box;
}

```

- В результате страница имеет следующий вид



Добавим нижний колонтитул для данной страницы, чтобы размещать внутри него информацию о создателе и ссылках.

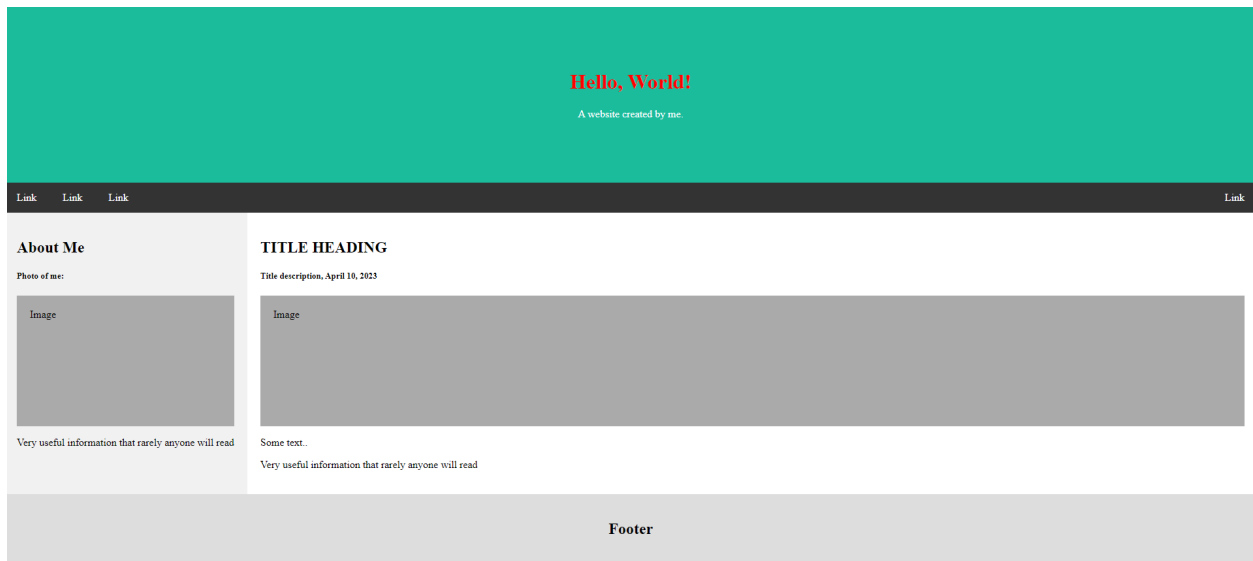
Добавьте ниже заголовок контента, который обернут внутри `<div>` с классом `footer`.

```
<div class="footer">
  <h2>Footer</h2>
</div>
```

И определите для нижнего колонтитула следующие свойства

```
.footer {
  padding: 20px;
  text-align: center;
  background: #ddd;
}
```

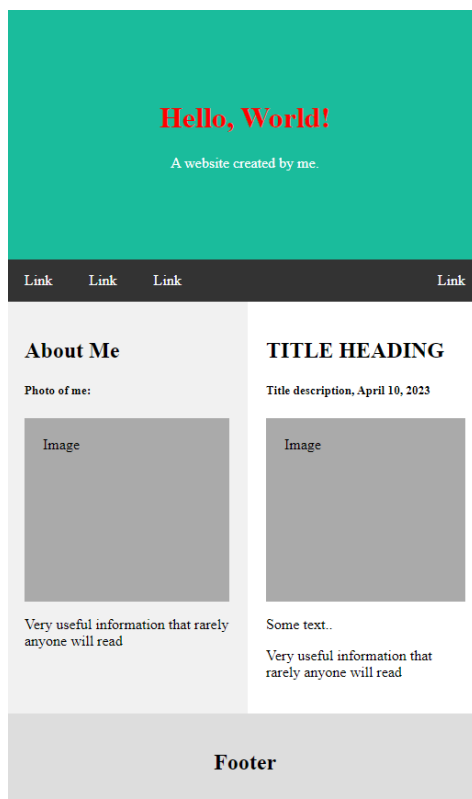
Теперь на странице расположен нижний колонтитул на котором можно будет разместить дополнительную информацию.



Адаптивная верстка. Отображение страницы при минимальной ширине экрана.

Попробуйте изменить ширину окна браузера до самого минимума.

В таком случае, страница будет выглядеть следующим образом:



Пользователям смартфонов придётся просматривать очень мелкую информацию на странице, чтобы адаптировать страницу под такие устройства можно воспользоваться **медиавыражениями**:

```
@media screen and (max-width: 700px) {  
  .row {  
    flex-direction: column;  
  }  
}  
  
@media screen and (max-width: 400px) {  
  .navbar a {  
    float: none;  
    width: 100%;  
  }  
}
```

Таким образом:

- В случае, если ширина экрана будет меньше 700px, то элементы внутри класса row будут отображать вертикально.
- В случае, если ширина экрана будет меньше 400px, то элементы внутри класса navbar будут отображаться вертикально.

Подробнее про медиавыражения можно прочитать тут:

https://developer.mozilla.org/ru/docs/Web/CSS/Media_Queries/Using_media_queries#медиа_для_разных_типов_устройств

Задание

1. Преобразуйте данный макет страницы в страницу интернет-магазина.
2. Создайте 3 файла html, на которые будет производиться переход при помощи навигационной панели. (Наполнять страницы не нужно. Добавьте текст который будет сообщать на какой странице вы находитесь)
3. Загрузите работу в репозиторий и разместите в папке docs
4. Сделайте так, чтобы страница была доступна для просмотра всем. Для этого воспользуйтесь GitHub Pages. (или руководством ниже)

Публикация и доступ по домену

Перейдите в настройки репозитория и откройте раздел Pages

General

Access

Collaborators

Moderation options

Code and automation

Branches

Tags

Actions

Webhooks

Environments

Codespaces

Pages

Security

Code security and analysis

Deploy keys

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Build and deployment

Source

Deploy from a branch

Branch

Your GitHub Pages site is currently being built from the /docs folder in the main branch. [Learn more.](#)

main /docs Save

Learn how to add a custom domain

Custom domain

Custom domains allow you to serve your site from a domain other than dm-larionov.github.io. [Learn more](#)

Save Remove

Перейдите к странице вашего репозитория и дождитесь галочки. На вашей странице также должен быть раздел github-pages.



Обратите внимание, папка docs расположена в корне репозитория!

main 1 branch 0 tags

Go to file Add file <> Code

torvalds Миграция в папку docs

7c1a9b9 3 minutes ago 3 commits

docs	Миграция в папку docs	3 minutes ago
.gitattributes	Initial commit	24 minutes ago
README.md	Initial commit	24 minutes ago

README.md

My-site

Мой самый лучший сайт на всей планете

About

Мой самый лучший сайт на всей планете

Readme

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Contributors



torvalds Linus Torvalds

Environments 1

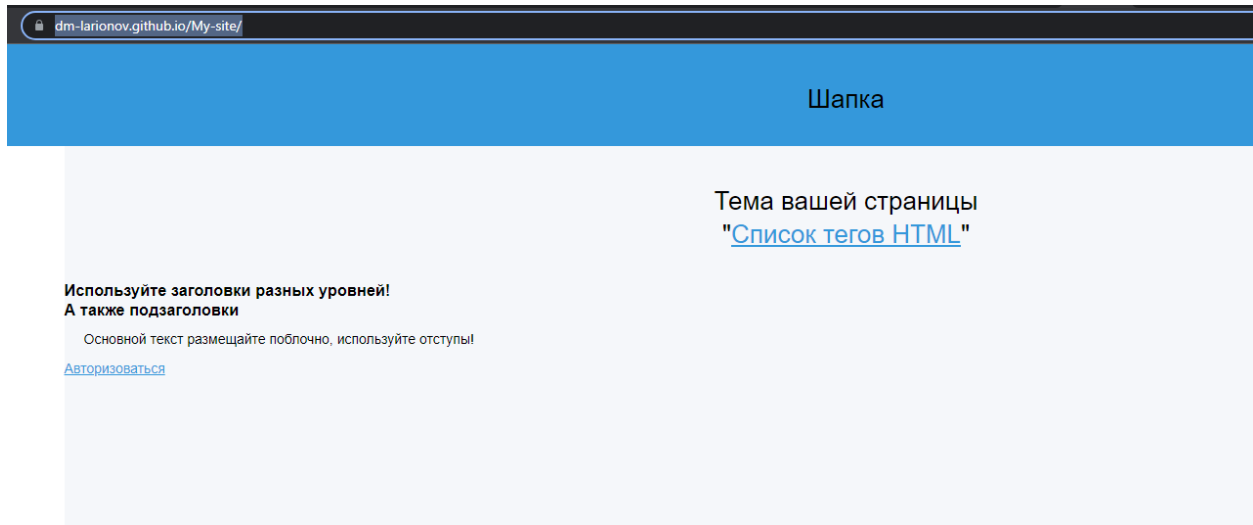
github-pages Active

Перейдите в github-pages и нажмите на View Deployment

Deployments / History Show: github-pages ▾

 github-pages at 7c1a9b9 Deployed by GitHub Pages on behalf of  dm-larionov 4 minutes ago Active View deployment

Ваш сайт расположен по данному URL-адресу.



The screenshot shows a web browser with the address bar displaying `dm-larionov.github.io/My-site/`. The page has a blue header with the word "Шапка" (Header) in the center. Below the header, the main content area is light gray and contains the text "Тема вашей страницы" (Theme of your page) followed by "[Список тегов HTML](#)". On the left side of the main area, there is a section titled "Используйте заголовки разных уровней! А также подзаголовки" (Use headings of different levels! And also subtitles) with a subtext "Основной текст размещайте поблочно, используйте отступы!" (Place the main text block by block, use indentation!). At the bottom left of this section is a link "Авторизоваться" (Log in).