Выполнен на 0%

Углублённая теория

элементов

№ Кейс 1, лёгкий уровень

№ Кейс 2, лёгкий уровень

№ Кейс 3, лёгкий уровень

№ Кейс 4, лёгкий уровень

🖸 Кейс 5, средний уровень

🖸 Кейс 6, средний уровень

🖸 Кейс 7, средний уровень

Кейс 8, сложный уровень

Кейс 9, сложный уровень

🖸 Кейс 10, сложный уровень

🖪 Построение сетки: выбор технологии

В Немного про доступность и ARIA

В Методика вёрстки карточных

Построение сетки: выбор технологии

Попробуем решить раз и навсегда, так всё-таки *CSS Flexbox* или *CSS Grid*. Или, может, они нужны для разного?

🚱 Виталий Ризун 🔻

Мы выпустили по навыку по обоим: «Построение сеток на флексах по макету» и «Построение сеток на гридах по макету», а теперь, кажется, должны решить, кто из них больше подходит.

Но получается, что удобны обе технологии, просто у каждой есть свои достоинства и недостатки, и есть виды макетов, в которых напрашивается тот или иной подход к сетке. Поговорим об этом.

Основное различие между гридами и флексами заключается в количестве измерений. Флекс управляет одной осью, вторая — вспомогательная. Флекс работает как описание правил работы контейнера, дальнейшее зависит от контента. Грид двухмерный, и колонки, и ряды можно описывать сколько угодно подробно, при этом структура описывается заранее. Чем это оборачивается для верстальщика?

```
Title
  Menu
                   Content
                                                                                          Ads
  Footer
Типичный и схематичный макет Holy Grail
```

Разметка, допустим, такая:

```
<body class="hg container">
  <header class="hg-header">
   Title
  </header>
  <nav class="hg-menu">
   Menu
  </nav>
  <main class="hg-main">
    Content
  </main>
  <aside class="hg-aside">
   Ads
 </aside>
  <footer class="hg-footer">
    Footer
  </footer>
</body>
```

Вот условные стили для него в гриде:

Этот макет можно сверстать и гридами и флексами.

wгар.

```
.container {
 width: 1200px;
 margin: 0 auto;
.hg {
 display: grid;
 grid-template-columns: 150px 1fr 150px;
 grid-template-rows:
   100px
   1fr
   30px;
 grid-gap: 10px;
 min-height: 100vh;
.hg-header {
 grid-column: 1 / 4;
.hg-footer {
 grid-column: 1 / 4;
```

<body class="hg container">

С флексом придётся добавить контейнер для боковых панелей и основного содержимого. Назовём его классом

```
<header class="hg-header">
   Title
 </header>
 <div class="wrap">
   <nav class="hg-menu">
     Menu
   </nav>
   <main class="hg-main">
     Content
   </main>
   <aside class="hg-aside">
     Ads
   </aside>
 </div>
 <footer class="hg-footer">
   Footer
 </footer>
</body>
.container {
```

```
width: 1200px;
 margin: 0 auto;
.hg {
 display: flex;
  flex-direction: column;
  flex-wrap: wrap;
 min-height: 100vh;
.wrap {
 display: flex;
  flex-direction: row;
  height: calc(100vh - 130px);
 justify-content: space-between;
.hg-header,
.hg-footer {
 box-sizing: border-box;
 min-width: 100%;
.hg-header {
 flex-basis: 100px;
.hg-footer {
  flex-basis: 30px;
.hg-main {
  box-sizing: border-box;
 width: calc(100% - (150px * 2 + 10px * 2));
.hg-aside,
.hg-menu {
 width: 150px;
  align-items: stretch;
```

Или приведём пример с «расталкиванием» элементов. Сложно переоценить удобство свойства флексбокса justify-content, особенно когда речь идёт о подобных вещах.

У грида было удобно оформить промежуток с помощью свойства <code>grid-gap</code>, а у флекса пришлось немного

по отдельным местам в двухмерной сетке.

grid-auto-flow: column;

Первый

Bторой

увеличенная зона клика.

display: grid;

о поддержке гридов).

CSS Grid Layout (level 1) - cr

Method of using a grid concept to lay out content, providing a mechanism for authors to divide available space for layout into

grid-auto-flow: column;

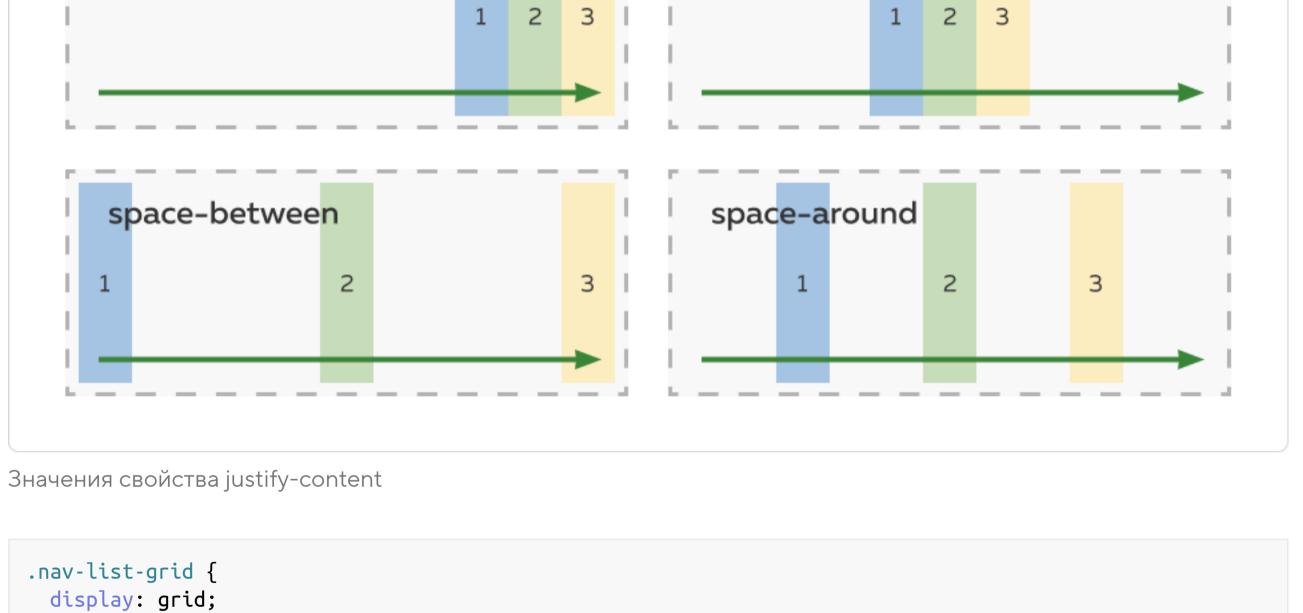
grid-template-columns: repeat(8, 1fr);

Eщё один

Последний

заняться вычислениями. У флекса получилось больше кода, но это пока мы не принялись расставлять блоки

flex-end center



display: flex;

```
.nav-list-flex {
   justify-content: space-between;
По идее работает хорошо и то, и другое. Но теперь мы хотим, чтобы только одна из ссылок оказалась справа,
а остальные остались бы слева. При вот таком примитивном коде уже будет разница между реализацией в гриде
и флексбоксе:
```

.header-menu { display: flex; justify-content: space-between;

Вот флексбокс. Ничего плохого в отступах как таковых нет: если это ссылки, то тем лучше, у ссылок будет

```
list-style: none;
 .header-menu li {
   padding: 0 15px;
 .header-menu li:last-child {
   margin-left: auto;
Вот грид. Управление тоньше и конкретнее, и, хотя грид позволил выполнить задачу быстрее, у макета остаётся
меньше гибкости, и меньше зависимости от контента: грид интересуется содержимым меньше, чем флекс.
 .header-menu {
```

list-style: none; .header-menu li:last-child { grid-column: -2 / -1;

Иногда возникают вопросы: а можно ли использовать флекс для крупных сеток? Отвечаем: можно. Использовать гриды для совсем мелких элементов тоже можно. Важно, что если вы работаете с уже готовым сайтом, на котором, например, используются флексы, стоит встроиться в существующий процесс. А вот если вы верстаете с нуля, то выбор за верстальщиком. При этом нужно внимательно изучить техническое задание к сайту: если необходима поддержка старых браузеров, то флексбокс предпочтительнее, так как гриды могут не поддерживаться или поддерживаться хуже (информация

% of all users \$?

94.7% + 1.16% = 95.86%

Usage Global

Остальное

Написать нам

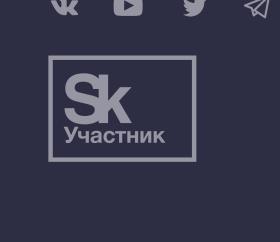
Мероприятия

Форум

```
columns and rows using a set of predictable sizing behaviors.
      Includes support for all grid-* properties and the fr unit.
       Current aligned Usage relative Date relative Filtered All 🌣
                       Firefox Chrome Safari Opera iOS Safari Opera Mini Android Opera Mobile
                        2-39 4-28
                      <sup>8</sup>40-51 <sup>1</sup>29-56 <sup>1</sup>
                                              10-27
        6-9 12-15 52-53 57 3.1-10 28-43 3.2-10.2
                                                                                                             4-5.4
       10 16-86 54-83 58-86 10.1-13.1 44-71 10.3-13.7
                                                                                                            6.2-12.0
                                                                     2.1-4.4.4 12-12.1
                                             72 14.2 all 81 59 87 83 12.12 13.0 10.4 7.12 2.
                      85-86 88-90 TP
       Notes Test on a real browser NEW Known issues (3) Resources (10) Feedback
       See also support for <u>subgrids</u>
      ■ Enabled in Chrome through the "experimental Web Platform features" flag in chrome://flags
       2 Partial support in IF refers to supporting an older version of the specification
Поддержка грида как технологии
Мы предпочитаем совмещать в зависимости от собственного удобства и условий. Рекомендуем попробовать
сверстать одни и те же элементы и гридами, и флексами: это позволит определиться, что и для каких ситуаций
```

лучше подходит и больше вам нравится.

```
Ознакомились со статьёй?
   Сохранить прогресс
< Углублённая теория
                                                       Немного про доступность и ARIA >
```



```
Практикум
                              Курсы
                                                                                             Блог
                              HTML и CSS. Профессиональная вёрстка сайтов
                                                                                             С чего начать
Тренажёры
                              HTML и CSS. Адаптивная вёрстка и автоматизация
                                                                                             Шпаргалки для разработчиков
Подписка
                                                                                             Отчеты о курсах
Для команд и компаний
                              JavaScript. Профессиональная разработка веб-интерфейсов
Учебник по РНР
                              JavaScript. Архитектура клиентских приложений
                              React. Разработка сложных клиентских приложений
Профессии
                                                                                             Информация
                                                                                             Об Академии
Фронтенд-разработчик
                              РНР. Профессиональная веб-разработка
React-разработчик
                              РНР и Yii. Архитектура сложных веб-сервисов
                                                                                             О центре карьеры
Фулстек-разработчик
                              Node.js. Разработка серверов приложений и API
```

© ООО «Интерактивные обучающие технологии», 2013-2022 **</>/> → ♥** → **!!!**

Анимация для фронтендеров

Vue.js для опытных разработчиков

Алгоритмы и структуры данных

Регулярные выражения для фронтендеров

Вёрстка email-рассылок

Шаблонизаторы HTML

Анатомия CSS-каскада

Бэкенд-разработчик

Работа наставником

Услуги

Для учителей

Стать автором