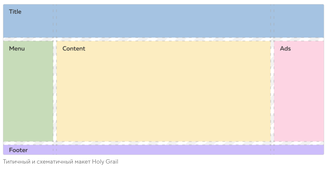
Построение сетки: выбор технологии

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

Мы выступили 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 оба 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, второе — 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 обрачивается для верстальщика?



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>
```

Этот макет можно сверстать и гридами и флексами.

Вот условные стили для него в гриде:

```
.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;
}
```

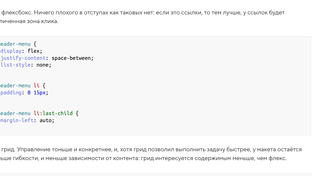
С флексом придётся добавить контейнер для боковых панелей и основного содержимого. Назовём его классом `wrap`:

```
<body class="hg container">
  <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;
}
```

У грида было удобно оформить промежуток с помощью свойства `grid-gap`, а у флекса пришлось немного заняться вычислениями. У флекса получилось больше кода, но это пока мы не принялись расставлять блоки по отдельным местам в двухмерной сетке.

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



Значения свойства justify-content

```
.nav-list-grid {
  display: grid;
  grid-auto-flow: column;
}

.nav-list-flex {
  display: 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 оставались бы слева. При вот таком примитивном коде уже будет разница между реализацией в гриде и флексбоксе:

```
<ul class="header-menu">
  <li>Первый</li>
  <li>Второй</li>
  <li>Ещё один</li>
  <li>Последний</li>
</ul>
```

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

```
.header-menu {
  display: flex;
  justify-content: space-between;
  list-style: none;
}

.header-menu ul {
  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 {
  display: grid;
  grid-auto-flow: column;
  grid-template-columns: repeat(8, 1fr);
  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 учитывать изучить техническое задание и сайту: если необходима поддержка старых браузеров, то флекс-боксе предпочтительнее, так как гриды могут не поддерживаться или поддерживаться хуже (информация о поддержке гридов).

Поддержка грида как технологии

Мы предпочитаем совмещать в зависимости от собственного удобства и условий. Рекомендуем попробовать сверстать один 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