**Министерство образования и науки РФ**

[**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**](http://www.nngasu.ru/)

[**«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ АРХИТЕКТУРНО-СТРОИТЕЛЬНЫЙ УНИВЕРСИТЕТ»**](http://www.nngasu.ru/)

**Курсовая работа**

Разработка Web-сайта спортклуба «River»

**Студент:** Фадин И.И. ПРИ19.19/1

**Научный руководитель:** Родионова С.В.

**2021год.**

Оглавление

[1 Введение 3](#_Toc90423934)

[2 Определение требований к сайту 4](#_Toc90423935)

[3 Техническое задание на разработку сайта 5](#_Toc90423936)

[4 Разработка эскизов дизайна 6](#_Toc90423937)

[5 Верстка, программирование 11](#_Toc90423938)

[6 JavaScript 22](#_Toc90423939)

[7 Вывод 27](#_Toc90423940)

[8 Список источников 28](#_Toc90423941)

# Введение

В последние годы можно наблюдать значительные изменения в сети интернет. Сайты, которые ранее являлись платформой для расположения статического контента, теперь стали многофункциональными, интерактивными системами для предоставления различной информации.

Актуальность темы исследования обусловлена необходимостью анализа существующих методов разработки корпоративных и информационных ресурсов для выявления наиболее эффективного метода их создания. Целью данного исследования является определение эффективных методов разработки для грамотного построения корпоративных и информационных Интернет-ресурсов. Под методами создания интернет-сайтов понимается совокупность приемов и инструментов разработки.

Сторонники здорового образа жизни знают о пользе фитнеса, но не понимают, какой зал или программу занятий выбрать. Чтобы лучше в этом разобраться, потенциальные клиенты ищут информацию онлайн, сравнивают сайты различных фитнес центров.

Чтобы создать конверсионный сайт фитнес клуба важно учесть его функциональность, полезность. Нужна удобная структура, набор посадочных страниц, интеграция дополнительных сервисов.

Цель: научиться создавать Web – сайт с помощью программных средств таких как HTML, CSS, JavaScript.

# Определение требований к сайту

В зависимости от бюджета и основных задач, дизайн спортивного сайта может быть максимально простым, статичным или ярким, интерактивным. Главное правило — создать единый визуальный стиль с социальными сетями компании, общим оформлением тренажёрных залов. Например, если салон фитнес-клуба темно-фиолетовый, то дизайн сайта должен этому соответствовать.

Компания «River» занимается оказанием услуг в сфере спорта и фитнеса. Имеет собственный спортивный комплекс (СК) в котором находятся различные площади для проведения занятий спортом.

Основные требования к сайту:

* Наличие логотипа клуба
* Дизайн должен соответствовать стилистике клуба
* Галерея фотографий различных зон СК
* Основная информация (адрес, контактный телефон, часы работы и т.п.)
* Возможность узнать стоимость абонементов, а также стоимость аренды
* Наглядно видеть актуальные акции и новости СК
* Основной аудиторией являются дети (посещение детских секций, а также взрослые от 35 лет)

# Техническое задание на разработку сайта

В первую очередь сайт должен содержать главную страницу, на ней должны присутствовать элементы:

* Шапка сайта (логотип, контактный телефон, адрес СК, меню с пятью вкладками) она должна быть на всех вкладках.
* Содержание меню: (Главная, Абонементы, Галерея, О нас, Контакты)
* Отсчет времени, до конца актуальной акции
* Галерея фотографий основных зон СК
* Часть с контентом (лента постов, слева колонка с недавними постами и дополнительной информацией)
* Снизу страницы виджеты, с информацией о часах работы, адресе и контактных данных.

На странице абонементы должны быть виджеты для получения информации о ценах аренды и т.д.

На странице Галерея, иллюстрируются, внутренние помещения, а также различные фотоотчеты с соревнований и пр.

Страница О нас содержит основную информацию СК

На странице контакты пользователь может найти контактные данные клуба (номер телефона, адрес, электронная почта и пр.)

# Разработка эскизов дизайна

Для разработки сайта, понадобится создать макет, и далее несколько элементов дизайна.

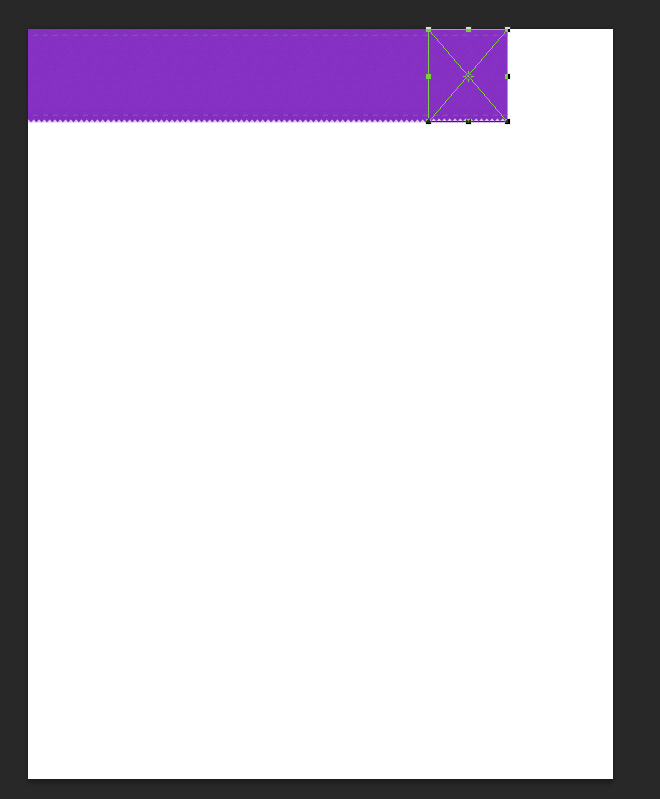
Наши ключевые цвета это темно фиолетовые оттенки.

#490a7b #8731c5

Для создания макета мы будем использовать программу Adobe Photoshop cs 6.

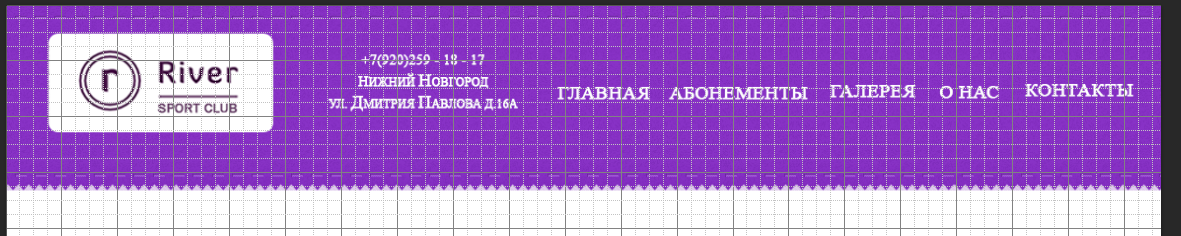
Сначала создадим header (вверх нашего сайта)



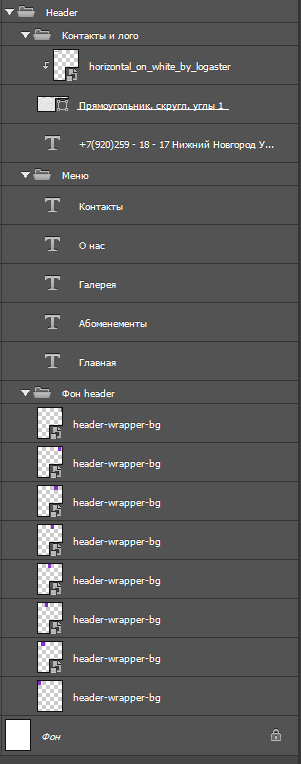
Следующий шаг будет добавление логотипа, и названия клуба.

 шрифт Franklin Gothic book

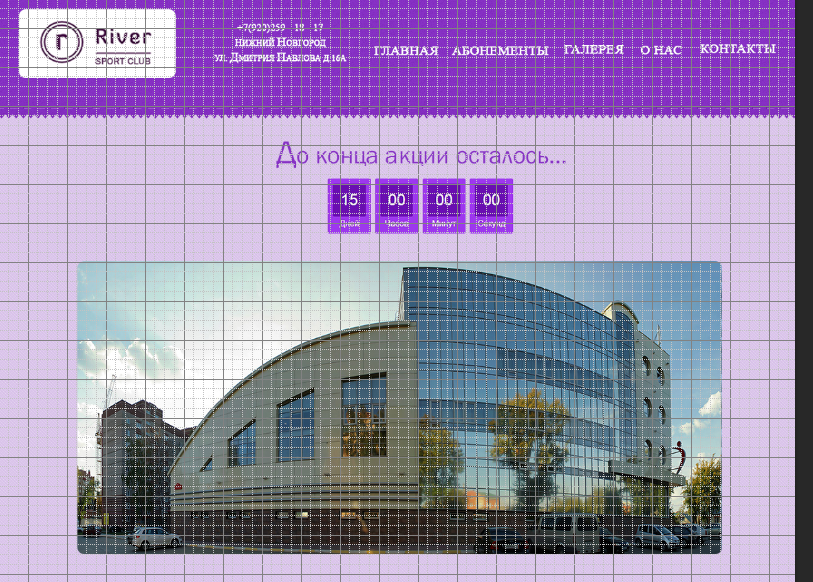
Далее добавим контактные данные, а также кнопки меню



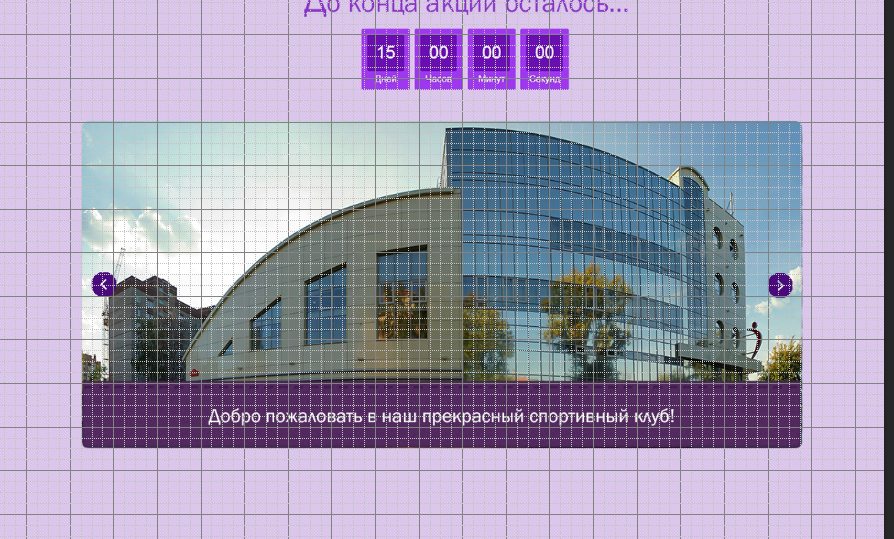
Шрифт по стандарту Times New Roman

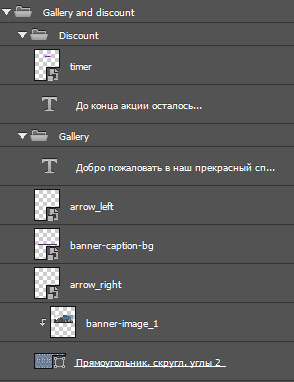
 Расположение слоев в макете

Далее создадим фон для таймера и галереи и добавим туда их.



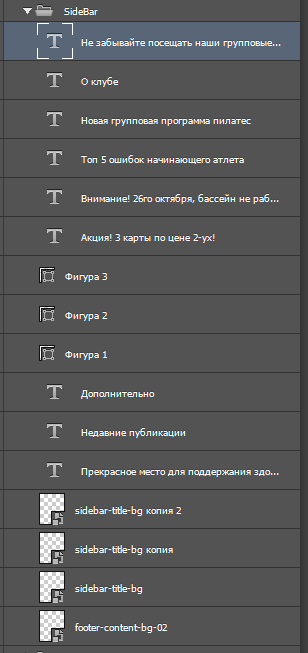
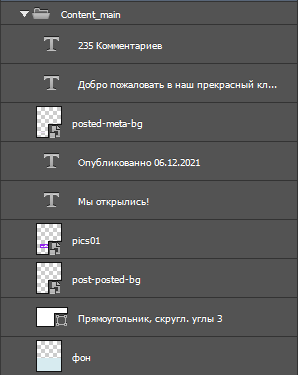
Также нужны стрелки для перелистывания и строка приветствия.





Следующим шагом будет создание части контента (недавние посты и колонка с информацией)





Аналогично создадим нижнюю строку виджетов

Итоговый макет выглядит следующим образом



# Верстка, программирование

Следующий этап создания сайта, это верстка по заранее созданному макету

Первая часть — это создание шапки (header)

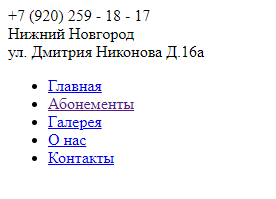
В нем мы используем следующие теги:

* **Header** - представляет собой вводный контент, обычно группу вводных или навигационных средств. Он может содержать другие элементы-заголовки, а также логотип, форму поиска, имя автора и другие элементы.
* **Div** - является блочным элементом и предназначен для выделения фрагмента документа с целью изменения вида содержимого. Как правило, вид блока управляется с помощью стилей. Чтобы не описывать каждый раз стиль внутри тега, можно выделить стиль во внешнюю таблицу стилей, а для тега добавить атрибут class или id с именем селектора.
* **Br** - указывает место переноса строки в тексте HTML страницы. Тег br ставится перед местом конца строки, текст после тега будет выведен с новой строки.
* **Ul** - используется для создания маркированного (неупорядоченного) списка
* **Li** - Каждый элемент в маркированном списке должен быть вложен в тег [<li>](https://ru.w3docs.com/uchebnik-html/html-teg-li.html)

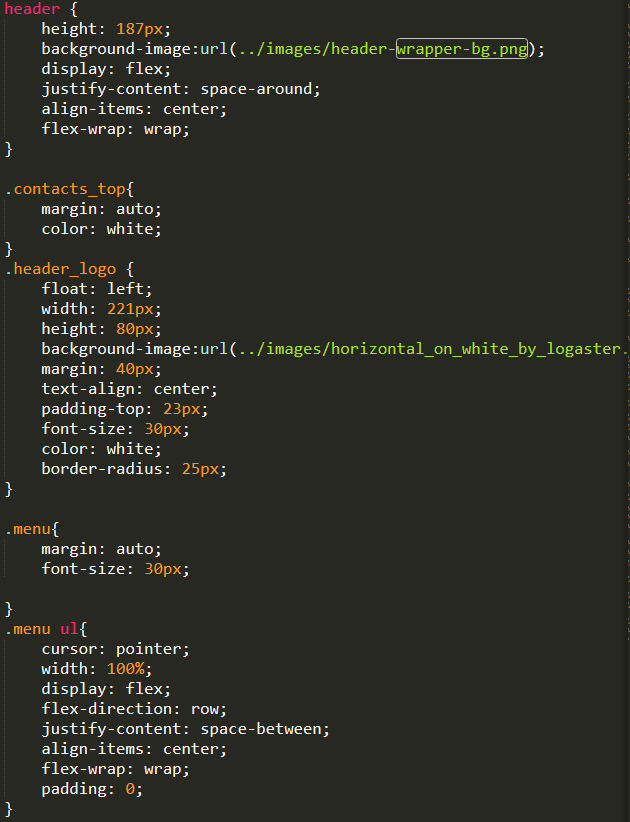
Далее представлен код на языке разметки html.



Если открыть данный файл через браузер, то мы увидим следующую картину.



Для придания стиля разметке, мы будем использовать CSS.



Присваивая классы для элементов, мы задаем этим самым классам определенные параметры, например, такие как color или margin.

Для нужного нам расположения элементов на экране мы будем использовать **display : flex**;

Это новая технология, которая уже имеет достаточно широкую поддержку браузеров. Flexbox предоставляет инструменты для быстрого создания сложных, гибких макетов, и функции, которые были сложны в традиционных методах CSS.

Когда элементы выложены как **flex блоки**, они располагаются вдоль двух осей:

**Главная ось** (main axis) проходит в том направлении, вдоль которого расположены Flex элементы (например, в строку слева направо или вдоль колонок вниз.) Начало и конец этой оси называются **main start и main end**.

**Поперечная ось** (cross axis) проходит перпендикулярно Flex элементам. Начало и конец этой оси называются **cross start and cross end**.

Родительский элемент, на который назначено свойство **display: flex**

Элементы, размещённые в нём как Flex блоки называются **flex items**

В **Flexbox** есть свойство под названием **[flex-direction](https://developer.mozilla.org/ru/docs/Web/CSS/flex-direction)**, которое определяет направление главной оси (в каком направлении располагаются flexbox-дочерние элементы) — по умолчанию ему присваивается значение row, т.е. располагать дочерние элементы в ряд слева направо (для большинства языков).

[**align-items**](https://developer.mozilla.org/ru/docs/Web/CSS/align-items) контролирует, где на поперечной оси находятся flex элементы.

По умолчанию стоит значение stretch, которое растягивает все flex элементы, чтобы заполнить родителя вдоль поперечной (cross axis) оси. Если у родителя нет фиксированной ширины вдоль поперечной оси, все flex элементы примут длину самого длинного flex элемента. Вот почему наш первый пример по умолчанию получил столбцы с одинаковой высотой.

Значение center , которое мы использовали в коде вверху, заставляет элементы сохранять свои собственные размеры, но центрирует их вдоль поперечной оси. Вот почему кнопки текущего примера центрированы по вертикали.

Также есть значения flex-start и flex-end, которые выравнивают все элементы по началу и концу поперечной оси соответственно.

[**justify-content**](https://developer.mozilla.org/ru/docs/Web/CSS/justify-content) контролирует, где flex элементы располагаются на главной оси.

По умолчанию стоит значение flex-start, которое располагает все элементы в начале главной оси.

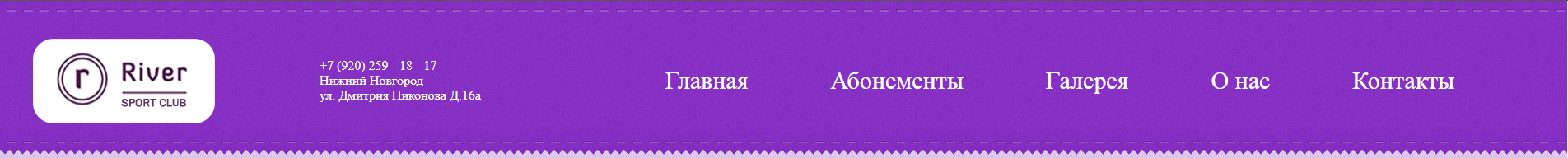
Также можно использовать flex-end, чтобы расположить их в конце.

center - также одно из значений justify-content, располагает все элементы по центру главной оси.

Значение, которое мы использовали выше, space-around, весьма полезно — оно распределяет все элементы равномерно по главной оси, с небольшим количеством свободного места на обоих концах.

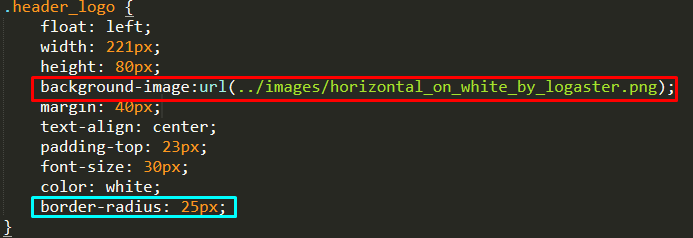
И ещё одно значение, space-between, которое очень похоже на space-around, за исключением того, что оно не оставляет места на обоих концах.

Как выглядит header после применения к нему стиля.



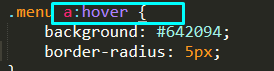
Как мы видим у нас также появилось лого, которое мы заранее создали в Photoshop и затем добавили, как background-image (далее выделено красным)

И Также скруглили края, с помощью border-radius.



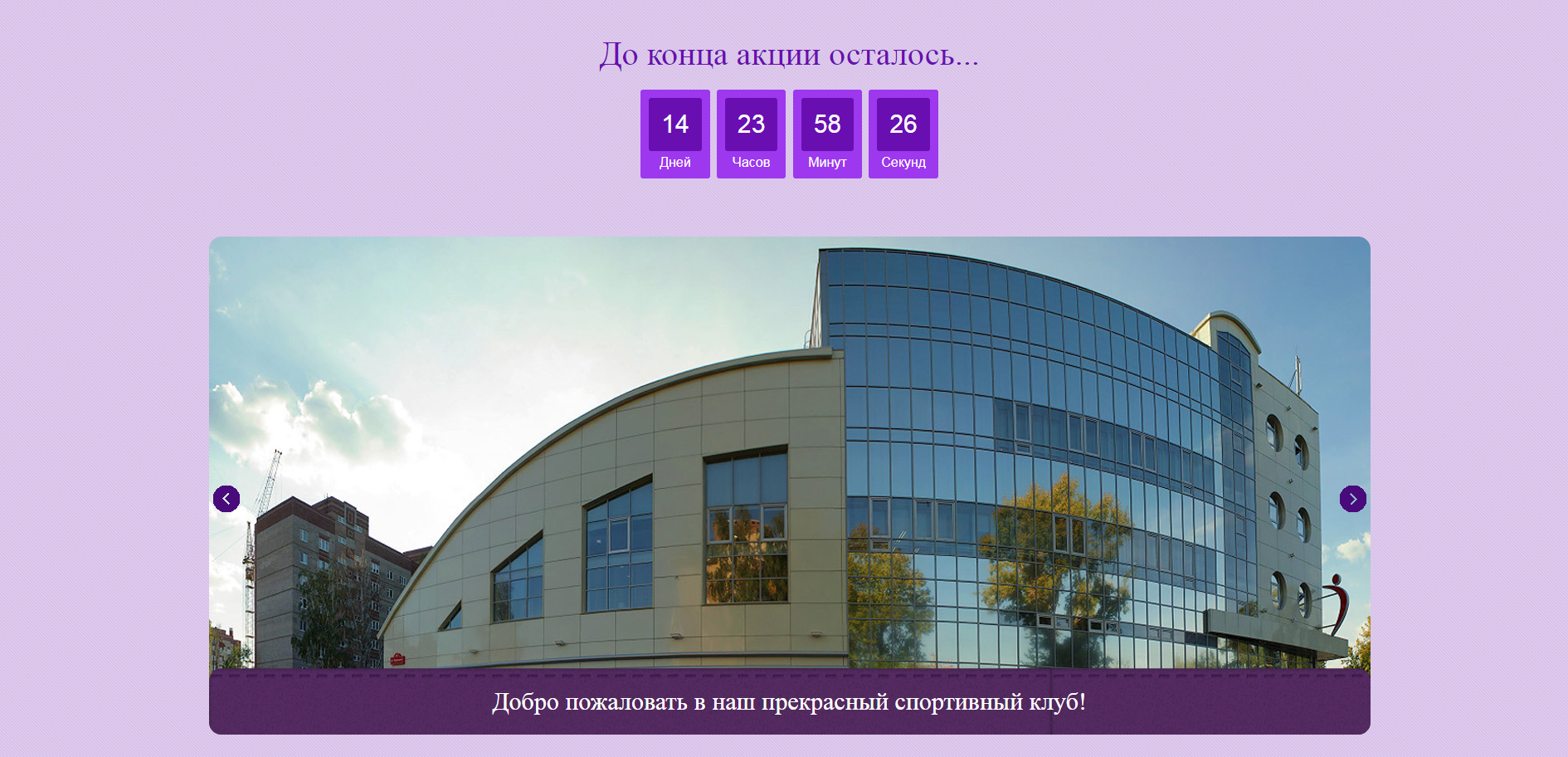
Чтобы при наведении мыши у нас подсвечивалась кнопка меню, нам нужно задать параметры для псевдокласса hover, класса menu.





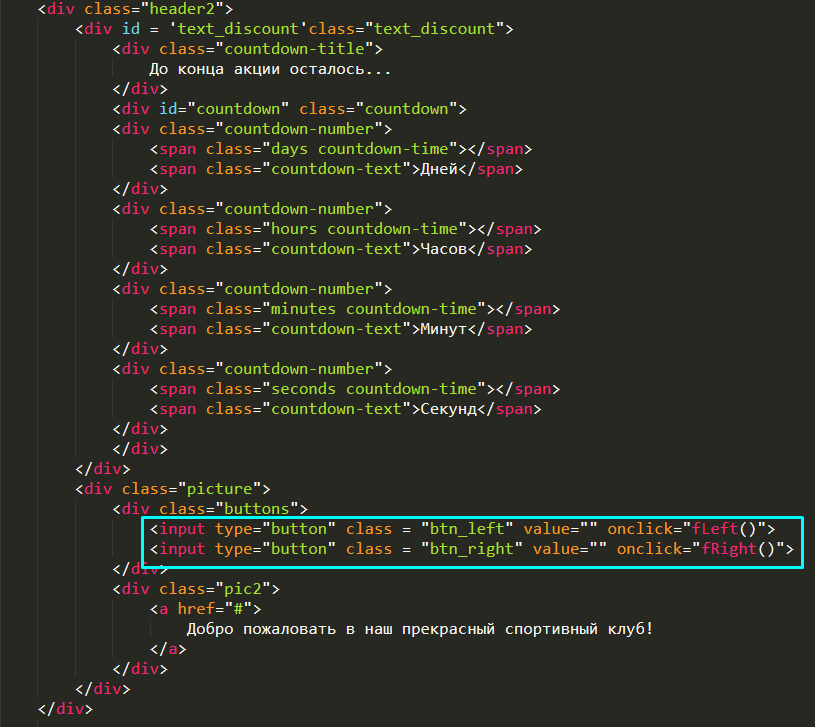
Теперь при наведении пользователем мыши на обьект меню, будет срабатывать hover, и изменять параметры на другой цвет фона и создавать немного скругленную рамку.

Далее, перейдем к созданию отсчета до конца акции и галереи фотографий.



Здесь мы также будем использовать div и затем изменять его в CSS, а также появились несколько новых тэгов:

* **Span** - является основным строковым контейнером для фразового контента, который, по существу, ничего не представляет. Он может использоваться для группировки элементов в целях стилизации
* **Input** - используется для создания интерактивных элементов управления в веб-формах для получения данных от пользователя
* **A** - является одним из важных элементов HTML и предназначен для создания ссылок. В зависимости от присутствия атрибутов name или href тег <a> устанавливает ссылку

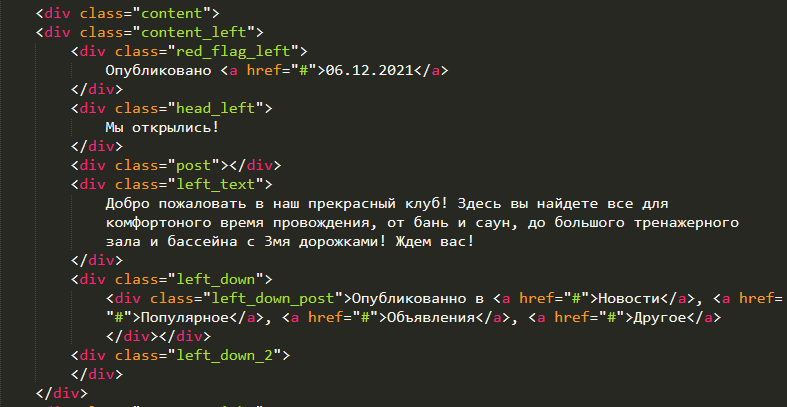


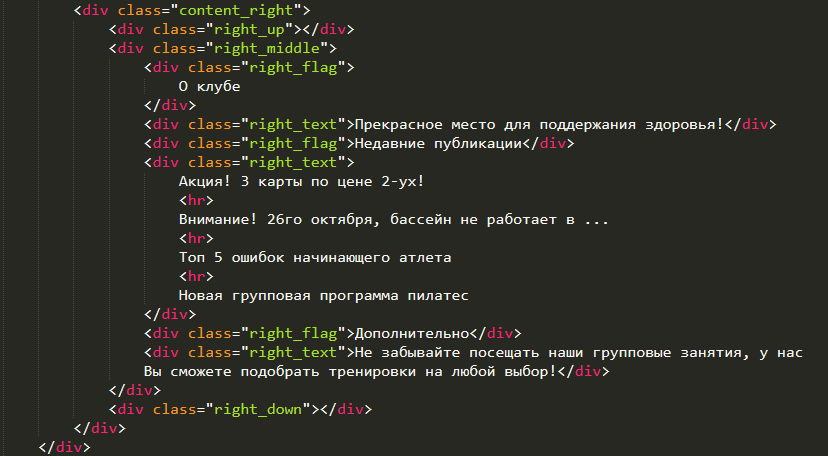
Мы будем использовать type = button, для использования кнопок, которые будут перелистывать фотографии нашего СК. Для них мы позже напишем код на языке JavaScript. В стиле зададим им нужный вид, а точнее изменим background-image на иконки стрелочек, и расположим их внутри флекс бокса, параметр которого будет ставить их на равном расстоянии друг от друга. justify-content: space-between;

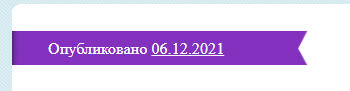


Перейдем к созданию части с контентом. Там должны располагаться недавние публикации, а также колонка с новостями и дополнительной информацией.

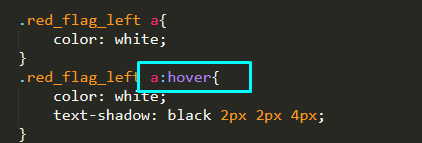
Код на HTML будет выглядеть следующим образом:





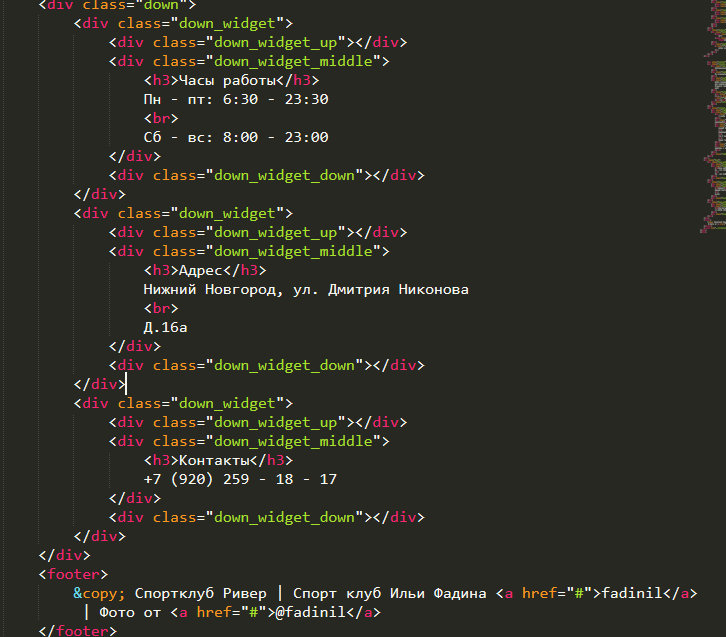


Чтобы при наведении на дату появлялась тень, мы также используем Hover.

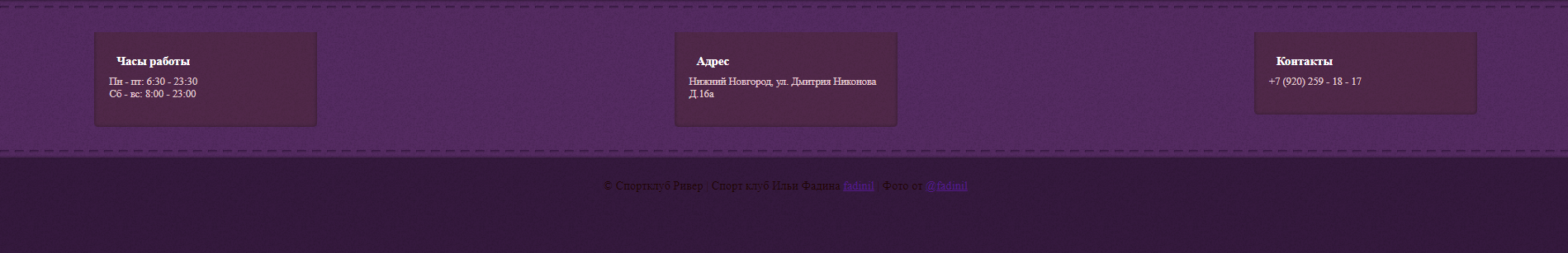




Последний этап создания страницы будет прикрепление виджетов снизу, на которых будут отображаться, время работы, Адрес, и контакты.



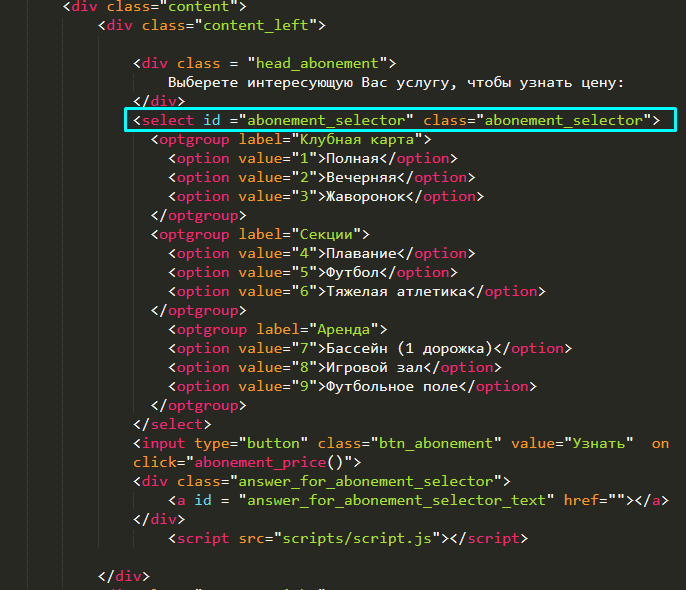
Тут мы видим, тэг footer. Он представляет собой нижний колонтитул (футер, подвал) для своего ближайшего [секционного контента](https://developer.mozilla.org/ru/docs/Web/Guide/HTML/Sections_and_Outlines_of_an_HTML5_document#%D0%97%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5_%D1%80%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%D0%BE%D0%B2_%D0%B2_HTML5) или [секционного корня](https://developer.mozilla.org/ru/docs/Web/Guide/HTML/Sections_and_Outlines_of_an_HTML5_document#%D0%9A%D0%BE%D1%80%D0%BD%D0%B8_%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D1%8F_%D1%80%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%D0%BE%D0%B2). Футер обычно содержит информацию об авторе раздела, информацию об авторском праве или ссылки на связанные документы.



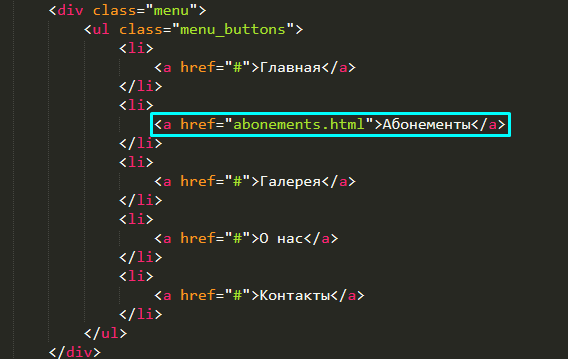
Он будет выглядеть следующим образом.

Главная страница закончена, теперь создадим следующую страницу с ценами на различные типы абонементов. Для этого создадим отдельный HTML документ, а также отдельный CSS. Т.к. сайт должен быть в единой стилистике, то перенесем некоторые элементы с главной страницы на страницу абонементы. Самое важное на этой странице это селектор, позволяющий выбирать нужный для пользователя тип абонемента, или же тип аренды. Выглядит это следующим образом.

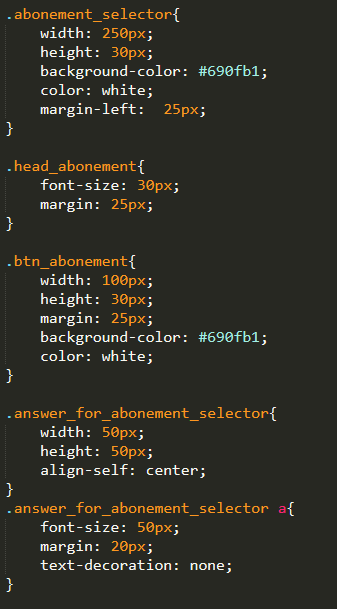
**Тег <select>** позволяет создать элемент интерфейса в виде раскрывающегося списка.



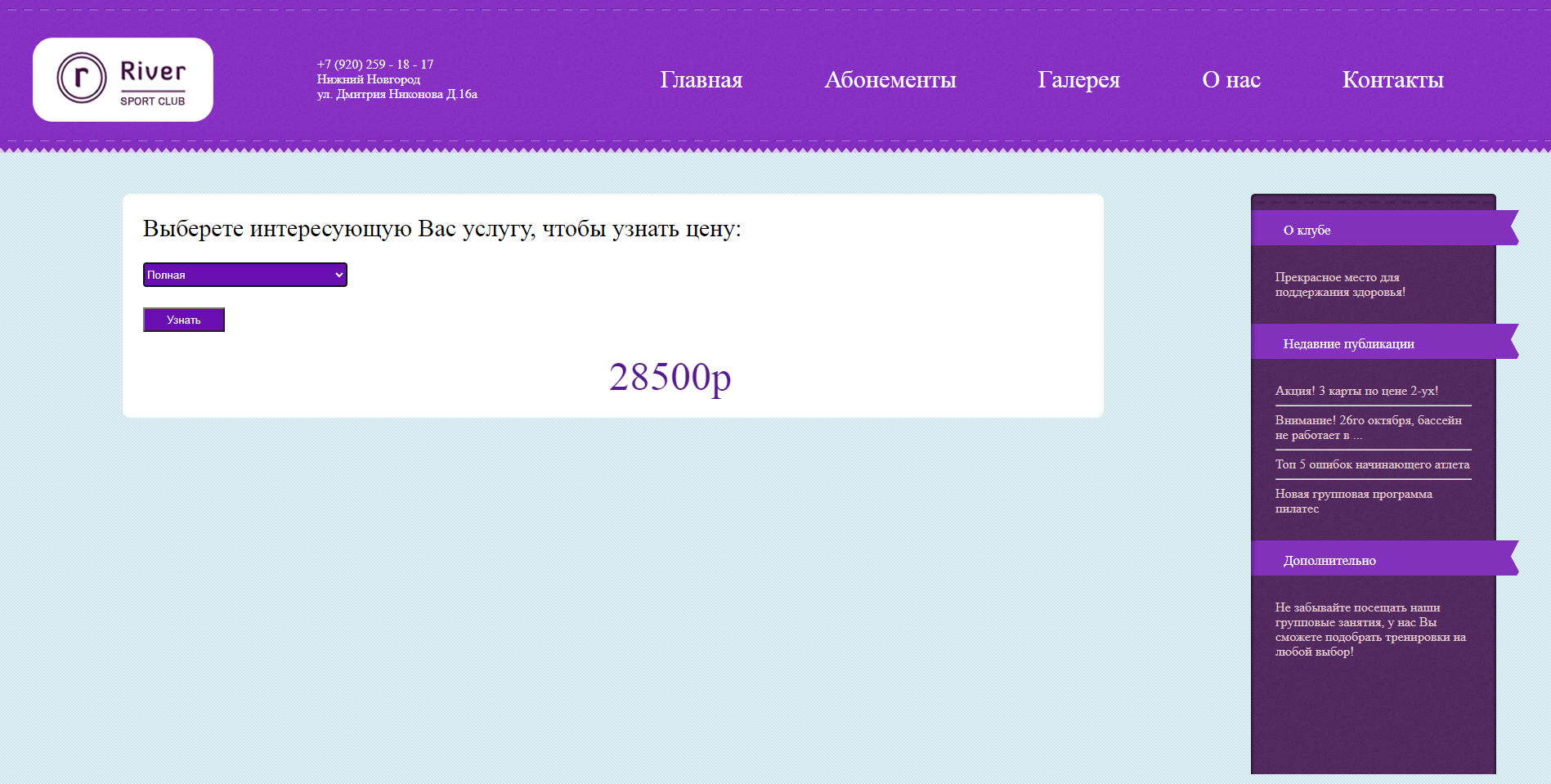
Также для перехода на эту страницу через меню, добавим ссылку в главной странице.

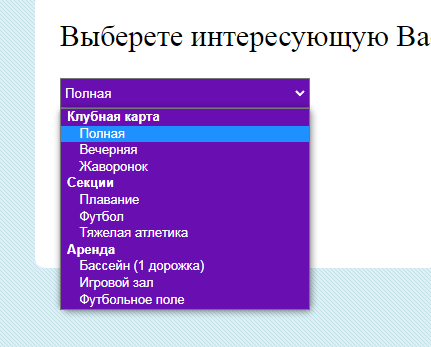


В Style для абонементов добавим параметры для следующих классов:



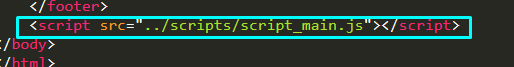
Результат будет выглядеть следующим образом:





# JavaScript

На главной странице сайта располагается галерея с кнопками, но, чтобы они функционировали потребуется написать скрипт, для этого создадим отдельный файл с расширением .js. Далее подключим его к основному документу HTML через специальный тэг script. **Тег <script>** предназначен для описания скриптов, может содержать ссылку на программу или ее текст на определенном языке. Скрипты могут располагаться во внешнем файле и связываться с любым HTML-документом. Такой подход позволяет использовать одни и те же общие функции на многих веб-страницах и ускоряет их загрузку, т.к. внешний файл кэшируется при первой загрузке, и скрипт вызывается быстрее при последующих вызовах.



Т.к. в HTML к нашим кнопка привязаны функции, создадим их.





Первое, что мы использовали – это метод GetElementsByClassName().

getElementsByClassName() метод возвращает объект [HTMLCollection](https://developer.mozilla.org/ru/docs/Web/API/HTMLCollection), содержащий в себе все дочерние элементы, которые имеют заданные имена классов. Если вызван на объекте document, будут возвращены все элементы, содержащиеся в документе. Простым языком, этот метод находит элемент класса по названию и присваивает его переменной p. Далее мы начинаем изменять параметры стиля. Т.к. нам требуется изменять картинку при вызове этой функции через кнопку, то мы будем использовать elem.style.backgroundImage. которая меняет картинку на другую, в зависимости от очереди в этой галереи. Аналогично поступим с кнопкой справа.

Следующий скрипт, отсчитывает время до конца акции в спорт клубе.



Рассмотрим первую функцию GetTimeRemaining

Метод Date.parse() разбирает строковое представление даты и возвращает количество миллисекунд, прошедших с 1 января 1970 года 00:00:00 по UTC.

Переменная t высчитывает разницу количества миллисекунд между поставленным дедлайном и текущим временем.

Далее идет разбиение миллисекунд на дни, часы, минуты и секунды. А метод Math.floor() это  - округление вниз. Округляет аргумент до ближайшего меньшего целого. Далее мы возвращаем все эти переменные.

Следующая функция intializeClock.

В ней мы находим в документе элементы, в которые будем выводить числа.

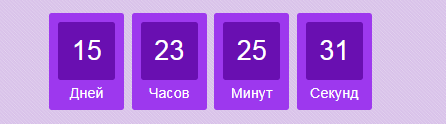
Метод  document.getElementById - Возвращает ссылку на элемент по его идентификатору ([ID](https://developer.mozilla.org/en-US/docs/Web/API/Element/id)); идентификатор является строкой, которая может быть использована для идентификации элемента; она может быть определена при помощи атрибута id в HTML или из скрипта.

Далее находим конкретные элементы для каждого промежутка времени с методом **querySelector()** возвращает первый элемент ([Element](https://developer.mozilla.org/ru/docs/Web/API/Element)) документа, который соответствует указанному селектору или группе селекторов. Если совпадений не найдено, возвращает значение null.

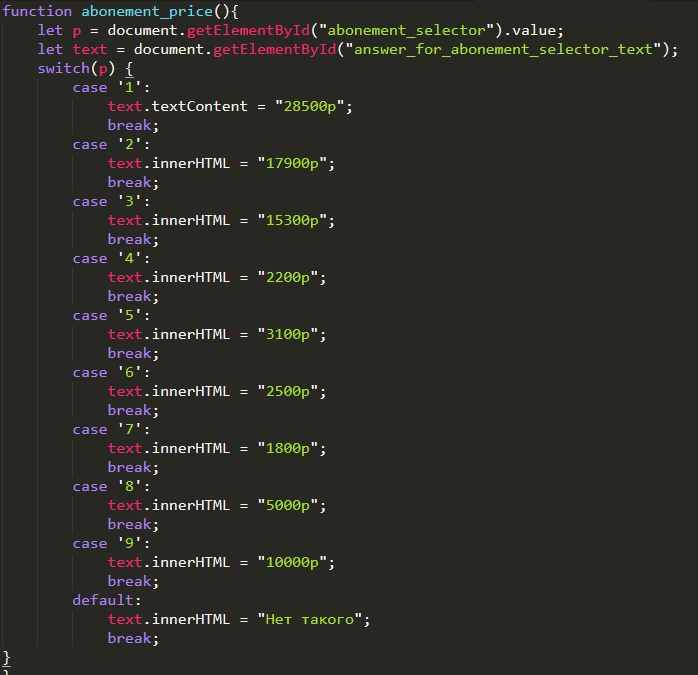
Внутри intializeClock объявим функцию с обновлением отсчета.

Свойство интерфейса [Element](https://developer.mozilla.org/ru/docs/Web/API/Element) innerHTML устанавливает или получает HTML разметку дочерних элементов. Изменим это значение на текущее. Метод **slice()** возвращает новый массив, содержащий копию части исходного массива.

И наконец переменная timeinterval содержит в себе setInterval, которая позволяет вызывать функцию регулярно, повторяя вызов через определённый интервал времени.

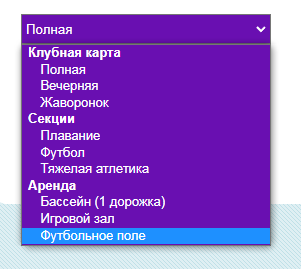
Результат: 

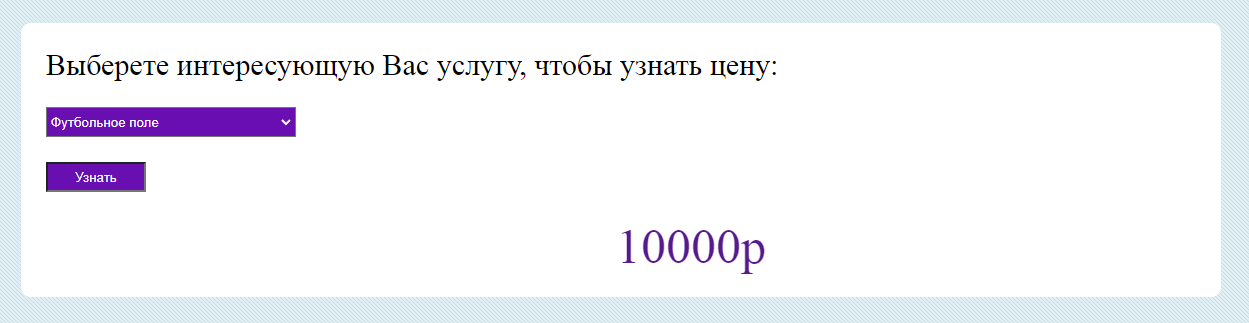
Следующий скрипт будет служить неким консультантом, который будет подсказывать цену на продукт, в зависимости от выбора клиента. Например, человек хочет узнать, сколько будет стоить аренда футбольного поля, выбирает его и нажимает кнопку узнать, и ему выводится вся нужная информация. Реализация алгоритма:



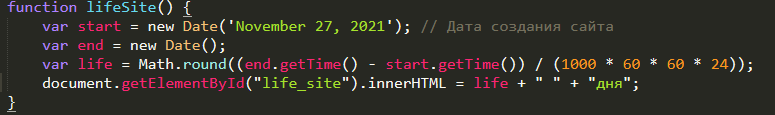
Здесь все те же методы для поиска объекта в документе, кроме оператора switch(). **Оператор switch** сравнивает выражение со случаями, перечисленными внутри неё, а затем выполняет соответствующие инструкции. Конкретно тут, мы изменяем значение, через innerHTML.

И теперь, при выборе клиентом определенного значения в селекторе, он получает ответ.





Чтобы пользователи могли видеть, сколько прошло от времени создания СК, добавим в футере специальную надпись, сколько существует сайт.



Простейшая функция, идентичная функции с отсчетом.

# Вывод

За время выполнения курсовой работы были изучены принципы разработки web-сайта, основы работы с HTML, CSS, языком программирования JavaScript.

На основе полученных знаний был спроектирован и разработан собственный web-сайт, который поможет компании не только привлекать новых клиентов в сети Интернет, но также служить неким информационным ресурсом для спорт клуба.

# Список источников

* Байдачный С. С. Маленко Д. А. ASP .NET 2. 0: Секреты создания Web-приложений СОЛОНПРЕСС. 2007. 5-98003-295-9.
* http://htmlbook.ru/html/
* <https://web-valley.ru/articles/fitnes-lending>
* <https://developer.mozilla.org/>
* <https://denis-creative.com/jstimer/>
* https://myrusakov.ru/javascript-life-site.html
* https://learn.javascript.ru/settimeout-setinterval