**CSS Grid — мощная сеточная верстка для любых задач**

**8.1. Что такое CSS Grid и зачем он нужен?**

CSS Grid — это современный инструмент для создания двумерных макетов. Он позволяет легко и гибко управлять расположением элементов сразу в двух измерениях: горизонтально (столбцы) и вертикально (строки).

**В чём главные преимущества CSS Grid:**

* Простое создание сложных адаптивных макетов.
* Возможность точного позиционирования элементов.
* Удобство работы с большими и сложными сетками.

**8.2. Основы работы с CSS Grid**

**Создание сетки (grid-контейнера)**

css

КопироватьРедактировать

.container {

display: grid;

grid-template-columns: 200px 1fr 200px;

grid-template-rows: auto;

gap: 20px; /\* Отступы между ячейками \*/

}

**grid-template-columns и grid-template-rows**

Определяют размеры столбцов и строк сетки:

* Фиксированные размеры (px):

css

КопироватьРедактировать

grid-template-columns: 150px 200px;

* Гибкие размеры (fr):

css

КопироватьРедактировать

grid-template-columns: 1fr 2fr;

* Адаптивные размеры (функция repeat() и minmax()):

css

КопироватьРедактировать

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

**8.3. Расположение элементов в Grid**

**Явное указание позиции элементов**

css

КопироватьРедактировать

.item {

grid-column: 2 / 4; /\* занимает место со второго по четвертый столбец \*/

grid-row: 1 / 3; /\* занимает место с первой по третью строку \*/

}

**Grid-area (именованные области)**

css

КопироватьРедактировать

.container {

display: grid;

grid-template-areas:

"header header header"

"sidebar content content"

"footer footer footer";

}

.header { grid-area: header; }

.sidebar { grid-area: sidebar; }

.content { grid-area: content; }

.footer { grid-area: footer; }

**8.4. Управление сеткой и выравниванием элементов**

**Выравнивание контента (justify-items и align-items)**

* По горизонтали:

css

КопироватьРедактировать

.container {

justify-items: center;

}

* По вертикали:

css

КопироватьРедактировать

.container {

align-items: center;

}

**Управление ячейками (justify-content и align-content)**

* Выравнивание всей сетки по горизонтали или вертикали:

css

КопироватьРедактировать

.container {

justify-content: center;

align-content: center;

}

**8.5. Современные возможности CSS Grid**

**auto-fit и auto-fill**

css

КопироватьРедактировать

grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));

Позволяют автоматически заполнять доступное пространство сеткой, что удобно для адаптивных галерей.

**Использование grid-auto-flow для автоматического размещения элементов**

css

КопироватьРедактировать

.container {

grid-auto-flow: dense; /\* заполнение пустых ячеек сетки \*/

}

**8.6. Реальные примеры применения Grid**

**Галерея изображений**

css

КопироватьРедактировать

.gallery {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

gap: 10px;

}

**Макет страницы с шапкой, боковым меню и контентом**

css

КопироватьРедактировать

.layout {

display: grid;

grid-template-areas:

"header header"

"sidebar main"

"footer footer";

grid-template-columns: 250px 1fr;

grid-template-rows: auto 1fr auto;

}

**8.7. Современные трюки и советы по Grid**

* **Используй minmax() и auto-fit/auto-fill для адаптивности**
* **Именуй области сетки (grid-template-areas) для удобства поддержки**
* **Используй grid-auto-rows для автоматических размеров строк**

**8.8. Flexbox vs Grid — в каких случаях Grid лучше?**

* Используй Grid, если тебе нужен двумерный макет с контролем и над колонками, и над строками.
* Grid отлично подходит для сложных структур: dashboards, галереи, каталоги товаров.

Flexbox — для более простых случаев (одномерных структур).

**8.9. Лучшие практики работы с Grid**

* Не усложняй сетку без необходимости.
* Всегда используй gap для отступов между элементами.
* Применяй repeat(), auto-fit и minmax() для адаптивности.
* Используй grid-template-areas для больших и сложных макетов.

**8.10. Частые ошибки и анти-паттерны CSS Grid**

* Использование Grid там, где достаточно Flexbox.
* Игнорирование адаптивности (жёсткие фиксированные размеры).
* Неиспользование возможностей gap, что приводит к сложному margin.

**8.11. Глоссарий CSS Grid**

| **Термин** | **Определение** |
| --- | --- |
| Grid-template-columns | Шаблон для задания размеров столбцов |
| Grid-template-rows | Шаблон для задания размеров строк |
| Grid-area | Область сетки с именем |
| Gap | Отступы между ячейками сетки |
| Auto-fit, auto-fill | Адаптивные методы заполнения сетки |
| Minmax() | Задаёт минимальный и максимальный размеры элемента |
| Grid-auto-flow | Автоматическое размещение элементов |

**8.12. Вопросы для самопроверки**

1. В чём основное отличие CSS Grid от Flexbox?
2. Для чего используются grid-template-areas?
3. Чем auto-fit отличается от auto-fill?
4. Как сделать адаптивную галерею через Grid?
5. Когда стоит использовать grid-auto-flow?