1. Flexblox, обладая свойствами блочной верстки, дает большой инструментарий для позиционирования контента и распределения имеющегося места. То есть если мы работаем с не объемными элементами, содержащими большие тексты, с помощью этого метода легче разместить все элементы по странице, выделяя им необходимое количество места и они будут одинаково выглядеть на различных экранах и браузерах.
2. В целом, использование метода CSS для позиционирования display:table подходит для отображения большого контента с фиксированной шириной, например, сбоку страницы от элемента Flexblox. Элементы такой таблицы плохо реагируют на указание определенной ширины, то есть стремятся занять все пространство, независимо от контента. Помимо этого, ячейки принимают высоту самого большого элемента, для того чтобы rows располагались строго по упорядоченным линиям. Элементами таблиц сложно управлять, поэтому такая верстка плохо поддерживаемая.
3. При методе Flexblox:

* justify-content:center  по направлению главной оси
* align-content : center выравнивание линий контента

text-align:center центрирует инлайн-элементы в блоке

margin: 0 auto центрирует блок внутри родительского блока при заданной ширине

1. Главная и поперечная оси. Напрвление задается свойствами:

* Главной оси – flex-direction
* Поперечной оси – flex-wrap

В целом эти свойства задаются с помощью свойства flex-flow

1. Свойство Margin со значением auto позволяет элементу поглощать все свободное пространство вокруг. Если,например, установить Margin-left:auto, то это сдвинет элемент вправо на самый край, если добавить Margin-top:auto, то элемент будет в правом нижнем углу.
2. С помощью box-sizing мы можем регулировать размеры элементов (блочных и блочно-строчных), и настраивать высоту и ширину с учетом или без отступов. Т.е. не нужно высчитывать внутренние width height, а просто задать общие границы или границы с внутр. Отступом и бордером, ширина исчисляется автоматически.
3. Flex-grow обозначает, может ли ширина элемента расти, если увеличится размер родит. Блока, а flex-shrink- уменьшаться, если меняется размер всей страница в разных браузерах и экранах.
4. В flexbox с помощью justify-content:space-between
5. В соответствии с блочной моделью = ширина 148,523, высота 17,807
6. С помощью свойства order можно регулировать порядок блоков в плиточной раскладке, они будут расположены как column, но порядок будет идти в строчку, при этом высота элементов будет распределена равномерно, а не от верхнего начало каждой новой строки по объему содержимого.
7. Например, в HTML
8. <style>
9. table {
10. width: 500px;
11. height: 300px;
12. }
13. td {
14. border: 1px solid black;
15. }
16. </style>
17. </head>
18. <body>
19. <table>
20. <tr>
21. <td>Column 1</td>
22. <td>Column 1</td>
23. <td>Column 1</td>
24. </tr>
25. <tr>
26. <td rowspan="2">Row 1 Cell 1</td>
27. <td>Row 1 Cell 2</td>
28. <td>Row 1 Cell 3</td>
29. </tr>
30. <tr>
31. <td>Row 2 Cell 2</td>
32. <td>Row 2 Cell 3</td>
33. </tr>
34. <tr>
35. <td colspan="3">Row 3 Cell 1</td>
36. </tr>
37. </table>
38. </body>

12. Такой метод фиксирует элемент относительно окна и не дает ему двигаться, если, например, страница большая и нам необходимо ее прокручивать. Можно закрепить меню или навигацию, при прокручивании всегда будет видно эти элементы.

13. Стандартной семантич. Версткой блоками, в одном текстовом блоке можно регулировать расположение текста в редакционном формате или др, задавать классы и менять дизайн. Другие форматы данных выводить в отдельные блоки.

14. С помощью свойства flex-basis, если оно не указано, то по width блока, либо по контексту, если и ширина не задана.