1. Flexbox – с его помощью можно просто и гибко расставить элементы в контейнере, распределить доступное пространство между ними, и выровнять элементы несколькими способами и по разным осям. Он хорош именно своей гибкость, позволяет создавать гибкие макеты, так что без конкретных размеров элементы будут подстраваться под экраны пользователей. Адаптивный дизайн с ним также проще создавать.
2. HTML-таблица — это когда для создания таблицы используется нативный HTML-тег <table>, а CSS-таблица — это имитация того же поведения, но с помощью CSS-свойств.

CSS-свойство display: table и другие, делают вывод группы элементов подобно таблице <table>, но с ограничением – объединения ячеек colspan и rowspan не поддерживаются.

Проблема табличной верстки в том, что с ней не добиться адаптивности (или весьма тяжело добиться). При изменении содержимого какой либо ячейки происходит пересчет всей таблицы (расходует ресурсы). Табличную верстку сложнее поддерживать.

**Таблицы лучше использовать по их прямому назначению - для отображения таблиц.**

1. Горизонтальное выравнивание 3ся способами:

- если у нас flex-direction:row, то justify-content:center

-если у нас flex-direction:column, то align-items:center-

-если хотим элемент флексконтейнера выровнять по центру, можем указать align-self:center

можно при absolute/relative позициониировании указаать значение left (просто подбором px)

1. есть две оси во флекс верстке: главная (слева направо, задает основное направление) и поперечная (сверху вниз, перпендикулярна главной, определяет направление флекс элементов при переносе на новую линию). Направление главной оси задается и меняется flex-direction. Если flex-direction:row (то слева направо), если flex-direction:column (то сверху вниз). С помощью reverse можно изменять направление (то есть будет справа-налево, снизу-вверх).
2. Margin:auto займет все доступное пространство в своём измерении. Например, если установить margin-left и margin-right в значение auto, обе стороны блока попытаются занять все доступное пространство и, таким образом, заставят двигаться блок в центр.
3. box-sizing определяет как вычисляется общая ширина и высота элемента. По умолчанию если у элемента есть граница или внутренний отступ, то они добавляются к ширине и высоте, чтобы получить отображаемый на экране размер. Свойство box-sizing может изменять это поведение:

* content-box даёт стандартное поведение свойства box-sizing, значение по умолчанию (обычный вариант-все отступы и бордеры суммируются, выставляемая ширина и высота – это только контент);
* border-box говорит браузеру учитывать любые границы и внутренние отступы в значениях, которые указывашь в ширине и высоте элемента (например, в ширину будут включаться границы и отступы, а контент сожмётся, чтобы выделить для них место).

1. отличие flex-shrink и flex-grow

flex-shrink: определяет, как flex-элемент будет уменьшаться относительно других flex-элементов во flex-контейнере

flex-grow: определяет, как flex-элемент будет увеличиваться относительно других flex-элементов во flex-контейнере,

а flex-basis: определяет начальный размер flex-элемента

1. Позиционирования как на картинке можно добиться указав:

Display:flex;

Flex-direction:row:

Justify-content:space-between

1. Так как в devtools нет указания, что это border-box, то считаем, что это content-box по умолчанию, и тогда наша длина будет 2 128,55px, а высота будет 2 002,82px
2. Свойство order нужно, чтобы изменить порядковый номер и перенести флекс элемент. Элементы располагаются в восходящем порядке по значению order. Например, если нужно sidebar расположить слева от основного контента, напишем sidebar {order:1}
3. Верстка таблицы

<table border="1" style="width: 50%;">

<tr>

<td>Column 1</td>

<td>Column 2</td>

<td>Column 3</td>

</tr>

<tr>

<td rowspan="2">Row 1 Cell 1</td>

<td>Row 1 Cell 2</td>

<td>Row 1 Cell 3</td>

</tr>

<tr>

<td>Row 2 Cell 2</td>

<td>Row 1 Cell 3</td>

</tr>

<tr>

<td colspan="3">Row 3 Cell 1</td>

</tr>

</table>

1. Position:fixed походит, когда нужно нужно зафиксировать объект относительно окна (то есть при прокрутке странице элемент остается на своем месте и никуда не пропадает).
2. Теги с блочным боксом предназначены для крупных блоков текста (можно взять display:block или display:inline-block). Flexbox также тут удобен.
3. В целом размер флекс контейнера можно рассчитать как в обычной блочной модели- либо все отступы и бордеры суммируются с контентом и дают увеличенный размер, либо при указании border-box в ширину/высоту будут включаться отступы и бордеры.