**Двумерная система сеток Grid Layout**

**10.1 Определение Grid Layout. Grid Container**

Grid Layout представляет специальный модуль CSS3, который позволяет позиционировать элементы в виде сетки или таблицы. Как и Flexbox, Grid Layout представляет гибкий подход к компоновке элементов, только если Flexbox размещает вложенные элементы в одном направлении - по горизонтали в виде столбиков или по вертикали в виде строк, то Grid позиционирует элементы сразу в двух направлениях - в виде строк и столбцов, образуя тем самым таблицу.

Основой для определения компоновки Grid Layout является grid container, внутри которого размещаются элементы. Для создания grid-контейнера необходимо присвоить его стилевому свойству display одно из двух значений: grid или inline-grid.

Далее представлен пример веб-страницы, в которой применяется Grid Layout. В данном примере для контейнера grid-container установлено свойство display:grid. В нем располагается пять grid-элементов.

**<head> <style>**

**.container {border: solid 3px black; display: grid; color: black;}  
 .item { text-align:center;font-size: 1.2em;padding: 1.4em; color: white;}  
 .col1 {background-color: MediumVioletRed;}.col2 {background-color: DeepPink;}  
 .col3 {background-color: HotPink;}.col4 {background-color:PaleVioletRed;} </style></head>**

**<body> <h3>Российские императоры</h3>  
<div class="container"> <div class="item col1">Пётр I Великий. Годы правления: c 22 октября 1721 - 28 января 1725гг.) </div>  
 <div class="item col2">Екатерина I. Годы правления: c 28 января 1725 - 6 мая 1727гг.</div>  
 <div class="item col3"> Анна Иоанновна. Годы правления: с 4 февраля 1730 - 17 октября 1740гг. </div>  
 <div class="item col4"> Елизавета Петровна. Годы правления: с 25 ноября 1741 - 25 декабря 1761гг. </div>  
 <div class="item col1">Екатерина II. Великая. Годы правления: 28 июня 1762 – 6 ноября 1796</div> </div> </body>**

Реализация данного кода представлена на рисунке 10.1.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.1 - Grid Container в Grid Layout в CSS3

Если значение grid определяет контейнер как блочный элемент, то значение inline-grid определяет элемент как строчный (inline). Далее представлен пример применения этого свойства. В этом случае весь грид занимает только то пространство, которое необходимо для размещения его элементов.

**<head> <style>**

**.container {border: solid 3px black;display: inline-grid;}  
 .item {box-sizing: border-box; text-align:center;font-size: 1.2em;  
 padding: 1.4em; color: black;}  
 .col1 {background-color: OrangeRed;} .col2 {background-color: Orange;}  
 .col3 {background-color: Gold ;} .col4 {background-color: Yellow;}  
 </style></head><body> <h3> Театры Оренбурга</h3><div class="container">  
 <div class="item col1"> Оренбургский государственный областной драматический театр им. М. Горького</div>  
 <div class="item col2">Оренбургский театр музыкальной комедии</div>  
 <div class="item col3">Государственный татарский драматический театр им. Мирхайдара Файзи</div>  
 <div class="item col4"> Оренбургский государственный областной театр кукол</div>  
 <div class="item col1">Театр кукол “Пьеро” </div></div></body>**

Реализация данного кода представлена на рисунке 10.2.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.2 – Применение Inline-grid в CSS3

**10.2 Строки и столбцы**

Грид образует сетку из строк и столбцов, на пересечении которых образуются ячейки. И для установки строк и столбцов в Grid Layout использовать следующие свойства CSS3:

* grid-template-columns: настраивает столбцы;
* grid-template-rows: настраивает строки.

Далее представлен пример, где определяется грид с двумя столбцами.

**<head> <style>**

**.container {border: solid 3px black; display: grid;  
 grid-template-columns: 30em 30em;}  
 .item { box-sizing: border-box; text-align:center;font-size: 1.2em;  
 padding: 1.4em; color: white;}  
 .col1 {background-color: Violet }.col2 {background-color: Magenta;}  
 .col3 {background-color: MediumOrchid;;}.col4 {background-color: BlueViolet;}  
 .col5 {background-color: DarkViolet;}**

**</style> </head>**

**<body>**

**<h2>Марки автомобилей и их фирмы-производители</h2><div class="container">  
 <div class="item col1">Форд. Ford Motor Company - американская автомобилестроительная компания. Четвёртый в мире производитель автомобилей по объёму выпуска за весь период существования. </div>  
 <div class="item col2"> Мазерати. Maserati S.p.A.-итальянская компания, производитель эксклюзивных автомобилей спортивного и бизнес-класса. Основана 1 декабря 1914 года в Болонье, Италия. Штаб-квартира находится в городе Модена, Италия.</div>  
 <div class="item col3"> Ауди. Audi AG - немецкая автомобилестроительная компания в составе концерна Volkswagen Group, специализирующаяся на выпуске автомобилей под маркой Audi. Штаб-квартира расположена в городе Ингольштадт, Германия. </div>  
 <div class="item col4"> Мазда. Mazda Motor Corporation-японская автомобилестроительная компания, выпускающая автомобили “Мазда”. Штаб-квартира расположена в посёлке Футю, уезд Аки, префектура Хиросима, Япония. </div>  
 <div class="item col5">Пежо. Peugeot - французский производитель автомобилей, с 1976 года входил в группу PSA Peugeot Citroen, а с 2021 года принадлежит корпорации Stellantis. </div> </div> </body>**

Реализация данного кода представлена на рисунке 10.3.

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.3 - Установка столбцов в Grid Layout в CSS3

В качестве значения свойству grid-template-columns передается ширина столбцов. Сколько нужно иметь в гриде столбцов, столько и нужно передать значений этому свойству. Так, в случае выше грид содержит два столбца, поэтому свойству передаются два значения, которые указывают ширину столбцов:

Соответственно если необходимо, чтобы в гриде было три столбца, то надо передать три значения, например:

**grid-template-columns: 8em 7em 8em;**

Если столбцов больше, чем элементов, то по умолчанию для их вмещения создаются новые строки.

Настойка строк во многом аналогичная настройке столбцов. Для этого у грид-контейнера необходимо установить свойство grid-template-rows, которое задает количество и размеры строк.

Далее представлен пример применения свойства grid-template-rows. Свойству grid-template-rows передается высота каждой из строк. Так, в данном случае высота первой строки составляет 5em, а второй - 8em.

**<head> <style> .container {border: solid 3px black;display: grid; grid-template-columns: 25em 25em 25em; grid-template-rows: 12em 16em;}  
 .item {text-align:center;font-size: 1.2em; padding: 1.4em;}  
 .col1 {background-color: DarkCyan;} .col2 {background-color: LightSeaGreen; }  
 .col3 {background-color: DarkTurquoise; } .col4 {background-color: MediumAquamarine;}  
 .col5 {background-color: Cyan;} .col6 {background-color: PaleTurquoise;}  
 .col7 {background-color: Aquamarine;} </style></head>**

**<body> <h3> Жанры изобразительного искусства. </h3> <div class="container">  
 <div class="item col1">Пейзаж - жанр изобразительного искусства (а также отдельные произведения этого жанра), в котором основным предметом изображения является первозданная либо в той или иной степени преображённая человеком природа. </div>  
 <div class="item col2">Портрет - это жанр изобразительного искусства, посвященный воспроизведению образа конкретного человека или группы людей, показывающий индивидуальные черты человека. </div>  
 <div class="item col3">Бытовой жанр - это жанр изобразительного искусства, предметом которого является изображение сцен повседневной, частной и общественной жизни, обычно современной художнику. </div>  
 <div class="item col4">Анималистика - жанр изобразительного искусства, главным мотивом и основным объектом которого являются животные, главным образом в живописи, фотографии, скульптуре, графике и реже в декоративном искусстве.</div>  
 <div class="item col5">Натюрморт – это жанр изобразительного искусства, в котором изображаются предметы неодушевлённой природы, соединённые художником в отдельную группу и представляющие собой целостную композицию либо включённые в композицию иного жанра.</div>  
 <div class="item col6">Мифический жанр - это жанр изобразительного искусства, в котором основой для сюжета произведений служат древние сказания, легенды, предания или сказки разных народов. </div>  
 <div class="item col7">Батальный жанр – это жанр изобразительного искусства, посвящённый темам войны и песочницей. Главное место в батальном жанре занимают сцены сухопутных, морских сражений и военных походов. /div> </div> </body>**

Реализация данного кода представлена на рисунке 10.4.

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.4 – Работа со строками в Grid Layout CSS3

В то же время, если элементов больше, чем ячеек грида, то образуются дополнительные строки (как в случае со столбцами). Поэтому, несмотря на то что выше были определены настройки только для двух строк, в реальности строк в гриде будет три строки, причем, как видно на скриншоте, высота третьей строки необязательно будет 5em, как у других строк, она будет вычисляться автоматически.

Можно изменить стили grid-контейнера, добавив еще две строки:

**.container {border: solid 3px black;display: grid; grid-template-columns: 25em 25em 25em; grid-template-rows: 12em 12em 12em 12em;}**

В результате будут сформированы три столбца и четыре строки (рисунок 10.5).

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.5 – Работа со столбцами и строками в grid в CSS3

И поскольку ячеек грида больше, чем элементов, то последняя строка оказывается пустой.

* 1. **Функция repeat и свойство grid**

**10.3.1 Повторение строк и столбцов. Свойство repeat**

Если столбцов и(или) строк много и они имеют одинаковые размеры, то есть смыл использовать специальную функцию repeat(), которая позволит настроить строки и столбцы. Так, в примере выше повторяется определение одинаковых строк и столбцов в grid-контейнере [1]:

**grid-template-columns: 8em 8em 8em;**

**grid-template-rows: 5em 5em 5em 5em;**

Здесь происходит повторение одних и тех же размеров - 8em и 5em для установки ширины столбцов и высоты строк. Поэтому переписываются стили, применив функцию repeat:

**.container { border: solid 3px black;display: grid;grid-template-columns: repeat(3, 8em);**

**grid-template-rows: repeat(4, 5em);}**

Первый параметр функции repeat представляет число повторений, а второй - определение строк или столбцов. Например, свойство grid-template-columns: repeat(3, 8em); говорит, что необходимо определить 3 столбца шириной в 8em.

Соответственно выражение grid-template-rows: repeat(4, 5em) определяет 4 строки высотой по 5em.

Можно задавать повторение нескольких столбцов и строк:

**.container { border: solid 3px black;display: grid;**

**grid-template-columns: repeat(2, 7em 8em);grid-template-rows: 6em repeat(3, 5em);}**

В данном случае будет создано 4 столбца: два раза будут повторяться два столбца с шириной 7em и 8em. В случае со строками будет создано 4 строки. Причем первая будет иметь высоту в 6em, а остальные три - 5em.

**10.3.2 Свойство grid**

Свойство grid объединяет свойства grid-template-rows и grid-template-columns и разом позволяет задать настройки для строк и столбцов в следующем формате:

**grid: grid-template-rows / grid-template-columns;**

К примеру, есть следующее определение класса grid-контейнера:

**.container { border: solid 3px black;**

**display: grid;grid-template-columns: 8em 8em 8em;**

**grid-template-rows: 5em 5em 5em 5em;}**

Можно сократить этот класс следующим образом:

**.container {border: solid 3px black; display: grid; grid: 5em 5em 5em 5em / 8em 8em 8em;}**

Либо опять же используя функцию repeat(), можно еще больше сократить определение грида:

**.container { border: solid 3px black; display: grid; grid: repeat(4, 5em) / repeat(3, 8em);}**

* 1. **Размеры строк и столбцов**

**10.4.1 Фиксированные размеры**

В примерах, которые были рассмотрены в предыдущих статьях, ширина столбцов и длина строк устанавливались на основании фиксированных значений, которые передаются свойствам grid-template-columns и grid-template-rows. Для определения размеров можно использовать самые различные единицы измерения, которые доступны в CSS (px, em, rem, pt, %), например:

**.container {border: solid 3px black; display: grid;**

**grid-template-columns: repeat(3, 200px); grid-template-rows: repeat(3, 4.5em);}**

**10.4.2 Автоматические размеры**

Кроме точных размеров можно задавать автоматические размеры с помощью слова auto. В этом случае ширина столбцов и высота строк вычисляются исходя из размеров содержимого. Далее представлен пример, демонстрирующий задание автоматических размеров столбцов и строк.

**<head> <style>**

**.container {border: solid 3px black; display: grid; grid-template-columns: 11em auto auto;**

**grid-template-rows: 8em auto;}**

**.item {text-align:center; font-size: 1.2em; padding: 1.4em; border: 1px solid black;}**

**.col1 {background-color: LightSteelBlue; } .col2 {background-color: PowderBlue;}**

**.col3 {background-color: LightBlue;} .col4 {background-color: SkyBlue;}**

**.col5 {background-color: LightSkyBlue;} .col6 {background-color: DeepSkyBlue;}**

**</style> </head> <body> <h2>Вузы Оренбургской области</h2>**

**<div class="container">**

**<div class="item col1">Оренбургский государственный университет</div>**

**<div class="item col2">Оренбургский государственный аграрный университет </div>**

**<div class="item col3">Оренбургский государственный институт искусств </div>**

**<div class="item col4">Оренбургский государственный медицинский университет</div>**

**<div class="item col5">Оренбургский государственный педагогический университет</div>**

**<div class="item col6">Бузулукский гуманитарно-технологический институт Оренбургского государственного университета</div> </div> </body>**

Реализация данного кода представлена на рисунке 10.6.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.6 – Задание автоматических размеров в Grid Layout в CSS3

Как видно на рисунке 10.6, здесь задано две строки и три столбца. Первый столбец имеет фиксированную ширину в 11em, а второй и третий столбцы - автоматическую ширину. И также первая строка имеют фиксированную высоту, а вторая строка - автоматическую.

**10.4.3 Пропорциональные размеры**

Для установки пропорциональных размеров применяется специальная единица измерения fr. Она представляет собой часть пространства (fraction), которое отводится для данного столбца или строки. Значение fr еще называют flex-фактором (flex factor). Вычисление пропорциональных размеров производится по формуле:

**flex-фактор \* доступное\_пространство / сумма всех flex-факторов**

При этом под доступным пространством понимается все пространство grid-контейнера за исключением фиксированных значений строк и столбцов. Далее приведен пример задания пропорциональных размеров блоков в CSS.

**<head> <style>**

**\*{box-sizing: border-box; }  
 html, body { margin:1px; padding:1px;}  
 .container {height: 80vh; border: solid 3px black;**

**display: grid; grid-template-columns: 1.5fr 15em 1fr;**

**grid-template-rows: 1fr 12em 1fr;}  
 .item {text-align:center; font-size: 1.2em;**

**padding: 1.4em;color: white;**

**border: 1px solid black;}  
 .col1 {background-color: DodgerBlue;}**

**.col2{background-color: CornflowerBlue;}  
 .col3 {background-color: MediumSlateBlue;}**

**.col4 {background-color: RoyalBlue;}  
 .col5 {background-color: SteelBlue;}**

**.col6 {background-color: CadetBlue;} </style></head>  
<body> <h3>Филиалы вузов в городе Оренбурге</h3>**

**<div class="container">  
 <div class="item col1">Оренбургский филиал Российского государственного университета нефти и газа им. И.М. Губкина</div>  
 <div class="item col2">Оренбургский филиал Российского экономического университета имени Г.В. Плеханова</div>  
 <div class="item col3">Оренбургский филиал Российской академии народного хозяйства и государственной службы при Президенте РФ</div>  
 <div class="item col4">Оренбургский институт путей сообщения Самарского государственного университета путей сообщений </div>  
 <div class="item col5">Оренбургский институт (филиал) Московского государственного юридического университета имени О.Е. Кутафина</div>  
 <div class="item col6"> Оренбургский филиал Поволжского государственного университета телекоммуникаций и информатики</div> </div></body>**

Реализация данного кода представлена на рисунке 10.7.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.7 - Использование Flex factor и fr в Grid Layout и CSS3

В данном случае имеются три столбца с шириной 2fr, 8em, 1fr. Поэтому ширина второго столбца будет вычисляться по формуле:

**2 \* (ширина\_грида - 15em) / (1.5 + 1)**

Ширина третьего столбца будет вычисляться по формуле:

**1 \* (ширина\_грида - 15em) / (1.5 + 1)**

И если первый столбец, фиксированный с шириной 15em, то ширина второго и третьего столбца будут зависеть от ширины контейнера и будут автоматически масштабироваться при ее изменении. В отношении строк все аналогично.

**10.5 Отступы между столбцами и строками**

Для создания отступов между столбцами и строками применяются свойства grid-column-gap и grid-row-gap соответственно. Далее приведен пример задания отступов между строками и столбцами.

**<head> <style>**

**\*{box-sizing: border-box;}  
 html, body{ margin:1px; padding:1px;}  
 .container {height: 80vh; display: grid; grid-template-columns: repeat(3, 1fr);  
 grid-template-rows: repeat(3, 1fr); grid-column-gap: 9px; grid-row-gap: 9px;}  
 .item {text-align:center; font-size: 1.2em; padding: 1.4em;border: 1px solid black;}  
 .col1 {background-color: Beige;} .col2 {background-color: OldLace;}  
 .col3 {background-color: AntiqueWhite;} .col4 {background-color: LavenderBlush;}  
 .col5 {background-color: MistyRose;} .col6{background-color: Linen;}</style> </head>**

**<body> <h2>Клавишные музыкальные инструменты</h2> <div class="container">  
 <div class="item col1">Фортепиано</div> <div class="item col2">Орган</div>  
 <div class="item col3">Рояль</div><div class="item col4">Клавесин</div>  
 <div class="item col5">Аккордеон</div><div class="item col6">Фисгармония</div> </div> </body>**

Реализация данного кода представлена на рисунке 10.8

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.8 – Задание отступов между столбцами и строками в Grid Layout

Если значения свойств grid-column-gap и grid-row-gap совпадают, то вместо них можно определить одно свойство gap (ранее назвалось grid-gap), которое установит оба отступа:

**.container {height: 80vh; display: grid; grid-template-columns: repeat(3, 1fr); grid-template-rows: repeat(3, 1fr); grid-column-gap: 9px; grid-row-gap: 9px;}**

**10.6 Позиционирование элементов в системе сеток Grid Layout**

Грид - это совокупность ячеек, образованных на пересечении столбцов и строк. Но сами строки и столбцы создаются с использованием grid-линий, пересекающих сетку по вертикали и горизонтали (рисунок 10.9):

Изображение выглядит как текст, линия, диаграмма, График

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.9 - Grid-lines in Grid Layout CSS3

По умолчанию каждый элемент сетки располагается в ячейке по очереди. Однако более точно настроить положение элемента в сетке можно с помощью ряда свойств:

* grid-row-start: указывает начальную горизонтальную линию сетки, с которой начинается элемент;
* grid-row-end: указывает, до какой горизонтальной grid-линии следует растягивать элемент;
* grid-column-start: определяет вертикальную начальную линию сетки, с которой начинается элемент;
* grid-column-end: указывает, до какой вертикальной линии сетки следует растянуть элемент.

Далее приведен пример растягивания элементов на несколько столбцов.

**<head> <style>.container { border: solid 3px black;display: grid;  
 grid-template-columns: repeat(4, 1fr);grid-template-rows: repeat(2, 5em); color:white;}  
 .s-item{ grid-column-start:2; grid-column-end: 5; }  
 .item {text-align:center; font-size: 1.2em; padding: 1.4em ; border:1px solid black;}  
 .col1 {background-color: Gainsboro;} .col2 {background-color: Silver;}  
 .col3 {background-color: DarkGrey;} .col4 {background-color: DimGrey;}  
 .col5 {background-color: SlateGrey ;} .col6 {background-color: DarkSlateGrey;} </style> </head> <body> <h2> Струнные музыкальные инструменты</h2><div class="container">  
 <div class="item col1">Гитара</div>  
 <div class="item col2 s-item">Скрипка</div>  
 <div class="item col3">Альт</div>  
 <div class="item col4">Виолончель</div>  
 <div class="item col5">Контрабас</div>  
 <div class="item col6">Балалайка</div> </div>**

**</body>**

Реализация данного кода представлена на рисунке 10.10.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.10 – Пример растяжения элементов на столбцы в Grid Layout

Здесь определяются четыре столбца, а второму элементу назначается специальный элемент специального класса, который располагается начиная со 2-й линии сетки или 2-го столбца (grid-column-start: 2) до 5-й вертикальной линии сетки (grid-column -end : 5).

Второй элемент не обязательно должен начинаться со второго столбца, он может быть любым другим: первым, третьим и т. д. Например, если размещается второй элемент, начиная с 3-го столбца, на месте второго появится пустота (рисунок 10.11):

**.s-item{ grid-column-start: 3; grid-column-end: 5;}**

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.11 - Использование Grid-column-start в Grid Layout в CSS3

Если установить второй элемент так, чтобы он начинался в первом столбце, второй элемент будет перемещен в следующую строку и, таким образом, начнется в первом столбце.

Вместо использования двух рассмотренных выше свойств можно использовать одно свойство - grid-column, которое принимает значения grid-column-start и grid-column-end, разделенные косой чертой:

**grid-column: grid-column-start / grid-column-end;**

Например, можно сократить стиль класса special-item следующим образом:

**.s-item{grid-column: 3 / 5;}**

Аналогично с помощью свойств grid-row-start и grid-row-end можно задать позиционирование элемента на несколько строк. Так, изменим класс special-item следующим образом:

**.s-item{ grid-column-start:2; grid-row-start: 1;grid-row-end: 3;}**

В данном случае второй элемент позиционируется во втором столбце первой строки и растягивается до 3-й строки (рисунок 10.12).

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.12 – Использование Grid-row-start в Grid Layout и CSS3

Вместо использования пары свойств grid-row-start и grid-row-end можно использовать одно общее свойство grid-row:

**grid-row: grid-row-start / grid-row-end;**

Так, можно изменить стиль special-item следующим образом:

**.s-item{grid-column-start:2;grid-row: 1 / 3; }**

**10.6.1 Растяжение ячеек с помощью свойства Span**

С помощью специального слова span можно задать растяжение элемента на несколько ячеек. После слова span указывается, на какое количество ячеек надо растянуть элемент.

**.s-item{grid-row: 1 / span 2; grid-column: 2 / span 2; }**

Элемент помещается в ячейку, которая находится на пересечении первой строки и второго столбца, и растягивается на две строки вниз и на два столбца вправо.

**10.6.2 Grid-area**

Свойство grid-area объединяет свойства grid-column и grid-row, позволяя сократить их запись:

**grid-area: row-start / column-start / row-end / column-end.**

На рисунке 10.13 стили класса special-item были изменены:

**.s-item{grid-area: 1 / 2 / 3 / 4; }**

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.13 – Использование Grid-area в Grid Layout в CSS3

**10.7 Направление и порядок элементов**

**10.7.1 Grid-auto-flow**

По умолчанию все элементы располагаются по порядку горизонтально, если места в строке больше нет, то элементы переносятся на следующую строку. Но с помощью свойства grid-auto-flow можно изменить направление элементов. Это свойство принимает два значения:

* row: значение по умолчанию, элементы располагаются в строку друг за другом, если места в строке не хватает, элементы переносятся на следующую строку;
* column: элементы располагаются в столбик, если места в столбце не хватает, то элементы переходят в следующий столбец.

Далее приведен пример применения данного свойства.

**<head> <style>**

**\*{ box-sizing: border-box;}  
 html, body{ margin:1px; padding:1px;}  
 .container { height: 80vh; display: grid;  
 grid-template-columns: repeat(3, 1fr);**

**grid-template-rows: repeat(3, 1fr);  
 grid-auto-flow: row; color: white;}  
 .item {text-align:center; font-size: 1.2em;**

**padding: 1.4em;}  
 .col1 {background-color:Purple;}**

**.col2 {background-color: Maroon;}  
 .col3 {background-color: Olive;}**

**.col4 {background-color: Green;}  
 .col5 {background-color: Teal;}**

**.col6 {background-color: Navy;}**

**</style></head>**

**<body><h3> Лучшие курорты краснодарского края</h3>**

**<div class="container">  
 <div class="item col1">Сочи</div>  
 <div class="item col2">Анапа</div>  
 <div class="item col3">Адлер</div>  
 <div class="item col4">Геленджик</div>  
 <div class="item col5">Туапсе</div>  
 <div class="item col6"> Сухуми</div></div></body>**

Реализация данного кода представлена на рисунке 10.14.

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.14 – Применение свойства Grid-auto-flow в CSS3

Свойство grid-auto-flow имеет значение row, поэтому элементы будут располагаться в строку. Теперь изменим стиль grid-контейнера:

**.container {height: 80vh; display: grid;**

**grid-template-columns: repeat(3, 1fr);**

**grid-template-rows: repeat(3, 1fr);**

**grid-auto-flow: column; }**

После этого элементы будут располагаться в столбец (рисунок 10.15).

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.15 – Использвание Grid-auto-flow в Grid Layout CSS3

**10.7.2 Свойство order**

Свойство order позволяет задать порядок элементов. По умолчанию для каждого элемента в гриде это свойство имеет значение 0. Поэтому элементы располагаются друг за другом, как они определены в разметке html. Но можно переопределить этот порядок. В ниже приведенном примере задано изменение порядка с помощью свойства order.

**<head> <style>**

**\*{ box-sizing: border-box; }  
 html, body{margin:1; padding:1; }  
 .container { height: 80vh; display: grid;**

**grid-template-columns: repeat(3, 1fr);**

**grid-template-rows: repeat(3, 1fr); }  
 .item { text-align:center;**

**font-size: 1.2em; padding:**

**1.4em;}  
 .l-item{ order: 1; } .f-item{ order: -1; }  
 .col1 {background-color: Fuchsia;}**

**.col2 {background-color: Red;}  
 .col3 {background-color: Yellow;}**

**.col4 {background-color: Lime;}  
 .col5 {background-color: Aqua;}**

**.col6{background-color: Blue ;} </style> </head>  
<body> <h2>Лучшие пляжи города-курорта Сочи</h2>**

**<div class="container">  
 <div class="item col1">“Ривьера”. Сочи, Ривьерский переулок, 5.</div>  
 <div class="item col2 l-item">”Мандарин”. Адлер, ул. Бестужева, 1. </div>  
 <div class="item col3 f-item">“Олимпийский пляж”. Поселок Сириус, рядом со стадионом “Фишт”. </div>  
 <div class="item col4">“Пляж Лоо”. Сочи, ул. Лучезарная, 14. </div>  
 <div class="item col5">“Альбатрос”. Микрорайон Новый Сочи, ул. Политехническая.</div>  
 <div class="item col6"> “Жемчужина”. Сочи, ул. Черноморская, 11.</div> </div> </body>**

Реализация данного кода представлена на рисунке 10.16.

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.16 – Задание порядка элементов в Grid Layout в CSS3

Элементы с классом l-item имеет порядок 1, поэтому они будут располагаться после других элементов, у которых порядок равен 0 или меньше.

Если же необходимо поставить одни элементы перед другими, то можно использовать отрицательное значение для свойства order. Так, у третьего элемента порядок равен -1, что меньше, чем у других элементов.

**10.8 Создание макета страницы в Grid Layout**

Рассмотрим создание простейшего адаптивного стандартного макета веб-страницы, который состоит из шапки, подвала, основного содержимого, блока навигации и сайдбара. Для этого определяется следующая веб-станица:

**<head> <style> \*{ box-sizing: border-box;}**

**html, body {margin:1; padding:1;}**

**.container {height:50vh; display: grid;**

**grid-template-areas: "header" "."**

**"snegur" "." "content" "." "dedsnow" "." "footer";**

**grid-template-columns: 1fr; grid-template-rows: 80px 5px 100px 5px 1fr 5px 120px 5px 50px;}**

**.header {grid-area: header; background-color: LightSkyBlue; }**

**.snegur {grid-area: menu; background-color: PowderBlue; }**

**.dedsnow {grid-area: sidebar; background-color: PowderBlue; }**

**.content {grid-area: content; background-color: LightSteelBlue; }**

**.footer {grid-area: footer; background-color: SkyBlue;}**

**h3{ text-align:center;}**

**@media screen and (min-width: 480px) {**

**.container { height:50vh; display: grid;**

**grid-template-areas: "header header header header header"**

**". . . . ." "snegur. content . dedsnow"**

**". . . . ." "footer footer footer footer footer";**

**grid-template-columns: 300px 5px 1fr 5px 300px;**

**grid-template-rows: 50px 5px 1fr 5px 50px;}} </style> </head> <body>**

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

**<h3>Лучший праздник – Новый год! </h3></div>**

**<div class="content"><p>Самый волшебный и долгожданный праздник в году для взрослых и детей – это Новый год! Его ждут с нетерпением в разных странах мира, чтобы загадать заветные желания и получить подарки о которых мечтали. Главным символом Нового года считается лесная красавица - елка, которую украшают гирляндами и разноцветами шарами. В новогоднюю ночь, под бой курантов, на своих северных оленях Дед Мороз приносит подарки каждому ребенку. Помогает ему творить волшебство, его вечная спутница - внучка – Снегурочка! </h3></div>**

**<div class="snegur"><h3>Снегурочка – внучка Деда Мороза. Русский сказочный и новогодний персонаж, внучка Деда Мороза, его постоянная спутница и помощница. В русском**[**фольклоре**](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D0%BB%D1%8C%D0%BA%D0%BB%D0%BE%D1%80)**Снегурочка является персонажем**[**народной сказки**](https://ru.wikipedia.org/wiki/%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B5_%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D1%8B%D0%B5_%D1%81%D0%BA%D0%B0%D0%B7%D0%BA%D0%B8)**о сделанной из снега девочке Снегурке, которая ожила.  Она обычно носит синюю шубу и корону. Русская Снегурочка - светловолосая, стройная и очень добрая. </h3></div>**

**<div class="dedsnow"><h3>Дед Мороз - главный сказочный персонаж на русском празднике Нового года. Прообразами являются Мороз - персонаж славянского сказочного фольклора и персонажи, основанные на образе святого Николая Чудотворца. Родиной Деда Мороза является небольшой город в Вологодской области Великий Устюг, который последнее время называют Сказочной столицей России.</h3></div>**

**<div class="footer"><h3>>&copy; Copyright Новый год 2024-2025 год </h3 ></div>**

**</div> </body>**

В данном примере в стилях имеются два разных определения грида. Одно определение грида для мобильных устройств (условно в качестве ширины устройств выбрано значение в 480px) (рисунок 10.17):

**.container { height: 50vh;display: grid;grid-template-areas: "header" "."**

**"snegur" "."**

**"content" "."**

**"dedsnow" "."**

**"footer";**

**grid-template-columns: 1fr; grid-template-rows: 80px 5px 100px 5px 1fr 5px 120px 5px 50px;}**

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.17 - Макет веб-станицы с Grid Layout в CSS3

В таком состоянии грид имеет только один столбец и 5 строк для каждой области плюс 4 строки-разделители. При увеличении ширины экрана в действие вступает другое определение грида (рисунок 10.18):

**.container {height:50vh; display: grid; grid-template-areas: "header header header header header"**

**". . . . ." "snegur . content . dedsnow"**

**". . . . ." "footer footer footer footer footer";**

**grid-template-columns: 300px 5px 1fr 5px 300px; grid-template-rows: 50px 5px 1fr 5px 50px;}**

В этом случае столбцов и строк будет по пять.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10.18 - Адаптивный макет станицы с Grid Layout в CSS3