УТВЕРЖДАЮ
Директор ГАПОУ «СГК»
_____О.Н. Шалдыбина
«_____» _____ 2021 г.

Дополнительная профессиональная программа

профессиональной переподготовки

«Основы HTML и CSS»

Квалификация: Разработчик Web и мультимедийных приложений

Самара 2021 г.

Содержание

1.Общие положения

Нормативную правовую основу разработки программы составляют:

- Федеральный закон от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации»;
- Порядок организации и осуществления образовательной деятельности по дополнительным профессиональным программам (утв. Приказом Министерства образования и науки Российской Федерации от 1 июля 2013 г. N 499);
- Профессиональный стандарт 06.035 «Разработчик Web и мультимедийных приложений».

2. Цели реализации программы

Программа направлена на формирование у слушателей профессиональных компетенций, необходимых для выполнения нового вида профессиональной деятельности, приобретение новой квалификации.

Наименование вида профессиональной деятельности: Создание, модификация и сопровождение web-сайтов, корпоративных порталов организаций, мультимедиа и интерактивных приложений, информационных ресурсов (далее - ИР).

Цель: сформировать у обучающихся профессиональные компетенции, необходимые для выполнения нового вида профессиональной деятельности – создание, модификация и сопровождение web-сайтов, корпоративных порталов организаций, мультимедиа и интерактивных приложений, информационных ресурсов (далее - ИР), для приобретения квалификации «Специалист-техник по Web».

Задачами программы являются:

Формирование практических умений и навыков работы в области профессиональной деятельности, содействие развитию профессиональной компетенции слушателя в области осуществления профессионального и личностного самообразования, помощь в проектировании дальнейших маршрутов для совершенствования профессиональных навыков.

▪ 3. Требования к результатам освоения программы

Планируемые результаты обучения:

В результате освоения программы слушатель должен:

Уметь:

- *Пользоваться технической документацией;*
- *Проектировать web-страницы;*
- *Верстать web-страницы с использованием языка разметки HTML;*
- *Стилизовать web-страницы с использованием CSS;*
- *Работать с системой контроля версий;*
- *Разворачивать web-страницы в глобальной сети*

Знать:

- *Принципы проектирования web-страниц;*
- *Принципы оформления web-страниц;*
- *Технологию статичной, резиновой и адаптивной верстки;*
- *Методы работы с системой контроля версий;*
- *Способы передачи данных в сети Интернет;*
- *Способы развертывания web-страницы в глобальной сети.*

В результате освоения Программы слушатель должен освоить требования Профессионального стандарта 06.035 «Разработчик Web и мультимедийных приложений», в соответствии с обобщенной трудовой функцией: *Техническая поддержка процессов создания (модификации) и сопровождения информационных ресурсов,* а также трудовыми функциями: *Проверка и отладка программного кода, Работа с системой контроля версий, Верстка страниц ИР, Проведение работ по резервному копированию ИР.*

Трудовые	Анализ и проверка исходного программного кода

действия	Оценка и согласование сроков выполнения поставленных задач
	Регистрация изменений исходного текста программного кода в системе контроля версий
	слияние, разделение и сравнение исходных текстов программного кода
	Сохранение сделанных изменений программного кода в соответствии с регламентом контроля версий
	Анализ дизайн-макета ИР
	Создание структуры кода, размещающего элементы web-страницы И
	Подключение к ИР стилей оформления web-страниц
	Тестирование отображения web-страниц в различных браузерах, на различных устройствах
	Создание программного кода в соответствии с техническим заданием (готовыми специ фикациями)
	Оптимизация программного кода с использованием специализированных программных средств
	Размещение программного кода в страницах, созданных при верстке ИР
	Размещение программного кода в клиентской части ИРщение программного кода в серверной части ИР
Необходимые умения	Выявлять ошибки в программном коде
	Применять методы и приемы отладки программного кода

	Интерпретировать сообщения об ошибках,предупреждения, записи технологических журналов
	Применять современные компиляторы, отладчики и оптимизаторы программного кода
	Применять систему контроля версий для обработки исходного текста программного кода
	Применять вспомогательные инструментальные программные средства для обработки исходного текста программного кода
	Применять нормативные документы, определяющие требования к оформлению страниц ИР
	Определять возможности отображения web-страниц в размерах рабочего пространства устройств для разных видов дизайн-макетов
	Применять специализированное программное обеспечение для верстки страниц ИР
	Использовать язык разметки страниц ИР
	Применять выбранные языки программирования для написания программного кода
	Использовать выбранную среду программирования и средства системы управления базами данных
	Использовать возможности имеющейся программной архитектуры ИР

Необходимые знания	Методы и приемы отладки программного код
	Типы и форматы сообщений об ошибках, предупреждениях
	Современные компиляторы, отладчики и оптимизаторы программного кода
	Сообщения о состоянии аппаратных средств
	Регламент использования системы контроля версий
	Особенности отображения элементов ИР в различных браузерах
	Особенности отображения ИР в размерах рабочего пространства устройств
	Методы повышения читаемости программного кода
	Отраслевая нормативная техническая документация
	Синтаксис выбранного языка программирования, особенности программирования на этом языке

-

4. Содержание программы

Категория слушателей: к освоению дополнительной профессиональной программы допускаются лица, имеющие среднее профессиональное и (или) высшее образование, а также лица, получающие среднее профессиональное и (или) высшее образование, без предъявления требований к стажу работы.

Объем: 256 часов

Форма обучения: очная

Документ, выдаваемый по результатам освоения программы: диплом о профессиональной переподготовке.

Квалификация: Специалист-техник по Web

4.1. Учебный план

№	Наименование модулей	Всего, ак.час.	В том числе			Форма контроля
			лекции	практ. занятия	промежут. и итог. контроль	
1	*2*	*3*	*4*	*5*	*6*	*7*
1.	**Модуль 1.** Веб-разработка.	12	6	4	2	*Устный опрос*
2.	**Модуль 2.** Основы HTML.	26	7	17	2	*Проверочная работа*
3.	**Модуль 3.** Основы CSS.	82	22	58	2	*Проверочная работа*
4.	**Модуль 4.** Верстка	58	14	38	6	*Проверочная работа*
5.	**Модуль 5.** Публикация web-страницы	26	9	11	6	*Проверочная работа*
6.	Производственная практика	40		36	4	*Проверочная работа*
7.	Консультация	6				
8.	Итоговая аттестация (экзамен)	6			6	*Экзамен*
	ИТОГО:	256	58	164	28	

4.2. Учебно-тематическое планирование

№			В том числе	

п/п	Наименование модулей	Всего часов	лекции	практ. занятия	промежут. и итог контроль	Форма контроля
1	*2*	*3*	*4*	*5*	*6*	*7*
1.	**Модуль 1.** Веб-разработка					*Устный опрос*
1.1	Всемирная сеть (World Wide Web)	2	2			
1.2	HTTP (HyperText Transfer Protocol)	4	2	2		
1.3	HTTP URL	4	2	2		
	Промежуточная аттестация	2			2	
2.	**Модуль 2.** Основы HTML					Проверочная работа
2.1	Структура HTML страницы	4	1	3		
2.2	Теги и атрибуты	2	1	1		
2.3	Блочные элементы	3	1	2		
2.4	Строчные элементы	3	1	2		
2.5	Семантика HTML	4	1	3		
2.6	Табличные элементы	4	1	3		
2.7	Элементы форм	4	1	3		
	Промежуточная аттестация	2			2	
3.	**Модуль 3.**					

	Основы CSS					
3.1	Каскадные таблицы стилей (CSS – Cascading Style Sheets)	4	2	2		
3.2	Структура CSS	4	1	3		
3.3	Базовый синтаксис	4	1	3		
3.4	Подключение CSS к HTML	4	1	3		
3.5	Атрибуты	4	1	3		
3.6	Селекторы	4	1	3		
3.7	Идентификаторы (ID)	4	1	3		
3.8	Классы (Class)	4	1	3		
3.9	Множественные классы (Multiple Classes)	4	1	3		
3.10	Псевдоклассы	4	1	3		
3.11	Наследование	4	1	3		
3.12	Комбинаторика	4	1	3		
3.13	Множественные селекторы	4	1	3		
3.14	Блочная модель	4	1	3		
3.15	Отображение и позиционирование	4	2	2		
3.16	Абсолютное, относительное и	4	1	3		

	фиксированное позиционирование					
3.17	Позиция Sticky	4	1	3		
3.18	Цветовое оформление	4	1	3		
3.19	Работа с текстом	4	1	3		
3.20	Работа с графикой	4	1	3		
	Промежуточная аттестация	2			2	
4.	**Модуль 4.** Верстка					Проверочная работа
4.1	Разметка страницы по макету	6	2	4		
4.2	Экспортирование графики	6	2	4		
4.3	Статичная верстка	4	2	2		
4.4	Резиновая верстка	6	2	4		
4.5	Адаптивная верстка	6	2	4		
4.6	Введение в гибкую сетку (Flexbox)	4	2	2		
4.7	Спецификация Grid Layout	4	2	2		
4.8	Верстка статичной страницы по макету	4		4		
4.9	Верстка резиновой страницы по макету	6		6		
4.10	Верстка адаптивной страницы по макету	6		6		

	Промежуточная аттестация	6			6	
5.	**Модуль 5.** Публикация web-страницы					Проверочная работа
5.1	Хостинг и доменные имена	4	2	2		
5.2	Подготовка файлов для публикации	2	1	1		
5.3	Знакомство с GitHub хостинг - сервисом	4	2	2		
5.4	Создание репозитория на GitHub	6	2	4		
5.5	Публикация web-страницы	4	2	2		
	Промежуточная аттестация	6			6	
	Производственная практика	40		36	4	Проверочная работа
	Консультация	6				
	Итоговая аттестация	6			6	Экзамен
	ИТОГО	256				

4.3 Структура программы

Модуль 1. Веб-разработка

Тема 1.1

Всемирная сеть (World Wide Web)

Всемирная сеть — сокращённо: WWW, W3, или Web; Сеть, паутина или веб — всемирная система публичных веб-страниц в сети Интернет. Сеть не является Интернетом: Сеть лишь использует Интернет как среду передачи информации и данных.

Тим Бернерз-Ли предложил архитектуру, которая стала известна под названием World Wide Web. В 1990 году в ЦЕРН (в своей лаборатории физ. исследований) он создал первый веб-сервер, браузер и веб-страницу на своём компьютере. В 1991 году он объявил про своё творение в группе новостей alt.hypertext, тем самым, обозначив момент, когда Web стал достоянием общества.

Система, которую мы называем Web состоит из нескольких компонентов:

Протокол HTTP обеспечивает обмен данными между сервером и клиентом.

Отсылая запрос серверу на определённый ресурс, клиент предоставляет неповторимый идентификатор, который зовётся URL (унифицированное расположение ресурсов) или URI (en-US) (унифицированный идентификатор ресурса).

HTML (язык гипертекстовой разметки) - самый распространённый формат веб-документов.

Связь страниц с помощью гиперссылок (en-US) является главной концепцией Web.

Тема 1.2

HTTP (HyperText Transfer Protocol)

Протокол передачи гипертекста (Hypertext Transfer Protocol - HTTP) - это прикладной протокол для передачи гипертекстовых документов, таких как HTML. Он создан для связи между веб-браузерами и веб-серверами, хотя в принципе HTTP может использоваться и для других целей. Протокол следует классической клиент-серверной модели, когда клиент открывает соединение для

создания запроса, а затем ждёт ответа. HTTP - это протокол без сохранения состояния, то есть сервер не сохраняет никаких данных (состояние) между двумя парами "запрос-ответ". Несмотря на то, что HTTP основан на TCP/IP, он также может использовать любой другой протокол транспортного уровня с гарантированной доставкой.

Тема 1.3

HTTP URL

Единый указатель ресурса (Uniform Resource Locator, URL) — строка, однозначно определяющая, где в интернете находится ресурс.

В контексте HTTP, URL называют "адрес" (web address)" или "ссылку" (link). В браузере URL отображается в адресной строке , например, https://developer.mozilla.org.

Кроме того, URL используют при передаче файлов через FTP , в электронной почте через (SMTP (en-US)) и других местах.

Промежуточная аттестация в форме устного опроса.

Модуль 2. Основы HTML

Тема 2.1

Структура HTML страницы

Любой HTML-документ начинается с базовой структуры. Она включает в себя теги, которые есть в любом HTML-файле. Эти теги и служебная информация нужны браузеру для корректного отображения информации.

Базовая структура любого HTML-документа:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Моя первая страница</title>
</head>
<body>

</body>
</html>
```

Этот набор кажется не очень большим, но браузеру он сообщает множество полезной информации. В этом уроке разберёмся с каждой строчкой этой структуры.

Тема 2.2

Теги и атрибуты

HTML-теги — основа языка HTML. Теги используются для разграничения начала и конца элементов в разметке.

Каждый HTML-документ состоит из дерева HTML-элементов и текста. Каждый HTML-элемент обозначается начальным (открывающим) и конечным (закрывающим) тегом. Открывающий и закрывающий теги содержат имя тега.

Все HTML-элементы делятся на пять типов:

- пустые элементы — <area>, <base>,
, <col>, <embed>, <hr>, <img>, <input>, <link>, <menuitem>, <meta>, <param>, <source>, <track>, <wbr>;
- элементы с неформатированным текстом — <script>, <style>;
- элементы, выводящие неформатированный текст — <textarea>, <title>;
- элементы из другого пространства имён — MathML и SVG;

- обычные элементы — все остальные элементы.

HTML-атрибуты сообщают браузеру, каким образом должен отображаться тот или иной элемент страницы. Атрибуты позволяют сделать более разнообразными внешний вид информации, добавляемой с помощью одинаковых тегов.

Тема 2.3

Блочные элементы

Блочные элементы характеризуются тем, что занимают всю доступную ширину, высота элемента определяется его содержимым, и он всегда начинается с новой строки.

Практика: Ознакомиться с блочными элементами из учебника: http://htmlbook.ru/html/type/block

Тема 2.4

Строчные элементы

Строчными называются такие элементы веб-страницы, которые являются непосредственной частью другого элемента, например, текстового абзаца. В основном они используются для изменения вида текста или его логического выделения.

Практика: Ознакомиться со строчными элементами из учебника: http://htmlbook.ru/html/type/block

Тема 2.5

Семантика HTML

Семантические элементы HTML5 доступно описывают свой смысл или назначение как для браузеров, так и для веб-разработчиков.

До появления стандарта HTML5 вся разметка страниц осуществлялась преимущественно с помощью элементов , которым присваивали классы class или идентификаторы id для наглядности разметки (например,). С их помощью в HTML-документе размещали верхние и нижние колонтитулы, боковые панели, навигацию и многое другое.

Стандарт HTML5 предоставил новые элементы для структурирования, группировки контента и разметки текстового содержимого. Новые семантические элементы позволили улучшить структуру веб-страницы, добавив смысловое значение заключенному в них содержимому (было , стало <header>). Для отображения внешнего вида элементов не задано никаких правил, поэтому элементы можно стилизовать по своему усмотрению. Для всех элементов доступны глобальные атрибуты.

Согласно спецификации HTML5 каждый элемент принадлежит к определенной (ноль или более) категории. Каждая из них группирует элементы со схожими характеристиками. Выделяют следующие общие категории:

- Метаданные
- Потоковое содержимое
- Секционное содержимое
- Заголовочное содержимое
- Текстовое содержимое
- Встроенное содержимое
- Интерактивное содержимое

Практика: Ознакомиться с семантическими элементами HTML5 из учебника: https://html5book.ru/html5-semantic-elements/

Тема 2.6

Табличные элементы

HTML-таблицы упорядочивают и выводят на экран данные с помощью строк или столбцов. Таблицы состоят из ячеек, образующихся при пересечении строк и столбцов.

Ячейки таблиц могут содержать любые HTML-элементы, такие как заголовки, списки, текст, изображения, элементы форм, а также другие таблицы. Каждой

таблице можно добавить связанный с ней заголовок, расположив его перед таблицей или после неё.

Таблицы больше не используются для вёрстки веб-страниц и компоновки отдельных элементов, потому что такой приём не обеспечивает гибкость структуры и адаптивность сайта, существенно увеличивая HTML-разметку.

Для всех элементов таблицы доступны глобальные атрибуты, а также собственные атрибуты.

Практика: Составить таблицу по примеру, используя HTML-разметку.

№ п/п	Наименование	Цена, руб.
1	Карандаш цветной	20,00
2	Линейка 20см	30,00

Тема 2.7

Элементы форм

HTML-формы являются элементами управления, которые применяются для сбора информации от посетителей веб-сайта.

Веб-формы состоят из набора текстовых полей, кнопок, списков и других элементов управления, которые активизируются щелчком мыши. Технически формы передают данные от пользователя удаленному серверу.

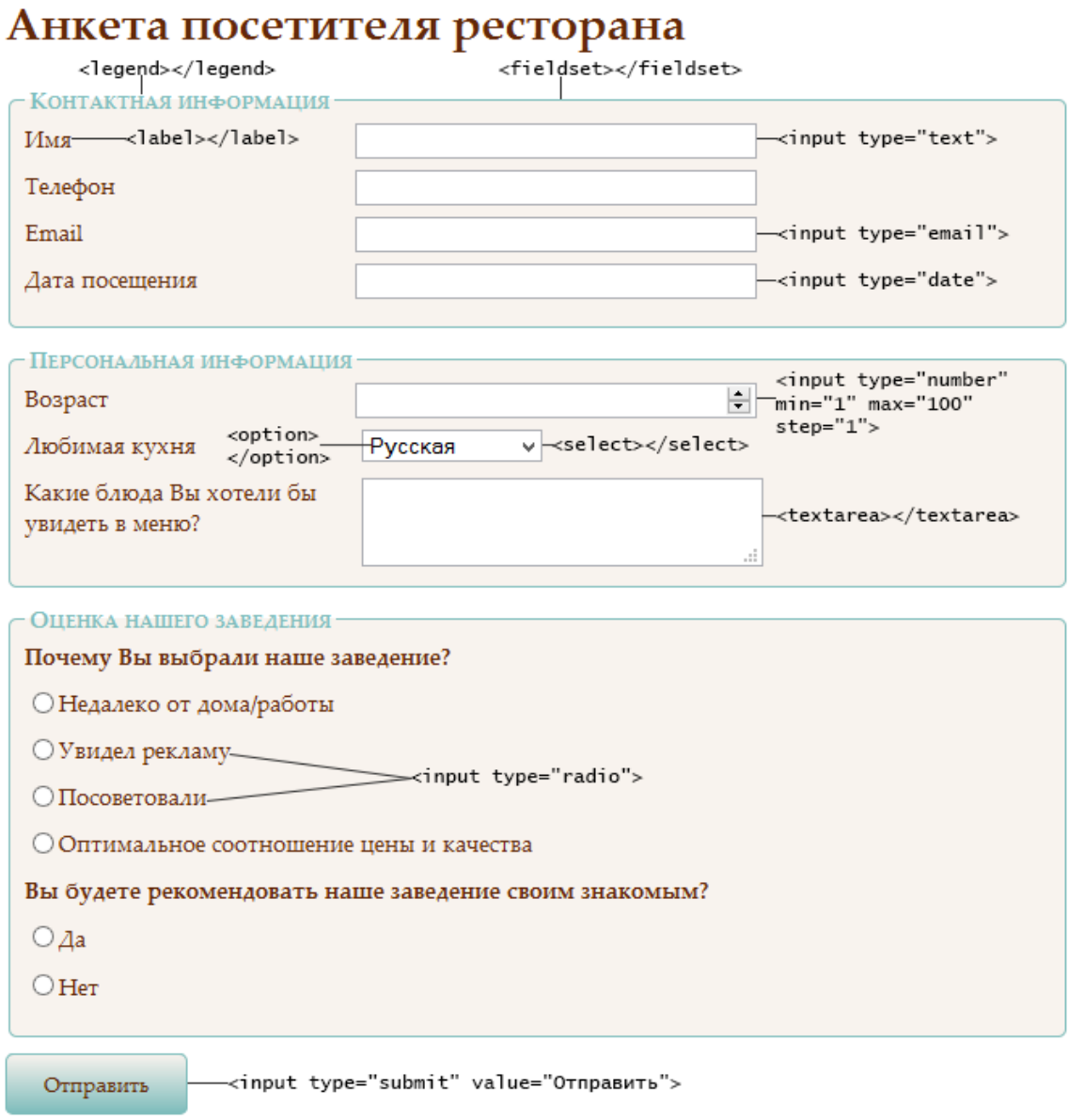
Для получения и обработки данных форм используются языки веб-программирования, такие как PHP, Perl.

До появления HTML5 веб-формы представляли собой набор нескольких элементов <input type="text">, <input type="password">, завершающихся кнопкой <input type="submit">. Для стилизации форм в разных браузерах приходилось прилагать немало усилий. Кроме того, формы требовали применения JavaScript для проверки введенных данных, а также были лишены специфических типов полей ввода для указания повседневной информации типа дат, адресов электронной почты и URL-адресов.

HTML5-формы решили большинство этих распространенных проблем благодаря наличию новых атрибутов, предоставив возможность изменять внешний вид элементов форм за счет CSS3.

Промежуточная аттестация в форме проверочной работы:

Задание: Составить форму по примеру, используя полученные знания и HTML-разметку.



Модуль 3. Основы CSS

Тема 3.1

Каскадные таблицы стилей (CSS – Cascading Style Sheets)

Стилем или CSS (Cascading Style Sheets, каскадные таблицы стилей) называется набор параметров форматирования, который применяется к элементам документа, чтобы изменить их внешний вид. Возможность работы со стилями издавна включают в развитые издательские системы и текстовые редакторы, тем самым позволяя одним нажатием кнопки придать тексту заданный, заранее установленный вид. Теперь это доступно и создателям сайта, когда цвет, размеры текста и другие параметры хранятся в определенном месте и легко «прикручиваются» к любому тегу. Еще одним преимуществом стилей является то, что они предлагают намного больше возможностей для форматирования, чем обычный HTML.

Тема 3.2
Структура CSS

CSS представляет собой мощную систему, расширяющую возможности дизайна и верстки веб-страниц. Обычно CSS-стили используются для создания и изменения стиля элементов веб-страниц и пользовательских интерфейсов, написанных на языках HTML и XHTML, но также могут быть применены к любому виду XML-документа, в том числе XML, SVG и XUL.

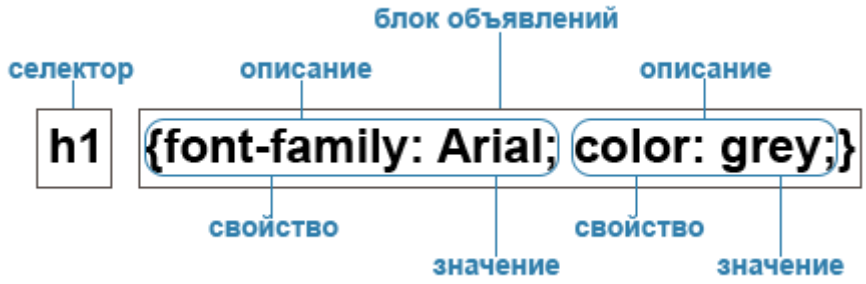
Отделяя стиль представления документов от содержимого документов, CSS упрощает создание веб-страниц и обслуживание сайтов

CSS поддерживает таблицы стилей для конкретных носителей, поэтому авторы могут адаптировать представление своих документов к визуальным браузерам, слуховым устройствам, принтерам, брайлевским устройствам, карманным устройствам и т.д.

Каскадные таблицы стилей описывают правила форматирования элементов с помощью свойств и допустимых значений этих свойств. Для каждого элемента можно использовать ограниченный набор свойств, остальные свойства не будут оказывать на него никакого влияния.

Тема 3.3
Базовый синтаксис

Объявление стиля состоит из двух частей: селектора и объявления. В HTML имена элементов нечувствительны к регистру, поэтому «h1» работает так же, как и «H1». Объявление состоит из двух частей: имя свойства (например, color) и значение свойства (grey). Селектор сообщает браузеру, какой именно элемент форматировать, а в блоке объявления (код в фигурных скобках) перечисляются форматирующие команды — свойства и их значения.



Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.4

Подключение CSS к HTML

Внешняя таблица стилей представляет собой текстовый файл с расширением .css, в котором находится набор CSS-стилей элементов. Файл создаётся в редакторе кода, так же как и HTML-страница. Внутри файла могут содержатся только стили, без HTML-разметки. Внешняя таблица стилей подключается к веб-странице с помощью тега <link>, расположенного внутри раздела <head></head>. Такие стили работают для всех страниц сайта.

К каждой веб-странице можно присоединить несколько таблиц стилей, добавляя последовательно несколько тегов <link>, указав в атрибуте тега media назначение данной таблицы стилей. rel="stylesheet" указывает тип ссылки (ссылка на таблицу стилей).

```
<head>
<link rel="stylesheet" href="css/style.css">
<link rel="stylesheet" href="css/assets.css" media="all">
</head>
```

Атрибут type="text/css" не является обязательным по стандарту HTML5, поэтому его можно не указывать. Если атрибут отсутствует, по умолчанию использует

Внутренние стили встраиваются в раздел <head></head> HTML-документа и определяются внутри тега <style></style>. Внутренние стили имеют приоритет над внешними, но уступают встроенным стилям (заданным через атрибут style).ся значение type="text/css".

```
<head>
<style>
h1,
h2 {
color: red;
font-family: "Times New Roman", Georgia, Serif;
line-height: 1.3em;
}
</style>
</head>
<body>
...
</body>
```

Когда мы пишем встроенные стили, мы пишем CSS-код в HTML-файл, непосредственно внутри тега элемента с помощью атрибута style:

```
<p style="font-weight: bold; color: red;">Обратите внимание на этот текст.</p>
```

Такие стили действуют только на тот элемент, для которого они заданы.

Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.5
Атрибуты

Атрибуты в CSS позволяют задавать стиль элементам веб-страницы в зависимости от наличия у них определенных атрибутов.

С помощью атрибутов можно также осуществлять выборку тех элементов, у которых имеется определенное значение атрибута.

Можно делать точную выборку элементов веб-страницы по началу или концу текста в значении атрибута.

Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.6

Селекторы

Селектор определяет, к какому элементу применять то или иное CSS-правило.

Обратите внимание - не существует селекторов, которые бы позволили выбрать родителя (содержащий контейнер) или соседа родителя или потомков соседа родителя.

Универсальный селектор

Выбирает все элементы. По желанию, он может быть ограничен определённым пространством имён или относиться ко всему пространству имён.

Синтаксис: * ns|* *|*

Пример: * будет соответствовать всем элементам на странице.

Селекторы по типу элемента

Этот базовый селектор выбирает тип элементов, к которым будет применяться правило.

Синтаксис: элемент

Пример: селектор input выберет все элементы <input>.

Селекторы по классу

Этот базовый селектор выбирает элементы, основываясь на значении их атрибута class.

Синтаксис: .имяКласса

Пример: селектор .index выберет все элементы с соответствующим классом (который был определён в атрибуте class="index").

Селекторы по идентификатору

Этот базовый селектор выбирает элементы, основываясь на значении их id атрибута. Не забывайте, что идентификатор должен быть уникальным, т. е. использоваться только для одного элемента в HTML-документе.

Синтаксис: #имяИдентификатора

Пример: селектор #toc выберет элемент с идентификатором toc (который был определён в атрибуте id="toc").

Селекторы по атрибуту

Этот селектор выбирает все элементы, имеющие данный атрибут или атрибут с определённым значением.

Синтаксис: [attr] [attr=value] [attr~=value] [attr|=value] [attr^=value] [attr$=value] [attr*=value]

Пример: селектор [autoplay] выберет все элементы, у которых есть атрибут autoplay (независимо от его значения).

Ещё пример: a[href$=".jpg"] выберет все ссылки, у которых адрес заканчивается на ".jpg".

Ещё пример: a[href^="https"] выберет все ссылки, у которых адрес начинается на "https".

Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.7

Идентификаторы (ID)

Глобальный атрибут id устанавливает элементу уникальный идентификатор (ID), имя которого должно быть уникальным в документе (на странице). Его назначением является идентификация элемента при связывании (использование идентификатора фрагмента), скриптинге или стилизации (с помощью CSS).

Значение id не должно содержать пропусков (пробелов, табуляции и т.д). Браузеры обрабатывают пробелы, как часть уникального идентификатора. В отличии от атрибута class, который позволяет разделять пробелами значение (указывать несколько классов через пробел), элементы могут иметь только один ID.

Тема 3.8

Классы (Class)

Глобальный атрибут class это разделённый пробелом список регистров зависимых классов элемента. Классы позволяют CSS и Javascript выбирать и получать доступ с помощью селекторов класса или функций, таких как методы DOM document.getElementsByClassName.

Хотя спецификация не предъявляет требований к именам классов, веб-разработчикам рекомендуется использовать имена, описывающие семантическое назначение элемента, а не представление элемента. Например, атрибут описывает атрибут, а не курсив, хотя элемент этого класса может быть представлен курсивом. Семантические имена остаются логичными даже при изменении представления страницы.

Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.9

Множественные классы (Multiple Classes)

Один и тот же класс может быть присвоен разным элементам и, кроме этого, одному элементу можно присвоить несколько классов, благо, спецификация CSS 2.1 об этом явно указывает. Маленькая ремарка — следует помнить о том, что название класса не может начинаться с цифры или дефиса.

Пример:

```
<p class="one two three">
    А что это за шаги такие на лестнице? - А это нас арестовывать идут.
</p>
```

Даже невооруженным глазом видно, что у параграфа 3 класса, отделенных друг от друга пробелами. Как таким элементом можно манипулировать?

Обратиться к такому многоклассовому параграфу можно называя один из классов

```
p.one {
    color:red;
}
```

два

```
p.one.three {
    color:blue;
}
```

или все

```
p.one.two.three {
    color:black;
}
```

Если все эти правила будут упомянуты в таблице стилей в точно таком же порядке, то цвет текста в абзаце станет черным, поскольку: 1) действует принцип "работает последний", и, что более важно 2) у правила с одновременным обращением к элементу с помощью нескольких классов «сила воздействия» (специфичность, говоря официальным языком) выше одиночных. Все последние версии браузеров легко справляются с такой конструкцией.

Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.10

Псевдоклассы

Псевдокласс в CSS — это ключевое слово, добавленное к селектору, которое определяет его особое состояние. Например, :hover может быть использован для изменения цвета кнопки при наведении курсора на неё.

Как и с обычными классами, можно совмещать вместе в одном селекторе любое число псевдоклассов.

Список стандартных псевдоклассов

- :active
- :any (en-US)
- :any-link
- :checked
- :default
- :defined
- :dir()
- :disabled
- :empty
- :enabled
- :first
- :first-child
- :first-of-type
- :fullscreen
- :focus
- :hover
- :indeterminate
- :in-range
- :invalid
- :lang()
- :last-child
- :last-of-type
- :left
- :link
- :not()
- :nth-child()
- :nth-last-child()
- :nth-last-of-type()
- :nth-of-type()
- :only-child
- :only-of-type
- :optional
- :out-of-range
- :read-only
- :read-write
- :required
- :right
- :root
- :scope (en-US)
- :target
- :valid
- :visited

Практика: Подробнее ознакомиться с псевдоклассами в электронном учебнике: *https://developer.mozilla.org/ru/docs/Web/CSS/Pseudo-classes*

Тема 3.11
Наследование

Каскад таблицы стилей, если говорить упрощённо, означает, что порядок следования правил в CSS имеет значение; когда применимы два правила, имеющие одинаковую специфичность, используется то, которое идёт в CSS последним.

В приведённом ниже примере у нас есть два правила, которые могут применяться к h1. В результате h1 окрасится синим цветом — эти правила имеют идентичный селектор и, следовательно, одинаковую специфичность, поэтому побеждает последний в порядке следования.

```
h1 {
    color: red;
}
h1 {
    color: blue;
}
```

```
<h1>This is my heading.</h1>
```

Специфичность определяет, как браузер решает, какое именно правило применяется в случае, когда несколько правил имеют разные селекторы, но, тем не менее, могут быть применены к одному и тому же элементу. Различные типы селекторов (селекторы элементов h1{...}, селекторы классов, селекторы идентификаторов и т.д) имеют разной степени влияние на элементы страницы. Чем более общее влияние оказывает селектор на элементы страницы тем меньше его специфичность, конкретность. По существу, это мера того, насколько специфическим будет отбор по селектору:

Селектор элементов (например h1) менее специфичен — он выберет все элементы этого типа на странице — поэтому получит меньше баллов.
Селектор класса более специфичен — он выберет только те элементы на странице, которые имеют конкретное значение атрибута class — поэтому получит больше баллов, селектор класса применится после селектора элемента и поэтому перекроет его стили.

Например. Как указано ниже, у нас опять есть два правила, которые могут применяться к h1. h1 в результате будет окрашен красным цветом — селектор класса даёт своему правилу более высокую специфичность, поэтому он будет применён, несмотря на то, что правило для селектора элемента расположено ниже в таблице стилей.

```
.main-heading {
    color: red;
}

h1 {
    color: blue;
}
```

```
<h1 class="main-heading">This is my heading.</h1>
```

Наследование также надо понимать в этом контексте — некоторые значения свойства CSS, установленные для родительских элементов наследуются их дочерними элементами, а некоторые нет.

Например, если вы установили значение color и font-family для элемента, то каждый элемент внутри него также будет иметь этот цвет и шрифт, если только вы не применили к ним напрямую стиль с другим цветом и шрифтом.

```
body {
    color: blue;
}

span {
    color: black;
}
```

```
<p>As the body has been set to have a color of blue this is inherited through the
descendants.</p>
<p>We can change the color by targetting the element with a selector, such as this
<span>span</span>.</p>
```

Некоторые свойства не наследуются — например, если вы установили для элемента width равным 50%, все его дочерние элементы не получат ширину в 50% от ширины своего родительского элемента. Если бы это было так, CSS было бы чрезвычайно трудно использовать!

Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.12
Комбинаторика

Селектор потомка — обычно представляется символом пробела ( ) — соединяет два селектора так, что элементы, соответствующие второму селектору, выбираются, если они имеют предка (родителя, родителя родителя, родителя родителя родителя и т.д.), соответствующего первому селектору. Селекторы, которые используют комбинатор потомка, называются селекторами потомка.

```
body article p
```

В приведённом ниже примере выбирается только тот элемент <p>, который находится внутри элемента с классом .box.

Interactive editor

```
.box p {
    color: red;
}
```

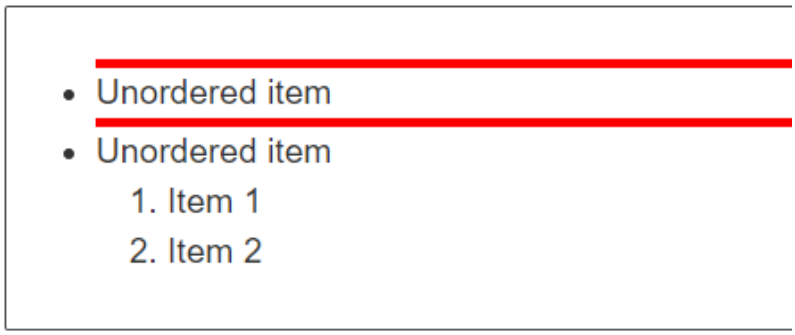
```
<div class="box"><p>Text in .box</p></div>
<p>Text not in .box</p>
```

Дочерний комбинатор (>) помещается между двумя селекторами CSS. При этом будут выбраны только те элементы, соответствующие второму селектору, которые являются прямыми потомками элементов, соответствующих первому селектору. Все элементы-потомки на более низких уровнях иерархии будут пропущены. Например, чтобы выбрать только те элементы <p>, которые являются дочерними элементами <article>, селектор пишется так:

```
article > p
```

Другой пример. Имеется неупорядоченный список, заключающий в себе другой, упорядоченный список. Дочерний комбинатор используется для того, чтобы выбрать только те элементы <li>, которые являются прямыми потомками <ul>, и присвоить им верхнюю границу красного цвета.

Если вы уберёте символ >, указывающий на то, что это селектор с дочерним комбинатором, селектор превратится в селектор всех потомков (комбинатор - пробел) и все элементы <li> получат верхнюю границу красного цвета.

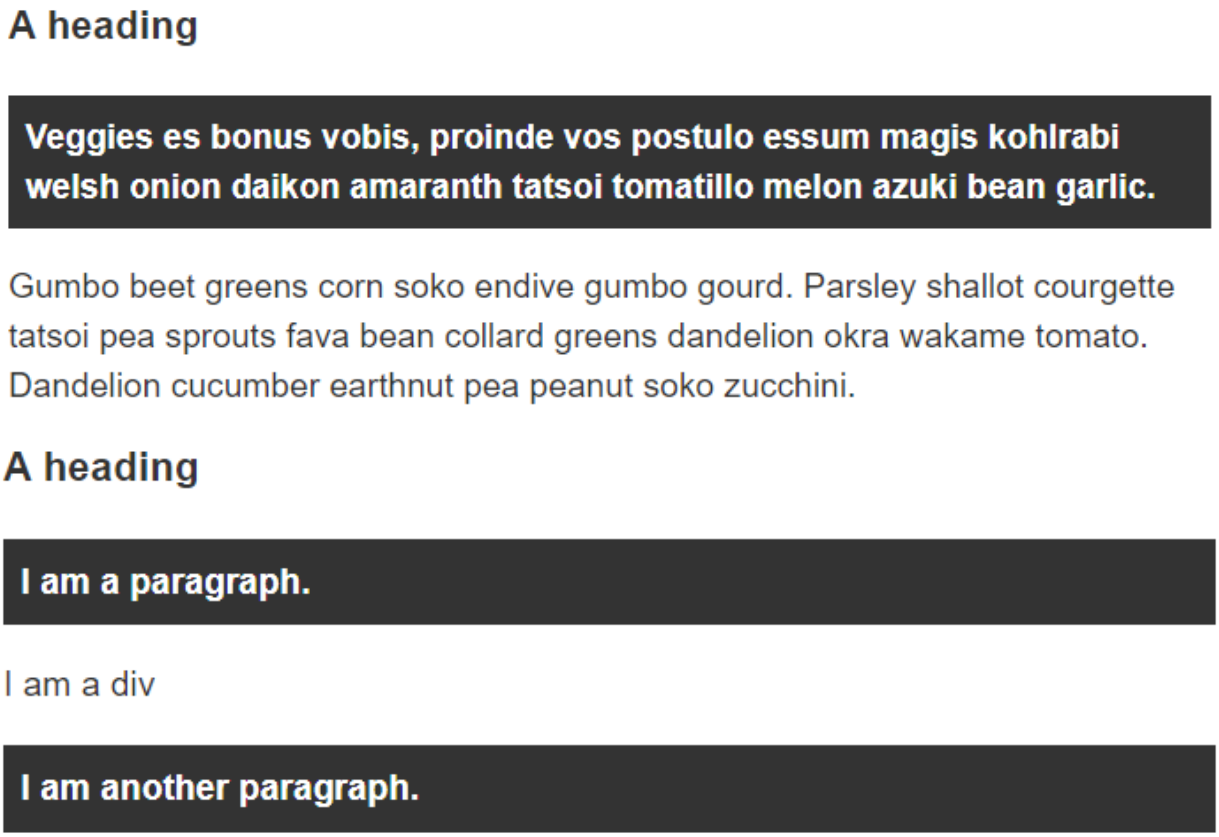


Interactive editor

```
ul > li {
    border-top: 5px solid red;
}
```

```
<ul>
    <li>Unordered item</li>
    <li>Unordered item
        <ol>
            <li>Item 1</li>
            <li>Item 2</li>
        </ol>
    </li>
</ul>
```

Практика: Самостоятельно ознакомиться с оставшимися видами комбинаторов (+ и ~) в электронном учебнике: https://developer.mozilla.org/ru/docs/Learn/CSS/Building_blocks/Selectors/Combinators . Используя комбинаторы, составить и оформить следующие блоки:



Тема 3.13

Множественные селекторы

Иногда может быть желательно иметь немного больше контроля, в том числе возможность выбрать родственный элемент, который следует непосредственно за другим родственным элементом. Соседний селектор будет выбирать только родственные элементы, непосредственно следующие за другим родственным элементом. Вместо символа тильды, как в случае с общим родственным селектором, соседний селектор использует символ плюс (+) между двумя элементами в селекторе. Опять же, первый элемент определяет, что второй элемент должен непосредственно следовать после него и быть ему родственным и у обоих элементов должен быть один и тот же родитель.

Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.14

Блочная модель

Каждый блок имеет область содержимого, в которой находится текст, дочерние элементы, изображение и т.п., и необязательные окружающие ее padding, border и margin. Размер каждой области определяется соответствующими свойствами и может быть нулевым, или, в случае margin, отрицательным.

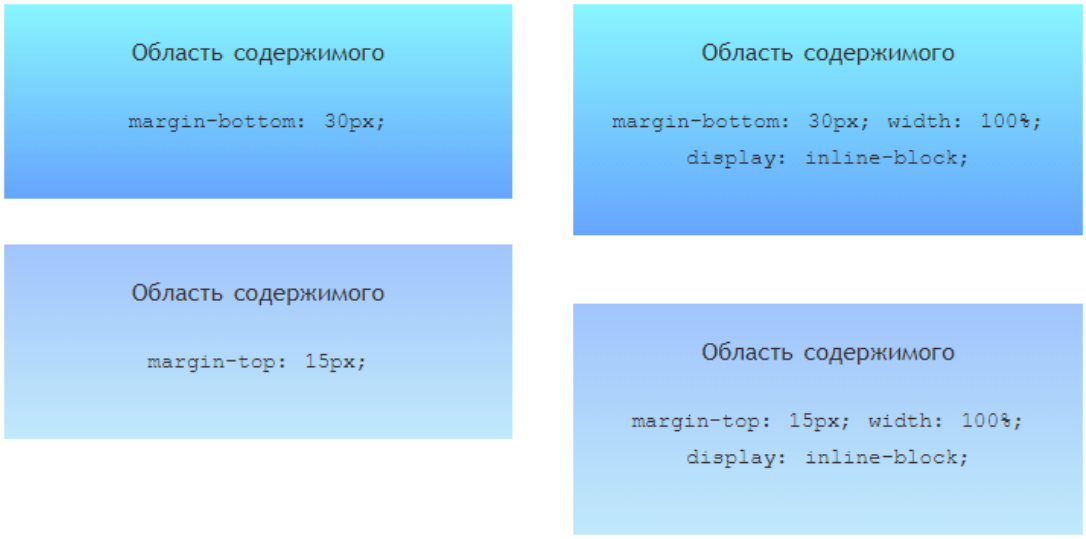
Поля, рамка и отступы могут быть разбиты на верхний, правый, нижний и левый сегменты, каждый из которых независимо управляется своим соответствующим свойством.

Фон области содержимого, полей и рамки блока определяется свойствами фона. Область рамки может быть дополнительно окрашена с помощью свойства border. Отступы элемента всегда прозрачны, что позволяет показывать фон родительского элемента.

Так как поля и отступы элемента не являются обязательными, по умолчанию их значение равно нулю. Тем не менее, некоторые браузеры добавляют этим свойствам положительные значения по умолчанию на основе своих таблиц стилей. Очистить стили браузеров для всех элементов можно при помощи универсального селектора:

```
* {
 margin: 0;
 padding: 0;
 }
```

Практика: Используя блочные элементы, оформить элементы web-страницы в соответствии с примером:



Тема 3.15

Отображение и позиционирование

CSS рассматривает макет html-документа как дерево элементов. Уникальный элемент, у которого нет родительского элемента, называется корневым элементом. Модуль CSS-позиционирование описывает, как любой из элементов может быть размещен независимо от порядка документа (т.е. извлечен из «потока»).

В CSS2 каждый элемент в дереве документа генерирует ноль или более блоков в соответствии с блочной моделью. Модуль CSS3 дополняет и расширяет схему позиционирования. Расположение этих блоков регулируется:

- размерами и типом элемента,
- схемой позиционирования (нормальный поток, обтекание и абсолютное позиционирование),
- отношениями между элементами в дереве документа,

- внешней информацией (например, размер области просмотра, внутренними размерами изображений и т.д.).

В CSS блок элемента может быть расположен в соответствии с тремя схемами позиционирования:

Нормальный поток

Нормальный поток включает блочный контекст форматирования (элементы с display block, list-item или table), строчный (встроенный) контекст форматирования (элементы с display inline, inline-block или inline-table), и относительное и «липкое» позиционирование элементов уровня блока и строки.

Обтекание

В обтекающей модели блок удаляется из нормального потока и позиционируется влево или вправо. Содержимое обтекает правую сторону элемента с float: left и левую сторону элемента с float: right.

Абсолютное позиционирование

В модели абсолютного позиционирования блок полностью удаляется из нормального потока и ему присваивается позиция относительно содержащего блока. Абсолютное позиционирование реализуется с помощью значений position: absolute; и position: fixed;.

Элементом «вне потока» может быть плавающий, абсолютно позиционированный или корневой элемент.

Тема 3.16

Абсолютное, относительное и фиксированное позиционирование

Absolute

Положение блока (и, возможно, размер) задается с помощью свойств top, right, bottom и left. Эти свойства определяют явное смещение относительно его содержащего блока. Абсолютно позиционированные блоки полностью

удаляется из нормального потока, не влияя на расположение сестринских элементов.

Отступы margin абсолютно позиционированных блоков не схлопываются.

Абсолютно позиционированный блок создает новый содержащий блок для дочерних элементов нормального потока и потомков с position: absolute;.

Содержимое абсолютно позиционированного элемента не может обтекать другие блоки. Абсолютно позиционированный блок могут скрывать содержимое другого блока (или сами могут быть скрыты), в зависимости от значения z-index перекрывающихся блоков.

Relative

Положение блока рассчитывается в соответствии с нормальным потоком. Затем блок смещается относительно его нормального положения и во всех случаях, включая элементы таблицы, не влияет на положение любых следующих блоков. Тем не менее, такое смещение может привести к перекрытию блоков, а также к появлению полосы прокрутки в случае переполнения.

Относительно позиционированный блок сохраняет свои размеры, включая разрывы строк и пространство, первоначально зарезервированное для него.

Относительно позиционированный блок создает новый содержащий блок для абсолютно позиционированных потомков.

Влияние position: relative; на элементы таблицы определяется следующим образом:

Элементы с table-row-group, table-header-group, table-footer-group ид table-row смещаются относительно их обычной позиции в таблице. Если ячейки таблицы занимают несколько строк, смещаются только ячейки начальной строки.

table-column-group, table-column не смещает соответствующий столбец и не оказывает визуального влияния.

table-caption and table-cell смещаются относительно своего нормального положения в таблице. Если ячейка таблицы охватывает несколько столбцов или строк, то она смещается целиком.

Fixed

Фиксированное позиционирование аналогично абсолютному позиционированию, с отличием в том, что для содержащим блоком устанавливается окно просмотра. Такой блок полностью удаляется из потока документа и не имеет позиции относительно какой-либо части документа. Фиксированные блоки не перемещаются при прокрутке документа. В этом отношении они похожи на фиксированные фоновые изображения.

При печати фиксированные блоки повторяются на каждой странице, содержащим блоком для них устанавливается область страницы. Блоки с фиксированным положением, которые больше области страницы, обрезаются.

Тема 3.17

Позиция Sticky

Положение блока рассчитывается в соответствии с нормальным потоком. Затем блок смещается относительно своего ближайшего предка с прокруткой или окна просмотра, если ни у одного из предков нет прокрутки.

«Липкий» блок может перекрывать другие блоки, а также создавать полосы прокрутки в случае переполнения.

«Липкий» блок сохраняет свои размеры, включая разрывы строк и пространство, первоначально зарезервированное для него.

«Липкий» блок создает новый содержащий блок для абсолютно и относительно позиционированных потомков.

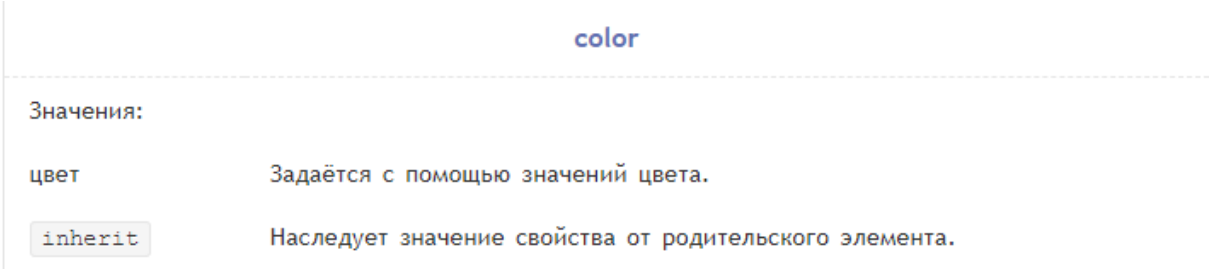
Тема 3.18

Цветовое оформление

Модуль CSS color подробно описывает значения, которые позволяют авторам определять цвета и непрозрачность html-элементов, а также значения свойства color.

Свойство color задаёт цвет шрифта с помощью различных систем цветопередачи. Свойство описывает цвет текстового содержимого элемента. Кроме того, оно используется для предоставления потенциального косвенного

значения (currentColor) для любых других свойств, которые принимают значения цвета. Свойство наследуется.



Список основных ключевых слов включает в себя следующие значения:

Название	HEX	RGB	Цвет
black	#000000	0,0,0	
silver	#C0C0C0	192,192,192	
gray	#808080	128,128,128	
white	#FFFFFF	255,255,255	
maroon	#800000	128,0,0	
red	#FF0000	255,0,0	
purple	#800080	128,0,128	
fuchsia	#FF00FF	255,0,255	
green	#008000	0,128,0	
lime	#00FF00	0,255,0	
olive	#808000	128,128,0	
yellow	#FFFF00	255,255,0	
navy	#000080	0,0,128	
blue	#0000FF	0,0,255	
teal	#008080	0,128,128	
aqua	#00FFFF	0,255,255	

Названия цветов не чувствительны к регистру.

Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.19

Работа с текстом

С помощью CSS можно определять стиль и вид текста. Аналогично тому, что используется тег <font>, задающий свойства шрифта, но стили обладают большими возможностями и позволяют сократить код HTML.

Практика: Создать репозиторий, отработать пройденный материал.

Тема 3.20

Работа с графикой

Свойство CSS background-image устанавливает одно или несколько фоновых изображений для элемента. Изображения рисуются в слоях контекстов наложения одно поверх другого. Первый слой выводится так, чтобы он был ближе всего к пользователю.

Границы border элемента затем рисуются поверх них, и background-color рисуется под ними. То, как изображения отрисовываются относительно рамки и её границ, определяется CSS-свойствами background-clip и background-origin.

Если указанное изображение не может быть нарисовано (например, когда файл, определённый указанным URI, не может быть загружен), браузеры обрабатывают его так, как если бы оно было значением none.

Промежуточная аттестация в форме проверочной работы:

Задание: Сверстать web-страницу – расписание для школы, стилизовать используя изученные технологии.

5 класс

Понедельник Вторник Среда Четверг Пятница Суббота

Понедельник

Урок	5 "А"	5 "Б"	5 "В"
1	Русский язык	Литература	История
2	Математика	Информатика	Английский язык
3	История	Математика	Информатика

Вторник

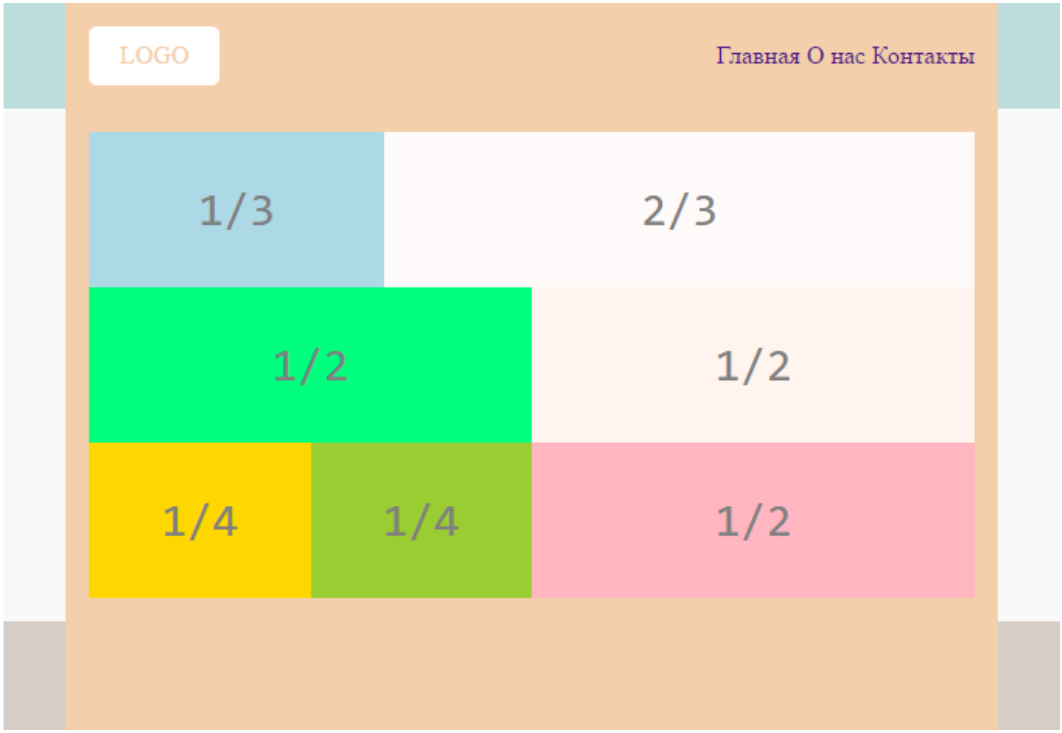
Урок	5 "А"	5 "Б"	5 "В"
1	Математика	История	Природоведение
2	Физкультура	Технология	Русский язык
3	Природоведение	Технология	Физкультура

Модуль 4. Верстка

Тема 4.1

Разметка страницы по макету

Вёрстка страницы представляет собой процесс разработки структуры html-документа, результатом которого является веб-страница. Структура веб-страницы определяется соответствующими html-тегами. Теги — прямоугольные блоки-контейнеры для содержимого — не отображаются в окне браузера. Они сообщают браузеру о типе контента, а браузер на основании этой информации выводит на экран их содержимое — текст или медиа-файлы.



Тема 4.2

Экспортирование графики

Чтобы добавить изображения на страницу мы используем строчный элемент <img>. Элемент <img> относится к самостоятельным или к пустым элементам; это означает, что он не содержит какого-либо контента и существует в виде единственного тега. Для работы <img> должен быть включен атрибут src со значением, указывающим исходник изображения. Значение атрибута src это URL, обычно относительно сервера, на котором размещён сайт.

Наряду с атрибутом src должен быть применён атрибут alt (альтернативный текст), который описывает содержимое изображения. Значение атрибута alt собирают поисковые системы и вспомогательные технологии, это помогает им донести назначение изображения. Альтернативный текст будет отображаться вместо изображения, если по какой-то причине изображение не доступно.

Тема 4.3

Статичная верстка

Статичная верстка - когда страничка имеет строго зафиксированные размеры, которые не зависят от параметров монитора пользователя. Преимущество таких сайтов – их стабильная работа во всех браузерах, простота.

Тема 4.4

Резиновая верстка

Резиновая верстка – это когда сайт и его отдельные блоки могут менять размеры в рамках определённого предела. То есть, если вы будете просматривать страницу на небольшом экране, текстовый блок уменьшится по ширине. Если же его открыть на большом экране, блоки растянутся. Такие страницы более удобны для пользователей, однако для того чтобы они стабильно работали в разных браузерах необходимо, чтобы над версткой сайта работал настоящий профессионал.

Тема 4.5

Адаптивная верстка

Адаптивная верстка применяется в том случае, если необходимо обеспечить его стабильную работу и привлекательный внешний вид на широком спектре устройств – от стационарного компьютера или лэптопа до небольших планшетов и смартфонов. Такие страницы выглядят по-разному, в зависимости

от размеров экрана – у них меняются не только размеры блоков, но и их содержимое, может регулироваться шрифт, картинки и т.д.

Тема 4.5
Введение в гибкую сетку (Flexbox)

Долгое время единственными надёжными инструментами CSS вёрстки были такие способы как Float (обтекание) и позиционирование. С их помощью сложно или невозможно достичь следующих простых требований к макету:

- Вертикального выравнивания блока внутри родителя.
- Оформления всех детей контейнера так, чтобы они распределили между собой доступную ширину/высоту, независимо от того, сколько ширины/высоты доступно.
- Сделать все колонки в макете одинаковой высоты, даже если наполнение в них различно.

4.4 Календарный учебный график (порядок освоения разделов)

Период обучения (недели)*	Наименование разделов
1 неделя 3 неделя 6 неделя 9 неделя 10 неделя	Модуль 1. Веб-разработка Модуль 2. Основы HTML Модуль 3. Основы CSS Модуль 4. Верстка Модуль 5. Публикация web-страницы
*Точный порядок реализации разделов обучения определяется в расписании занятий.	

5. Организационно-педагогические условия реализации программы

5.1 Материально-технические условия реализации программы

Мастерская: 4. Веб-дизайн и разработка	54.02.01 Дизайн (по отраслям)	- Компьютер i7-8700/Система охлаждения/H310M-R R2.0/DIMM 2x16GB DDR4/1000GB 7200/SSD 256GB/GTX 1650Ti 4GB/Win 10 Professional; - Сервер i7-9700/Система охлаждения/B365M-K/DIMM 16GB DDR; - Монитор Samsung F24T450FQI 23.8" LCD IPS LED ; - Клавиатура Microsoft Wired 600; - Мышь Microsoft 1850 Black; - Ноутбук MSI GL75 Leopard 10SDK-250RU Core i7 10750H/16Gb/SSD512Gb/nVidia GeForce GTX 1660 Ti 6Gb/17.3"/IPS/FHD (1920x1080)/Windows 10/black/WiFi/BT/Cam; - Кронштейн для 2-х мониторов ONKRON/ 10-32" - 25 шт; - Аудиосистема комплект: volta 15 А 500 Вт\ MP3 \USB / SD card / BlueTooth– 2 штуки. VOLTA MX-22- 1 штука Микрофонная радиосистема U-2 Комплект необходимой коммутации, стойки; - Веб-камера Logitech Webcam C505e Black; - Монохромное МФУ формата А4 SHARP MXB350WEE;

5.2 Учебно-методическое обеспечение программы

- Печатные раздаточные материалы для слушателей;
- электронные ресурсы и т.д.

5. Кадровые условия реализации программы

Реализация обеспечивается педагогическими кадрами, имеющими высшее образование, Зпрофилю деятельности.

6. Оценка качества освоения программы

Итоговая аттестация проводится в форме экзамена.

Включает в себя:

1. Проверку теоретических знаний;

2. Практическая работа.

Оцениваемые знания, умения	**Показатели оценки результата**
Должен знать: *Методы и приемы отладки программного кода;* *Типы и форматы сообщений об ошибках, предупреждениях;* *Современные компиляторы, отладчики и оптимизаторы программного кода;* *Сообщения о состоянии аппаратных средств;* *Регламент использования системы контроля версий;* *Особенности отображения элементов ИР в различных браузерах;* *Особенности отображения ИР в размерах рабочего пространства устройств;* *Методы повышения читаемости программного кода;* *Отраслевая нормативная техническая документация;* *Синтаксис выбранного языка программирования, особенности программирования на этом языке;*	Знает: *Принципы проектирования web-страниц;* *Принципы оформления web-страниц;* *Технологию статичной, резиновой и адаптивной верстки;* *Методы работы с системой контроля версий;* *Способы передачи данных в сети Интернет;* *Способы развертывания web-страницы в глобальной сети.*
Должен уметь: *Выявлять ошибки в программном коде;* *Применять методы и приемы отладки программного кода;* *Интерпретировать сообщения об ошибках, предупреждения, записи технологических журналов;* *Применять современные компиляторы, отладчики и оптимизаторы программного кода;*	Умеет: *Пользоваться технической документацией;* *Проектировать web-страницы;* *Верстать web-страницы с использованием языка разметки HTML;* *Стилизовать web-страницы с использованием CSS;* *Работать с системой контроля версий;* *Разворачивать web-страницы в глобальной сети*

Применять нормативные документы, определяющие требования к оформлению страниц ИР; *Применять вспомогательные инструментальные программные средства для обработки исходного текста программного кода;* *Применять нормативные документы, определяющие требования к оформлению страниц ИР;* *Применять специализированное программное обеспечение для верстки страниц ИР;* *Определять возможности отображения web-страниц в размерах рабочего пространства устройств для разных видов дизайн-макетов;* *Применять специализированное программное обеспечение для верстки страниц ИР;* *Использовать язык разметки страниц ИР ;* *Применять выбранные языки программирования для написания программного кода;* *Использовать выбранную среду программирования и средства системы управления базами данных;* *Использовать возможности имеющейся программной архитектуры ИР*	

6.1. Проверка теоретических знаний, сформированных умений

Задание для экзаменующегося

Проверка теоретических знаний

Вопрос 1: Что представляет собой процесс вёрстки сайта?

Создание программного кода, позволяющего отображать различные графические элементы сайта, интерактивные компоненты, и осуществлять навигацию.

Процесс создания графического макета всех страниц сайта с проработкой и описанием всех состояний элементов сайта.

Начальный этап создания сайта, на котором определяется расположение каждого из элементов сайта и создаётся первоначальный макет, на который в последствии накладываются графические элементы.

Процесс формирования на основе разработанного графического макета гипертекстового файла, позволяющего отображать все элементы сайта в браузерах в соответствии с установленными стандартами и требованиями.

Вопрос 2: Организация «Консо́рциум Всеми́рной паути́ны» (англ. World Wide Web Consortium, W3C) занимается разработкой стандартов, имеющих основополагающее значение для создания сайтов. Что содержат в себе эти стандарты?

Требования к функционированию программной части сайтов.

Требования к оформлению кода для корректного отображения на большинстве современных браузеров.

Требования к визуальному оформлению элементов (шрифтов, графики, форм) для оптимизации восприятия пользователем информации с сайта.

Указания порядка соблюдения авторских прав на материалы, размещённые на сайте.

Вопрос 3: Для какого из перечисленных ниже сайтов оптимальным решением будет являться «резиновая» вёрстка:

Сайт дизайнерской студии с чётко отрисованными элементами, среди которых имеется фотоколлаж. Сайт содержит незначительное количество текста.

Flash-сайт, презентация нового автомобиля престижной марки.

Новостной сайт с минимальным количеством графики и большим количеством текста.

Вопрос 4: Почему при вёрстке сайта необходимо уделять особое внимания правильности его отображения в браузере Internet Explorer.

Данный браузер использует наибольшее количество пользователей по всему миру.

Данный браузер строже всех следует стандартам отображения сайтов, таким образом, корректное отображения сайта в данном браузере является признаком высокой квалификации верстальщика.

Данный браузер является весьма перспективным и завоёвывает всё больше пользователей по всему миру.

Компания Microsoft (разработчик браузера) ведёт «чёрные списки» сайтов, некорректно отображающихся в Internet Explorer; подобные сайты блокируются.

Вопрос 5: Для чего используется CSS?

Для задания цветов, шрифтов, расположения и других аспектов представления документа.

Для создания разметки веб-страницы – указания расположения и параметров отображения элементов на странице.

Для создания анимации и подвижных элементов на сайте.

Для добавления комментариев к основному коду страницы.

Вопрос 6: Какое из приведённых преимуществ относится к «статичной» вёрстке?

Все элементы сайта располагаются строго на своих местах – элементы сайта не «расползаются» при слишком больших разрешениях и не «сбиваются в кучу» при слишком маленьких.

Не используется графическая анимация – упрощается восприятие сайта пользователем.

Все элементы дизайна повторяются на каждой из страниц, меняется только текстовое наполнение (контент) страниц – упрощается ориентация пользователя по сайту.

Используются стили, позволяющие разнообразить оформление сайта.

Вопрос 7: Какие из приведённых браузеров построены на «движке» WebKit?

Mozilla, Chrome, Internet Explorer.

Safari, Firefox.

Opera, Internet Explorer.

Chrome, Safari, Konqeror.

Вопрос 8: Назовите главное преимущество использования стандарта Unicode при вёрстке страницы:

Данный стандарт позволяет корректно отображать шрифт вне зависимости от используемого пользователем браузера.

Данный стандарт даёт верстальщику широкий выбор разнообразных шрифтов, среди которых можно выбрать подходящий для дизайна конкретной страницы.

Данный стандарт поддерживается всеми операционными системами.

Данный стандарт позволяет представить знаки практически всех письменных языков.

Вопрос 9: Какую функцию выполняет тег META?

Предназначен для хранения содержания веб-страницы (контента), отображаемого в окне браузера

Содержит словесное описание сайта, которое используется браузерами и поисковыми системами.

Определяет теги, которые используются для хранения информации, предназначенной для браузеров и поисковых систем.

Задаёт набор тегов, использующихся на конкретной странице.

Вопрос 10: Для решения какой из перечисленных задач будет уместно воспользоваться JavaScript?

Создание веб-стриницы с множеством полупрозрачных элементов, перекрывающих друг друга.

Создание на странице фотогаллереи с возможностью перелистывать фотографии.

Создание страницы с перечнем последних тем, обсуждавшихся на форме (сообщения форума хранятся в SQL)

Создание страниц с динамически изменяющимся содержимым.

Вопрос 11: Назовите основное отличие XHTML от HTML?

Все элементы должны быть закрыты. Теги, которые не имеют закрывающего тега должны иметь на конце “/”.

Комментарий может быть размещен в любом месте дерева.

XHTML позволяет добавлять Java-апплеты на страницы.

Регистр, в котором набрано имя элемента и имена атрибутов, значения не имеет.

Вопрос 12: Что представляет собой SEO-оптимизация?

Процесс оптимизации HTML-кода, текстового наполнения и структуры сайта , направленный на улучшение внешнего вида элементов сайта с целью повышения удобства восприятия информации и создания позитивного эмоционального фона при взаимодействии с сайтом.

Процесс корректировки HTML-кода, текстового наполнения, структуры сайта, контроль внешних факторов для соответствия требованиям алгоритма поисковых систем, с целью поднятия позиции сайта в результатах поиска в поисковых системах по определенным запросам пользователей.

Процесс оптимизации работы поискового механизма сайта с целью повышения релевантности поисковой выдачи, с учётом статистики запросов и семантики вводимых пользователями слов.

Процесс создания программного кода, позволяющего наиболее быстро выполнять необходимые для функционирования сайта операции.

Вопрос 13: Перед вами встала задача создать сайт о новинках кино. Для каждого из кинофильмов требуется сделать страницу обсуждения с условием, что новые комментарии будут добавляться на страницу без её перезагрузки. Какая из технологий позволит реализовать это?

XML
CSS
AJAX
HTML

Вопрос 14: Какой из приведённых ниже тегов не требует «закрытия»?

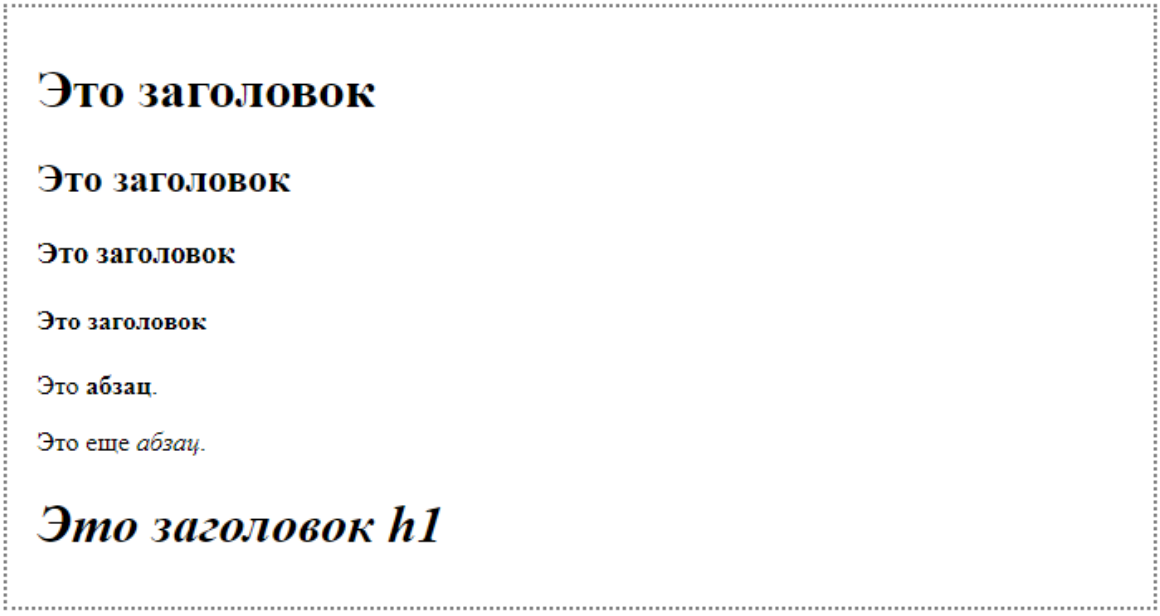
strong
h1
br
blockquote

Вопрос 15: Укажите правильный вариант написания элементарной программы на JavaScript (скрипт, выводящий модальное окно с классической надписью «Hello, World!» внутри браузера):

alert('Hello, World!');
echo('Hello, World!');
print 'Hello, World!';
text | 'Hello, World!';

Выполнение практического задания

1) Повторите страницу по данному по образцу:



2) Повторите страницу по данному по образцу:

ячейка 1	ячейка 2	ячейка 3
ячейка 4	ячейка 5	ячейка 6
ячейка 7	ячейка 8	ячейка 9

3) Повторите страницу по данному по образцу:

номер	тег	значение
1	p	абзац
2	a	ссылка
3	b	жирный текст
4	i	курсив
5	table	таблица
6	tr	ряд таблицы
7	td	ячейка таблицы
8	th	заголовок таблицы

4) Повторите страницу по данному по образцу:

Что такое CMS

CMS - «система управления контентом» (**движок**) – написанная PHP-программистами основа для сайта, с помощью которой вы сможете управлять сайтом (добавлять контент, менять пункты меню и т.п.) не зная HTML и CSS.

Однако, для того чтобы сделать сайт с помощью **CMS** *потребуются услуги* и программиста, и дизайнера, и верстальщика. И капиталовложения.

Какие бывают cms

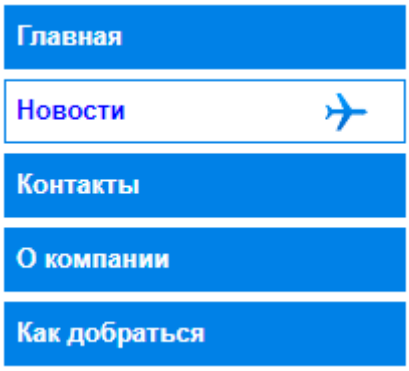
Бывают различные системы управления контентом: для интернет-магазинов, для блогов, для форумов и т.д.

Примеры cms

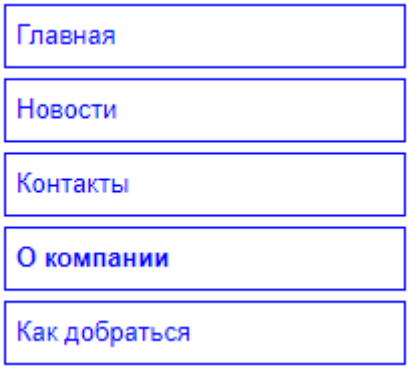
Примеры популярных CMS: Joomla, WordPress (для блогов), PhpBB (для форумов).

CMS-ки бывают *платные* и *бесплатные*.

5) Повторите страницу по данному по образцу:

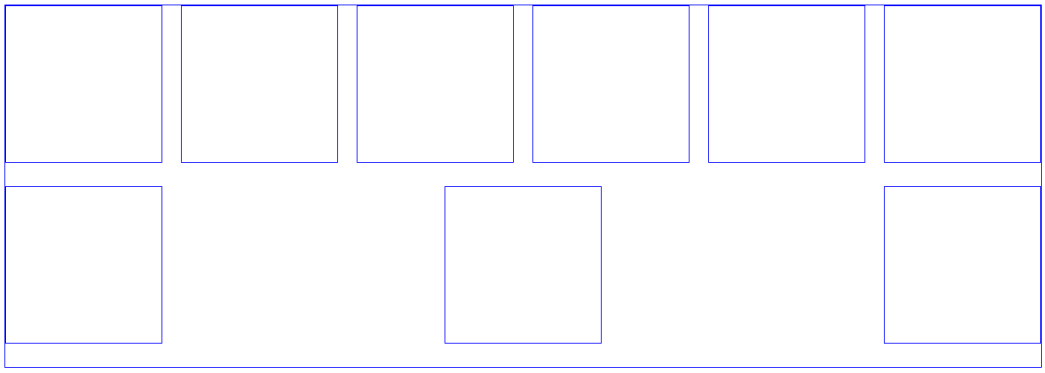


6) Повторите страницу по данному по образцу:

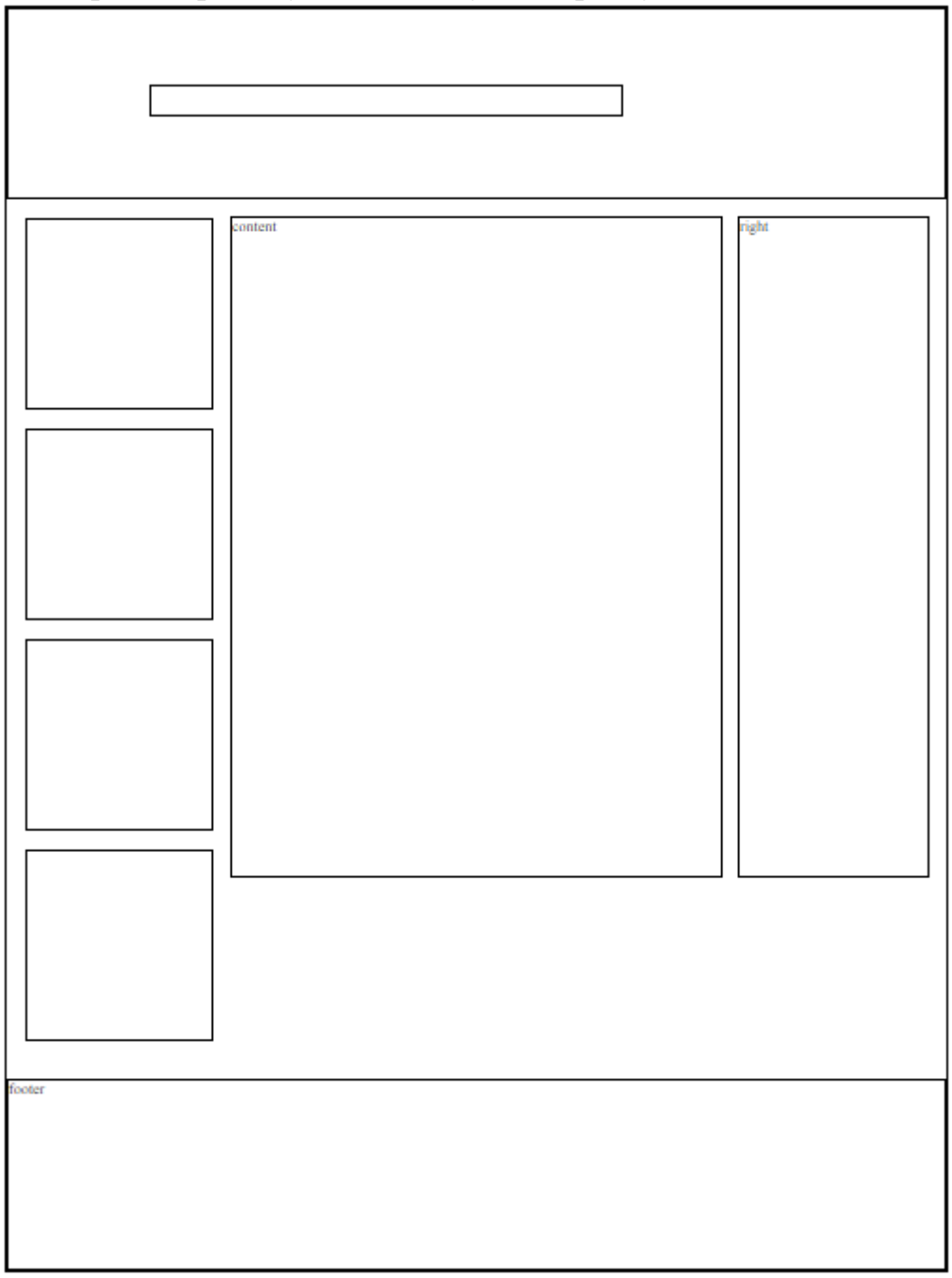


7) Повторите страницу по данному по образцу:

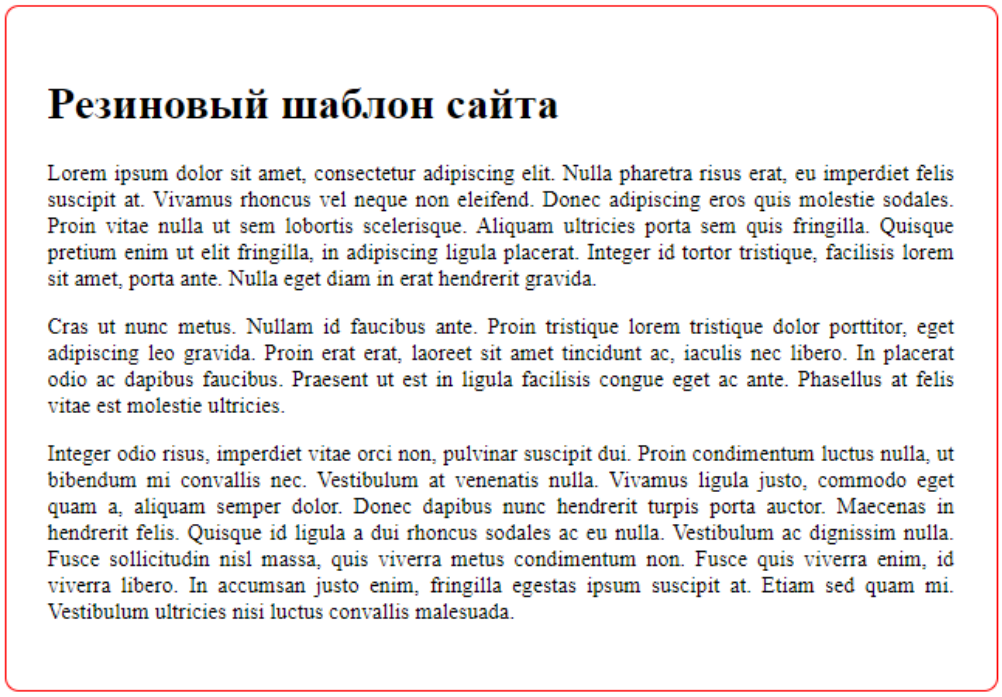
8) Повторите страницу по данному по образцу:



9) Повторите страницу по данному по образцу:



10) Повторите страницу по данному по образцу:



Резиновый шаблон сайта

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla pharetra risus erat, eu imperdiet felis suscipit at. Vivamus rhoncus vel neque non eleifend. Donec adipiscing eros quis molestie sodales. Proin vitae nulla ut sem lobortis scelerisque. Aliquam ultricies porta sem quis fringilla. Quisque pretium enim ut elit fringilla, in adipiscing ligula placerat. Integer id tortor tristique, facilisis lorem sit amet, porta ante. Nulla eget diam in erat hendrerit gravida.

Cras ut nunc metus. Nullam id faucibus ante. Proin tristique lorem tristique dolor porttitor, eget adipiscing leo gravida. Proin erat erat, laoreet sit amet tincidunt ac, iaculis nec libero. In placerat odio ac dapibus faucibus. Praesent ut est in ligula facilisis congue eget ac ante. Phasellus at felis vitae est molestie ultricies.

Integer odio risus, imperdiet vitae orci non, pulvinar suscipit dui. Proin condimentum luctus nulla, ut bibendum mi convallis nec. Vestibulum at venenatis nulla. Vivamus ligula justo, commodo eget quam a, aliquam semper dolor. Donec dapibus nunc hendrerit turpis porta auctor. Maecenas in hendrerit felis. Quisque id ligula a dui rhoncus sodales ac eu nulla. Vestibulum ac dignissim nulla. Fusce sollicitudin nisl massa, quis viverra metus condimentum non. Fusce quis viverra enim, id viverra libero. In accumsan justo enim, fringilla egestas ipsum suscipit at. Etiam sed quam mi. Vestibulum ultricies nisi luctus convallis malesuada.

Пакет экзаменатора

А) Условия

Количество вариантов задания для экзаменующихся/сдающих КЭ– 1

Оборудование: Бланк с заданиями для выполнения тестирования, ручка.

Экзаменационная/зачетная ведомость.

Б) Критерии оценки

Программа считается освоенной, если успешно выполнены все промежуточные зачетные мероприятия (зачеты и экзамены по 5-ти бальной оценке) и успешно пройдена итоговая аттестация.

Шкала оценки образовательных достижений

Процент результативности (правильных ответов)	**Оценка уровня подготовки**	
	балл (оценка)	*вербальный аналог*
90 ÷ 100	*30 - 35 (5)*	*отлично*
80 ÷ 89	*25 -29 (4)*	*хорошо*
70 ÷ 79	*20 - 24 (3)*	*удовлетворительно*
менее 70	*19 (2)*	*неудовлетворительно*

Критерии оценки:

• оценка «отлично» выставляется слушателю, если ответы на тест составили 90-100%.

• оценка «хорошо» выставляется слушателю, если ответы на тест составили 80-89%.

• оценка «удовлетворительно» выставляется слушателю, если ответы на тест составили 70-79%.

• оценка «неудовлетворительно» выставляется слушателю, если ответы на тест составили менее 70%.