Лекция №8

Разметка

СОДЕРЖАНИЕ

Интро

Методы CSS, которыми вы можете управлять разметкой элементов:

- **Свойство `display`** — Стандартные значения `block`, `inline` или `inline-block` могут изменять поведение элементов в обычном потоке (см.подробнее Types of CSS boxes). Также можно менять сами методы разметки такими значениями свойства `display`, как CSS Grid или Flexbox.
- **Floats** — Применение значения `float` типа `left` может заставить элемент блочного типа "прилепить" содержимое к одной стороне элемента, как иногда изображения обволакиваются текстом на газетных страницах.
- **Свойство `position`** — Позволяет точно контролировать положение блоков внутри других блоков. `static` позиционирование является стандартным, но также можно применять другие значения свойства, например фиксированное в углу экрана.
- **Макет Таблицы** — свойства для разметки таблиц могут быть использованы и для нетабличных элементов, с помощью `display: table` и соответствующих свойств.

Свойство display

Значения свойства `display` являются главными методами вёрстки разметки страницы в CSS. Это свойство позволяет нам менять то, как что-то отображается по умолчанию. Каждый элемент по умолчанию имеет свойство `display`, влияющее на то, как этот элемент отображается. Например, параграфы на английском располагаются один под другим только потому что они имеют по умолчанию свойство `display: block`. Если же вы создадите ссылку внутри параграфа, эта ссылка будет отображаться в общем потоке с остальным текстом, без переноса на новую строку. Это потому что у элемента `<a>` по умолчанию установлено свойство `display: inline`.

Flexbox

Flexbox (сокращение от Flexible Box Layout) это модуль, разработанный для облегчения вёрстки в одном из измерений — как ряд или как колонка. Для использования, установите свойство `display: flex` для родительского элемента тех элементов, к которым хотите применить этот тип вёрстки; все его прямые потомки станут flex элементами. Рассмотрим это на простом примере.

```
<div class="wrapper">
    <div class="box1">One</div>
    <div class="box2">Two</div>
    <div class="box3">Three</div>
</div>
```

One	Two	Three

```
.wrapper {
    display: flex;
}

.wrapper > div {
    flex: 1;
}
```

Grid Layout

В то время как flexbox предназначен для одномерной разметки, Grid Layout предназначен для двумерной — выстраивая предметы в ряды и столбцы.

Ещё раз, вы можете переключиться на Grid Layout при помощи конкретного значения отображения — `display: grid`. Пример ниже использует разметку подобную примеру flex, а также мы определяем некоторые дорожки рядов и столбцов в родительском элементе, используя свойства `grid-template-rows` и `grid-template-columns` соответственно. Мы определили три столбца каждый по `1fr` и два ряда по `100px`. Мне не надо вводить какие-либо правила для дочерних элементов; они автоматически помещаются в ячейки, созданные нашей сеткой.

```html
<div class="wrapper">
    <div class="box1">One</div>
    <div class="box2">Two</div>
    <div class="box3">Three</div>
</div>
```

Two

One

Three

```css
.wrapper {
    display: grid;
    grid-template-columns: 1fr 1fr 1fr;
    grid-template-rows: 100px 100px;
    grid-gap: 10px;
}

.box1 {
    grid-column: 2 / 4;
    grid-row: 1;
}

.box2 {
    grid-column: 1;
    grid-row: 1 / 3;
}

.box3 {
    grid-row: 2;
    grid-column: 3;
}
```

Floats

Делая элемент плавающим (float) мы меняем поведение этого элемента и элементов блочного уровня, следующих за ним в нормальном потоке. Элемент перемещается влево или вправо и удаляется из нормального потока (normal flow), а окружающий контент обтекает плавающий элемент.

Свойство `float` имеет четыре возможных значения:

- `left` — Элемент выравнивается слева и другие элементы обтекают его справа.
- `right` — Элемент выравнивается справа и другие элементы обтекают его слева.
- `none` — Не задаёт float совсем. Это значение по умолчанию.
- `inherit` — Определяет, что значение свойства `float` должно быть унаследовано от родительского элемента.

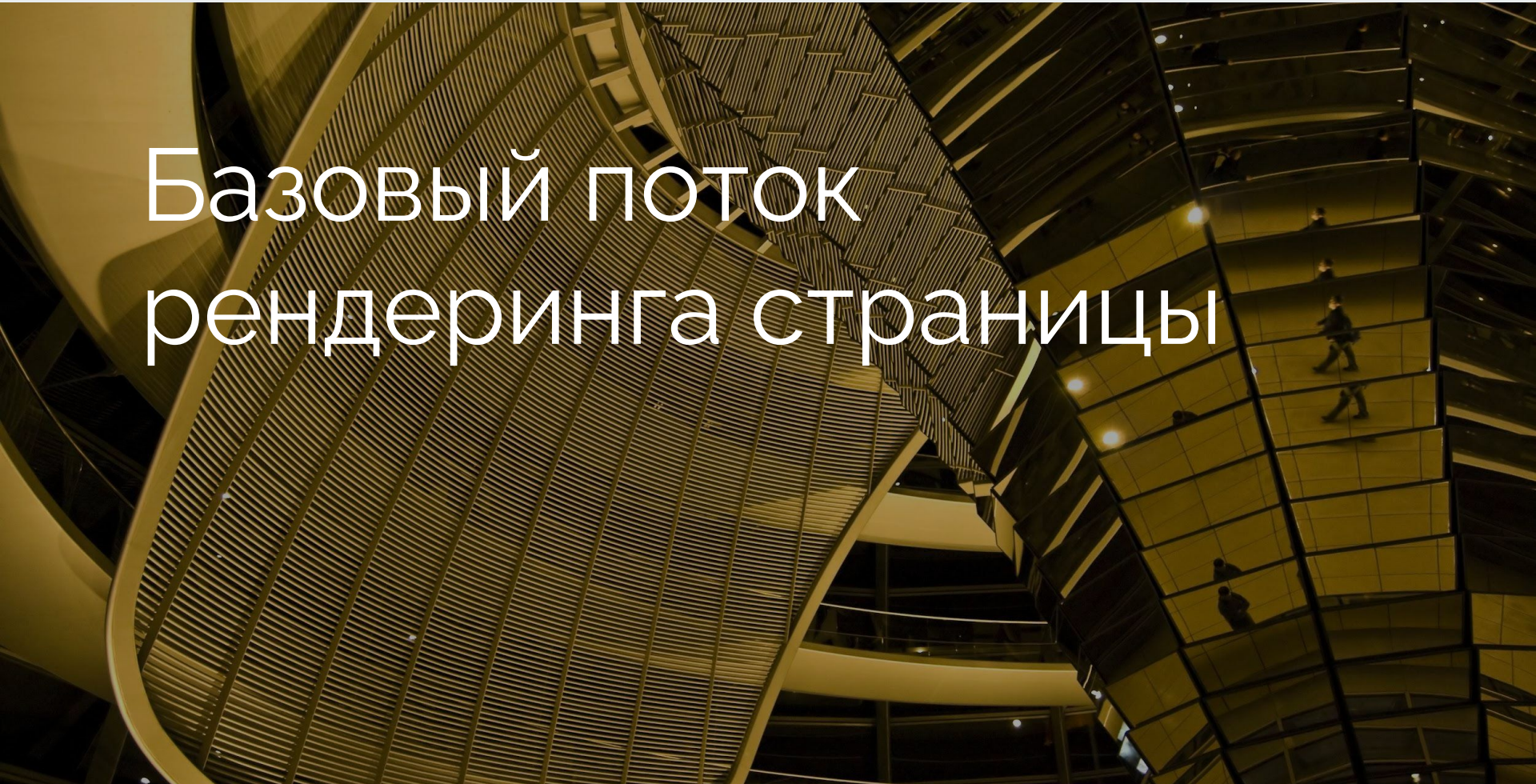
Методы позиционирования

Позиционирование позволяет вам перемещать элементы с места, где бы они располагались при нормальном потоке в другую локацию. Позиционирование не является методом создания основной разметки страницы, это больше об управлении и точной настройке положения определённых элементов на странице.

Однако, существуют полезные методы точной разметки шаблонов, которые полагаются на свойство `position`. Понимание позиционирования также способствует пониманию нормального потока и того, что значит вывести элемент из нормального потока.

Макет таблицы

HTML таблицы хороши для отображения табличных данных, но много лет назад — до того, как даже базовый CSS надёжно поддерживался в браузерах — веб-разработчики также использовали таблицы для разметки всей веб-страницы — размещая свои заголовки, нижние колонтитулы, различные колонки и т.д. в разных строках и столбцах таблиц. Это работало в то время, но оно имеет много проблем — разметка таблиц не гибкая, очень тяжёлая в вёрстке, сложна в отладке, и семантически не верная. (например, пользователи скринридеров имеют проблемы с навигацией в табличном макете).



Базовый поток рендеринга страницы

Как элементы располагаются по умолчанию?

Прежде всего, индивидуальные боксы элементов располагаются в зависимости от содержимого элементов, затем добавляя какой-нибудь padding, border и margin вокруг них - это опять-таки боксовая модель, которую мы рассмотрели ранее.

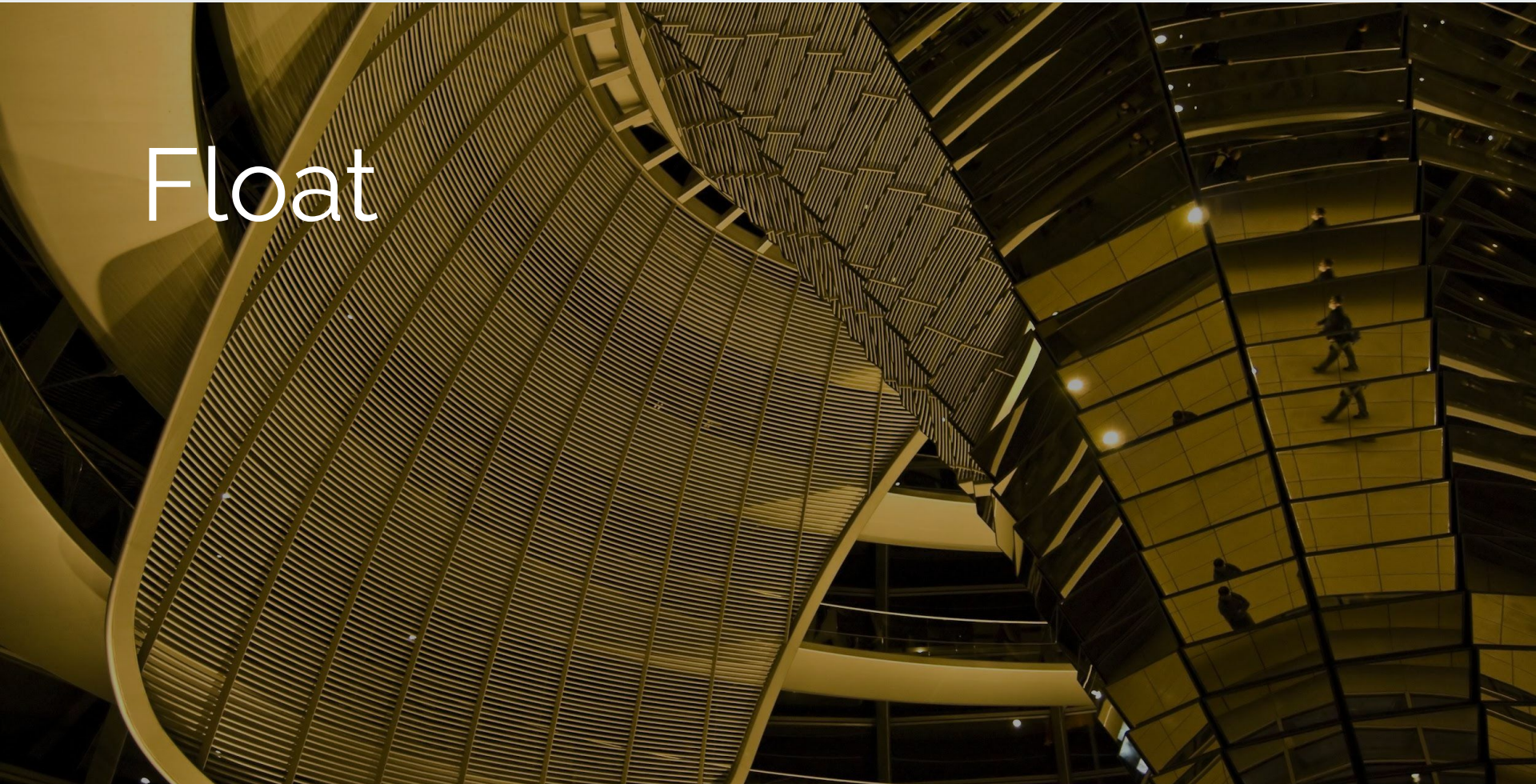
По умолчанию содержимое элемента уровня блока составляет 100% от ширины его родительского элемента и столь же высок, как и его содержимое. Строчные элементы высоки и широки, как их содержимое. Вы не можете установить ширину или высоту на строчные элементы — они просто находятся внутри содержимого элементов блочного уровня. Если вы хотите контролировать размер строчного элемента вам нужно настроить его так, чтобы он себя вёл как элемент блочного уровня при помощи `display: block;` (или даже, `display: inline-block;`, который смешивает характеристики обоих.).

Это объясняет отдельные элементы, но как насчёт того, как элементы взаимодействуют друг с другом? Нормальный поток макета (упомянутый в статье введения макета) - это система, посредством которой элементы размещаются внутри окна просмотра браузера. По умолчанию элементы уровня блока выкладываются в направлении, что блокирует отображение в режиме записи документа - каждый из них будет отображаться в новой строке ниже последней строки, и они будут разделены любым полем, установленным на них. Поэтому на английском языке или на любом другом, в котором режим писания горизонтальный, сверху вниз, элементы уровня блока располагаются вертикально.

Строчные элементы ведут себя по-другому — они не появляются на новых строках; они располагаются на той же строке, что и другие и любой смежной или завёрнутый текст располагается на всю ширину внутри элемента уровня родительского блока, до тех пор, пока не закончится пространство. Если пространства нет, тогда текст и/или элементы перейдут на новую строку (не с абзаца).

Если два смежных элемента имеют заданные для них поля/внешние отступы (margin) и эти поля соприкасаются друг с другом, большее из них остаётся, а меньшее исчезает — это называется схлопывание полей (margin collapsing), и мы рассматривали это ранее.

Float



Float

Первоначально используемое для "обтекания" картинок текстом, свойство `float` стало одним из наиболее часто используемых инструментов для создания макетов из нескольких столбцов на веб-страницах. С появлением flexbox и grid оно снова используется как задумывалось в начале, о чем подробнее в этой статье.

Общие сведения о float

Свойство `float` вводилось для того, чтобы разработчики могли включать изображение, с обтеканием текста вокруг него слева или справа, как это часто используется в газетах.

Но разработчики быстро осознали, что можно обтекать всё что угодно, не только изображения, поэтому использование float расширилось, например для вёрстки забавных эффектов таких как drop-caps (буквица).

Floats очень часто использовались для создания макетов целых веб-страниц, отображающих несколько колонок информации, обтекаемых так, что колонки располагаются друг за другом (поведение по умолчанию предполагает, что колонки должны располагаются друг за другом, в том же порядке в котором они появляются в источнике). Доступны более новые, лучшие методы и поэтому использование floats для этих целей следует рассматривать как устаревшей техникой.

Очистка обтекания

Мы увидели, что обтекаемый объект удалён из нормального потока и что другие элементы будут располагаться за ним, поэтому если мы хотим остановить перемещение следующего элемента нам необходимо очистить его; что достигается при помощи свойства `clear`.

Добавьте класс `cleared` ко второму параграфу после обтекаемого элемента в ваш HTML из предыдущего примера. Далее добавьте следующий CSS:

```
.cleared {
  clear: left;
}
```

Полезные ссылки

https://developer.mozilla.org/ru/docs/Learn/CSS/CSS_layout/Normal_Flow

https://developer.mozilla.org/ru/docs/Learn/CSS/CSS_layout/Introduction

https://developer.mozilla.org/ru/docs/Learn/CSS/CSS_layout/Floats

https://developer.mozilla.org/ru/docs/Learn/CSS/CSS_layout/Positioning

Задание

1. Создать шахматную доску с помощью элементов со свойством float

Спасибо!