

Лабораторная работа № 9

Тема работы: «Разработка таблиц стилей для HTML документа»

1. Цель работы

Формирование умения создавать составные селекторы и выбирать необходимые свойства CSS для оформления документа в соответствии с макетом.

2. Задание

Сверстать HTML-страницу используя средства HTML и CSS.

3. Оснащение работы

ПК, редактор кода, браузер.

1. Основные теоретические сведения

Расположение элементов на HTML-странице можно реализовать с помощью следующих свойств:

position
float
display

Свойство position позволяет точно задать новое местоположение блока относительно того места, где он находился бы в нормальном потоке документа. По умолчанию все элементы располагаются последовательно один за другим в том порядке, в котором они определены в структуре HTML-документа. Свойство не наследуется.

Значения свойства position представлены в таблице 9.1

Таблица 9.1 – Значения свойства position

Значение	Свойство
static	Значение по умолчанию, означает отсутствие позиционирования. Элементы отображаются последовательно один за другим в том порядке, в котором они определены в HTML-документе. Используется для очистки любого другого значения позиционирования
relative	Относительно позиционированный элемент сдвигается со своего обычного места в разных направлениях относительно границ родительского контейнера, а пространство, которое он занимал, не исчезает. При этом такой элемент может перекрывать другое содержимое на странице. Если для относительно позиционированного элемента одновременно задать свойства top и bottom или left и right, то в первом случае сработает только top, во втором — left. Относительное позиционирование позволяет задавать z-index для элемента, а также абсолютно позиционировать

	дочерние элементы внутри блока.
absolute	Абсолютно позиционированный элемент полностью удаляется из потока документа и позиционируется относительно границ его блока-контейнера (другого элемента или окна браузера). Блок-контейнер для абсолютно позиционированного элемента — ближайший элемент-предок, значение свойства position которого не равно static. Местоположение краёв элемента определяется с помощью свойств смещения. Пространство, которое занимал такой элемент, схлопывается, как будто элемента не существовало на странице. Абсолютно позиционированный элемент может перекрывать другие элементы или быть перекрытым ими (за счёт свойства z-index). Любой абсолютно позиционированный элемент генерирует блок, то есть принимает значение display: block;.
fixed	Фиксирует элемент в указанном месте страницы. Блоком-контейнером фиксированного элемента является окно просмотра, то есть элемент всегда фиксируется относительно окна браузера и не меняет своего положения во время прокрутки страницы. Сам элемент при этом полностью удаляется из основного потока документа и создаётся в новом потоке документа.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

Свойства описывают смещение относительно ближайшей стороны блока-контейнера. Задаются для элементов, для которых значение свойства position не равно static. Могут принимать как положительные, так и отрицательные значения. Не наследуются.

Для свойства top положительные значения перемещают верхний край позиционируемого элемента ниже, а отрицательные — выше верхнего края его блока-контейнера. Для свойства left положительные значения сдвигают край позиционируемого элемента вправо, а отрицательные значения — влево. То есть, положительные значения смещают элемент внутрь блока-контейнера, а отрицательные — за его пределы.

Позиционирование внутри элемента, пример представлен на рисунке 9.1. Для блока-контейнера абсолютно позиционированного элемента задаётся свойство position: relative без смещений. Это позволяет позиционировать элементы внутри элемента-контейнера.

```

<div class="wrap">
  <div class="white">
    </div>
  </div>
  .wrap {

```

```
padding: 10px;
height: 150px;
position: relative;
background: #e7e6d4;
text-align: right;
border: 3px dashed #645a4e;
}
.white {
position: absolute;
width: 200px;
top: 10px;
left: 10px;
padding: 10px;
background: #ffffff;
border: 3px dashed #312a22;}
```

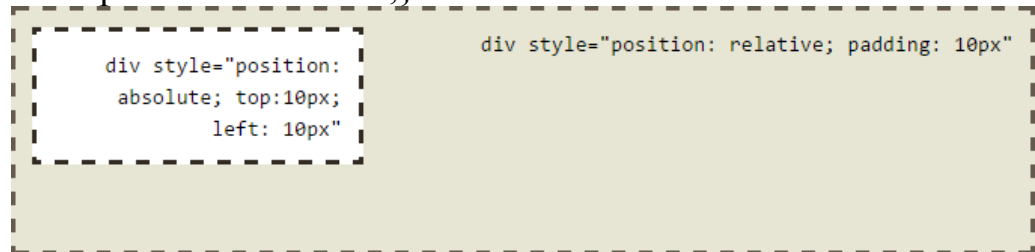


Рисунок 9.1 – Абсолютное относительное позиционирование

Свободное перемещение элементов, пример представлен на рисунке 9.2

В обычном порядке блочные элементы отображаются, начиная с верхнего края окна браузера по направлению к нижнему краю. Свойство float позволяет перемещать любой элемент, выравнивая его по левому или правому краю веб-страницы или содержащего его элемента-контейнера. При этом остальные блочные элементы будут его игнорировать, а строчные элементы будут смещаться вправо или влево, освобождая для него пространство и обтекая его.

Плавающий блок принимает размеры своего содержимого с учетом внутренних отступов и рамок. Верхние и нижние отступы margin плавающих элементов не схлопываются. Свойство float применяется как к блочным элементам, так и к строчным элементам.

Левый или правый внешний край перемещаемого элемента, в отличие от позиционированных элементов, не может располагаться левее (или правее) внутреннего края его блока-контейнера, т.е. выходить за его границы. При этом, если для блока-контейнера заданы внутренние отступы, то плавающий блок будет отстоять от края блока-контейнера на заданное расстояние.

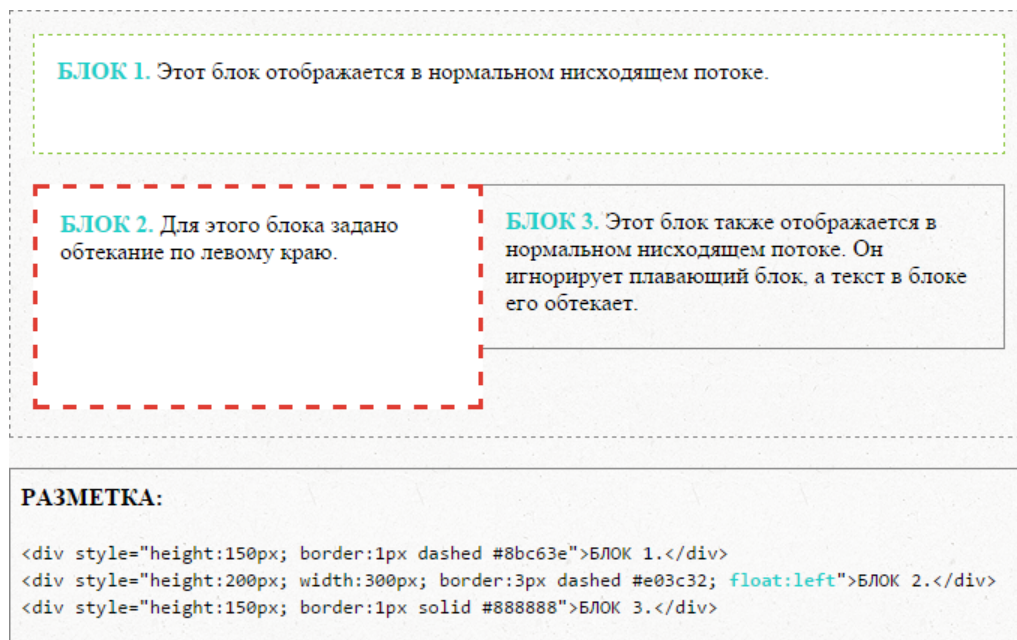


Рисунок 9.2 – Свободное перемещение элементов

Свойство автоматически изменяет вычисляемое (отображаемое в браузере) значение свойства `display` на `display: block` для следующих значений: `inline`, `inline-block`, `table-row`, `table-row-group`, `table-column`, `table-column-group`, `table-cell`, `table-caption`, `table-header-group`, `table-footer-group`. Значение `inline-table` меняет на `table`.

Свойство не оказывает никакого влияния на элементы с `display: flex` и `display: inline-flex`.

При использовании свойства `float` для блочных элементов обязательно задавать ширину. Тем самым браузер создаст место для другого содержимого. Но если совокупная ширина всех столбцов окажется больше доступного места, то последний элемент спустится вниз. Значения свойства `float` представлены в таблице 9.2.

Значение	Описание
<code>none</code>	Значение по умолчанию. Также отменяет любое перемещение для элемента из группы элементов, для которых уже установлено обтекание.
<code>left</code>	Элемент изымается из нормального потока элементов и позиционируется по левому краю блока-контейнера.
<code>right</code>	Элемент позиционируется по правому краю блока-контейнера
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

При этом вертикальные отступы `margin` обтекаемых элементов не схлопываются с отступами соседних элементов, в отличие от обычных блочных элементов.

CSS flexbox (Flexible Box Layout Module) — модуль макета гибкого контейнера — представляет собой способ компоновки элементов.

Flexbox состоит из flex-контейнера — родительского контейнера и flex-элементов — дочерних блоков. Пример визуализации представлен на рисунке 9.3. Дочерние элементы могут выстраиваться в строку или столбик, а оставшееся свободное пространство распределяется между ними различными способами.

В основе flexbox лежит идея оси. Flexbox является инструментом двумерной компоновки и использует для работы две оси — горизонтальную (главную ось) и поперечную.

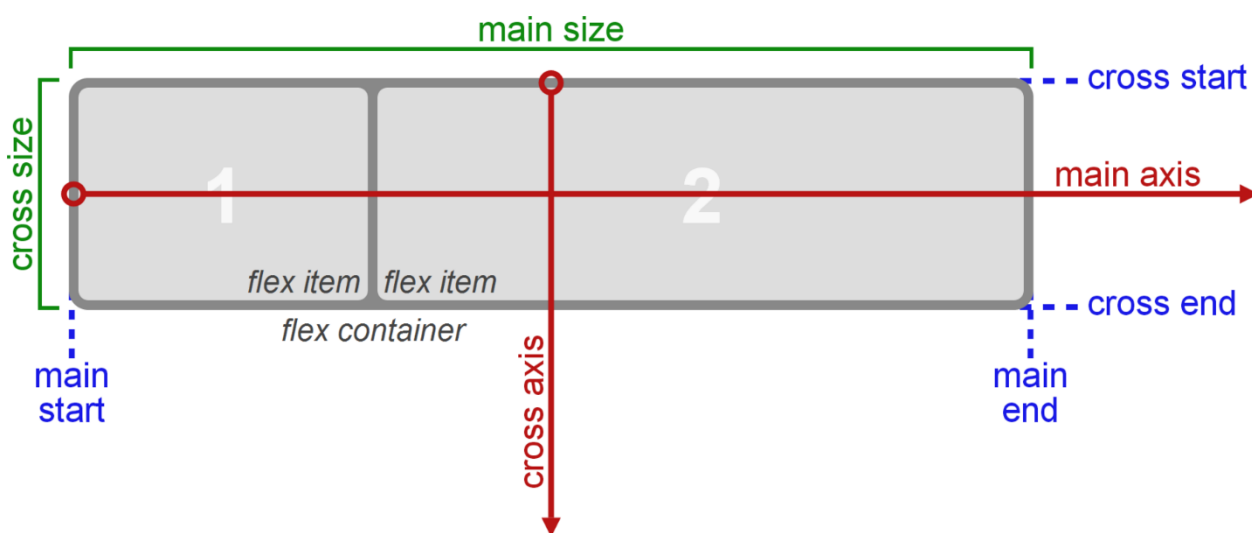


Рисунок 9.3 – Модель flexbox

Модель flexbox-разметки связана с определенным значением CSS-свойства `display` родительского `html`-элемента, содержащего внутри себя дочерние блоки. Для управления элементами с помощью этой модели нужно установить свойство `display` следующим образом:

```
.flex-container {
display: -webkit-flex;
display: flex; /*отображает контейнер как блочный элемент*/
}

.flex-container {
display: -webkit-inline-flex;
display: inline-flex; /*отображает контейнер как строчный элемент*/
}
```

После установки данных значений свойства каждый дочерний элемент автоматически становится flex-элементом, выстраиваясь в ряд (вдоль главной оси) колонками одинаковой высоты, равной высоте блока-контейнера. При этом блочные и строчные дочерние элементы ведут себя одинаково, т.е. ширина блоков равна ширине их содержимого с учетом внутренних полей и рамок элемента.

Выравнивание элементов по горизонтали `justify-content`, значения описаны в таблице 9.3.

Свойство выравнивает flex-элементы по ширине flex-контейнера, распределяя оставшееся свободное пространство, незанятое flex-элементами.

Для выравнивания элементов по вертикали используется свойство align-content. Свойство не наследуется.

Таблица 9.3 – Значения свойства justify-content

Значение	Описание
flex-start	Значение по умолчанию. Flex-элементы позиционируются от начала flex-контейнера.
flex-end	Flex-элементы позиционируются относительно правой границы flex-контейнера.
center	Flex-элементы выравниваются по центру flex-контейнера.
space-between	Flex-элементы выравниваются по главной оси, свободное место между ними распределяется следующим образом: первый блок располагается в начале flex-контейнера, последний блок – в конце, все остальные блоки равномерно распределены в оставшемся пространстве, а свободное пространство равномерно распределяется между элементами.
space-around	Flex-элементы выравниваются по главной оси, а свободное место делится поровну, добавляя отступы справа и слева.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

Синтаксис

```
.flex-container {  
  display: -webkit-flex;  
  -webkit-justify-content: flex-start;  
  display: flex;  
  justify-content: flex-start;  
}
```

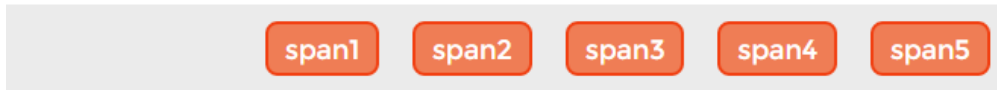
Пример визуализации свойства justify-content представлен на рисунке

9.4.

`justify-content: flex-start;`



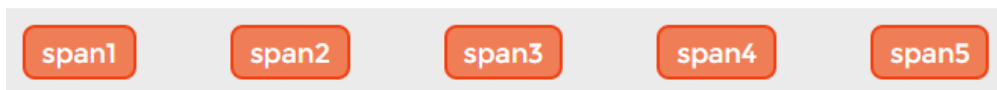
`justify-content: flex-end;`



`justify-content: center;`



`justify-content: space-between;`



`justify-content: space-around;`



Рисунок 9.4 – Выравнивание элементов свойства `justify-content`

Выравнивание элементов по вертикали `align-items`, значения свойства представлены в таблице 9.4.

Таблица 9.4 – Значения свойства `align-items`

Значение	Описание
<code>stretch</code>	Значение по умолчанию. Flex-элементы растягиваются, занимая все пространство по высоте.
<code>flex-start</code>	Flex-элементы выравниваются по левому краю flex-контейнера относительно верхнего края блока-контейнера
<code>flex-end</code>	Flex-элементы выравниваются по левому краю flex-контейнера относительно нижнего края блока-контейнера.
<code>center</code>	Flex-элементы выравниваются по центру flex-контейнера.
<code>baseline</code>	Flex-элементы выравниваются по базовой линии.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

Свойство выравнивает flex-элементы, в том числе и анонимные flex-элементы по перпендикулярной оси (по высоте). Не наследуется.

Пример визуализации свойства `align-items` представлен на рисунке 9.5.

```
align-items: stretch;
```



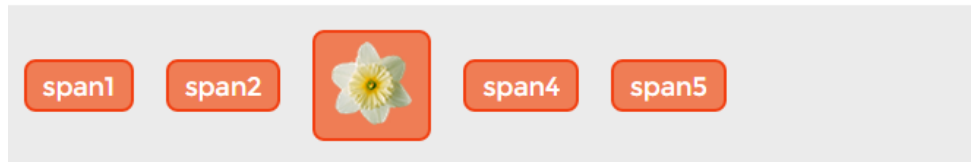
```
align-items: flex-start;
```



```
align-items: flex-end;
```



```
align-items: center;
```



```
align-items: baseline;
```



Рисунок 9.5 – Выравнивание элементов свойства align-items

Синтаксис

```
.flex-container {  
  display: -webkit-flex;  
  -webkit-align-items: flex-start;  
  display: flex;  
  align-items: flex-start;  
}
```

Направление главной оси `flex-direction`, значения свойства представлены в таблице 9.5.

Свойство определяет, каким образом flex-элементы укладываются во flex-контейнере, задавая направление главной оси flex-контейнера. Они могут располагаться в двух главных направлениях — горизонтально, как строки или вертикально, как колонки. Главная ось по умолчанию идет слева направо. Поперечная — сверху вниз. Свойство не наследуется.

Значение	Описание
row	Значение по умолчанию, слева направо (в rtl справа налево). Flex-

	элементы выкладываются в строку. Начало (main-start) и конец (main-end) направления главной оси соответствуют началу (inline-start) и концу (inline-end) инлайн оси (inline-axis).
row-reverse	Направление справа налево (в rtl слева направо). Flex-элементы выкладываются в строку относительно правого края контейнера (в rtl — левого).
column	Направление сверху вниз. Flex-элементы выкладываются в колонку.
column-reverse	Колонка с элементами в обратном порядке, снизу вверх.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

Пример визуализации свойства flex-direction представлен на рисунке 9.6.

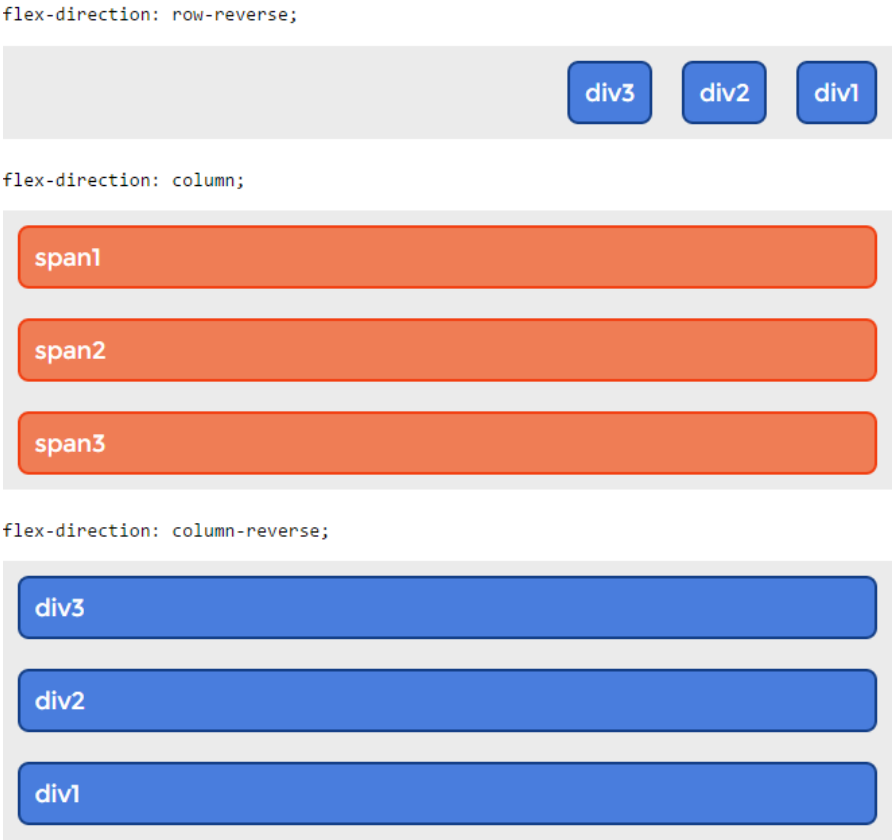


Рисунок 9.6 – Выравнивание элементов свойства flex-direction

Синтаксис

```
.flex-container {
display: -webkit-flex;
-webkit-flex-direction: row-reverse;
display: flex;
flex-direction: row-reverse;
}
```

Многострочность элементов flex-wrap, значения свойства

представлены в таблице 9.6

Таблица 9.6 – Значения свойства flex-wrap

Значение	Описание
nowrap	Значение по умолчанию. Flex-элементы не переносятся, а располагаются в одну линию слева направо (в rtl справа налево).
wrap	Flex-элементы переносятся, располагаясь в несколько горизонтальных рядов (если не помещаются в один ряд) в направлении слева направо (в rtl справа налево).
wrap-reverse	Flex-элементы переносятся, располагаясь в обратном порядке слева-направо, при этом перенос происходит снизу вверх.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

Свойство управляет тем, как flex-контейнер будет выкладывать flex-элементы — в одну строку или в несколько, и направлением, в котором будут укладываться новые строки. По умолчанию flex-элементы укладываются в одну строку. При переполнении контейнера их содержимое будет выходить за границы flex-элементов. Не наследуется.

Синтаксис

```
.flex-container {
display: -webkit-flex;
-webkit-flex-wrap: wrap;
display: flex;
flex-wrap: wrap;
}
```

Пример визуализации свойства flex-wrap представлен на рисунке 9.7.

`flex-wrap: wrap;`



`flex-wrap: wrap-reverse;`



Рисунок 9.7 – Выравнивание элементов свойства flex-wrap

CSS Grid Layout ("Grid") - это двумерная система компоновки основанная на сетке, цель которой заключается в том чтобы полностью изменить способ проектирования пользовательских интерфейсов основанных на сетке

Элемент к которому применяется [display: grid](#). Это прямой родитель для всех элементов сетки. В этом примере container является контейнером.

```
<div class="container">
  <div class="item item-1"></div>
  <div class="item item-2"></div>
  <div class="item item-3"></div>
</div>
```

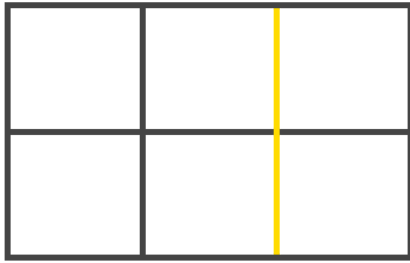
Элемент сетки

Дочерние элементы (прямые потомки) контейнера. Здесь item это элемент сетки, но не sub-item.

```
<div class="container">
  <div class="item"></div>
  <div class="item">
    <p class="sub-item"></p>
  </div>
  <div class="item"></div>
</div>
```

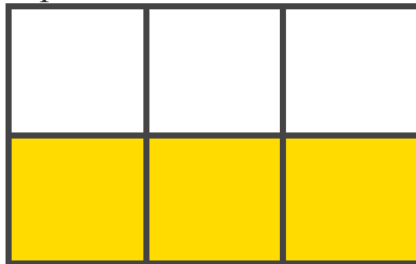
Линия сетки

Разделительные линии, составляющие структуру для сетки. Они могут быть вертикальными ("линии колонок") или горизонтальными ("линии строк") и располагаться по обе стороны от строки или столбца. На изображении жёлтая линия является примером вертикальной линии (линией колонки).



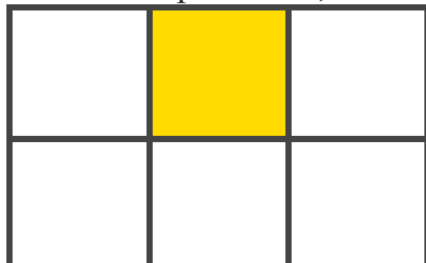
Трек сетки

Пространство между двумя соседними линиями. Вы можете думать об этом как о столбцах или строках сетки. Вот трек между второй и третьей линией строк.



Ячейка сетки

Пространство между линиями двух соседних строк и двух соседних столбцов. Это отдельная "единица измерения" сетки. Вот пример ячейки между линиями строк 1 и 2, линиями колонок 2 и 3.



Область сетки

Общее пространство окружённое четырьмя линиями. Область может состоять из любого количества ячеек. Вот пример области между строками 1 и 3, и колонками 1 и 3.



Основные свойства для работы с технологией grid представлены в таблице 9.7

Свойства для контейнера	Свойства для элементов
<u>display</u>	<u>grid-column-start</u>
<u>grid-template-columns</u>	<u>grid-column-end</u>
<u>grid-template-rows</u>	<u>grid-row-start</u>
<u>grid-template-areas</u>	<u>grid-row-end</u>

<u>grid-template</u>	<u>grid-column</u>
<u>grid-column-gap</u>	<u>grid-row</u>
<u>grid-row-gap</u>	<u>grid-area</u>
<u>grid-gap</u>	<u>justify-self</u>
<u>justify-items</u>	<u>align-self</u>
<u>align-items</u>	
<u>justify-content</u>	
<u>align-content</u>	
<u>grid-auto-columns</u>	
<u>grid-auto-rows</u>	
<u>grid-auto-flow</u>	
<u>grid</u>	

5. Порядок **выполнения работы**

1. Создайте страницу, представленную на рисунке 9.8. Используйте семантическую разметку HTML5

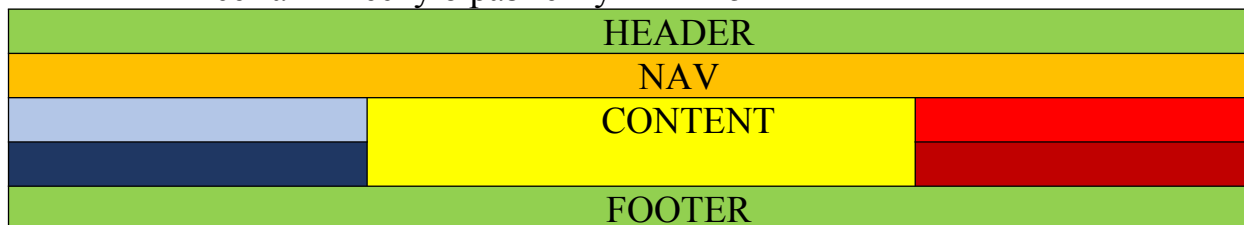
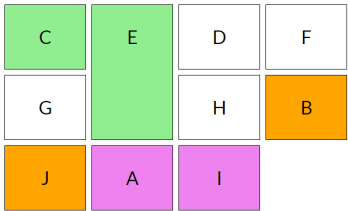
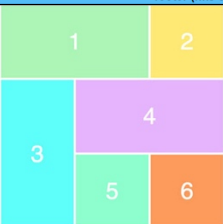


Рисунок 9.8 – Пример реализации задания

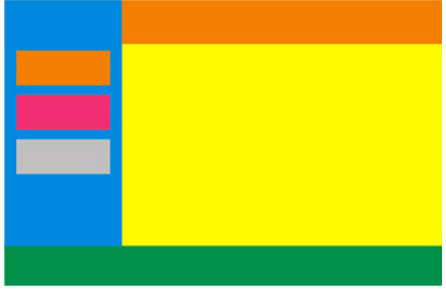

2. С помощью свойства `position` отпозиционируйте на html-странице три изображения.
3. Выполните задания по вариантам

Вариант	Задание	Пример
1	1 Оформите html-страницу с использованием свойства <code>float</code>	
	2 Оформите html-страницу с использованием технологии <code>flex-box</code>	
	3 Оформите html-страницу с использованием технологии <code>grid</code>	

		
2	1 Оформите html-страницу с использованием свойства float	
	2 Оформите html-страницу с использованием технологии flex-box	
	3 Оформите html-страницу с использованием технологии grid	
3	1 Оформите html-страницу с использованием свойства float	
	2 Оформите html-страницу с использованием технологии flex-box	
	3 Оформите html-страницу с использованием технологии grid	
4	1 Оформите html-страницу с использованием свойства float	

		
	3 Оформите html-страницу с использованием технологии grid	
7	1 Оформите html-страницу с использованием свойства float	
	2 Оформите html-страницу с использованием технологии flex-box	
	3 Оформите html-страницу с использованием технологии grid	
8	1 Оформите html-страницу с использованием свойства float	
	2 Оформите html-страницу с использованием технологии flex-box	

		 <p>Шапка сайта</p> <p>Навигация по сайту</p> <p>Меню сайта</p> <p>Основной контент</p> <p>Футер сайта</p>
	3 Оформите html-страницу с использованием технологии grid	
9	1 Оформите html-страницу с использованием свойства float	
	2 Оформите html-страницу с использованием технологии flex-box	
	3 Оформите html-страницу с использованием технологии grid	
10	1 Оформите html-страницу с использованием свойства float	
	2 Оформите html-страницу с использованием технологии flex-box	

		
	3 Оформите html-страницу с использованием технологии grid	

6. Форма отчета о работе

Лабораторная работа № ____

Номер учебной группы _____

Фамилия, инициалы учащегося _____

Дата выполнения работы _____

Тема работы: _____

Цель работы: _____

Оснащение работы: _____

Результат выполнения работы: _____

7. Контрольные вопросы

1. Перечислите значения свойства position. В чем их различие?
2. Чем отличаются значения свойства position absolute и relative?
3. Для чего применяется технология flex-box?
4. Охарактеризуйте технологию grid.

8. Рекомендуемая литература

1. **Макфарланд, Д.** Новая большая книга CSS / Дэвид Макфарланд. – СПб.: Питер, 2016. – 720с.
2. **Никсон, Р.** Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 /Р. Никсон. – 4-е изд. – СПб.: ООО «ПИТЕР М», 2017. – 768 с.

3. **Прохоренок, Н.А.** HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера / Н.А.Прохоренок. – СПб.: БХВ-Петербург, 2010. – 912с.
4. **Робсон, Э.** Изучаем HTML, XHTML и CSS / Э. Робсон. – 2-е изд. – СПб.: ООО «ПИТЕР М», 2017. – 720 с.
5. **Фрейн, Б.** HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств / Бен Фрейн. – СПб.: Питер Пресс, 2017. — 272с.
б.: Питер, 2016.