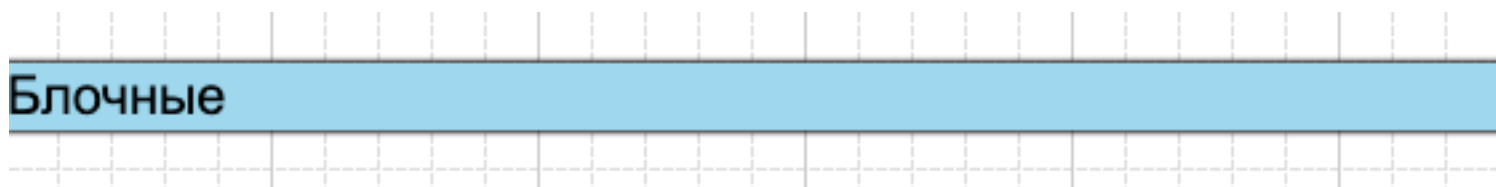


# Блочная Модель

Блочные элементы можно представлять как прямоугольные области на странице. Они имеют следующие особенности:

1. До и после блочного элемента существует перенос строки.
2. Блочным элементам можно задавать ширину, высоту, внутренние и внешние отступы.
3. Занимают всё доступное пространство по горизонтали.

К блочным элементам относятся такие теги как: `<p>`, `<h1>`, `<h2>`, `<ul>`, `<div>`.

A diagram illustrating a block element. It features a light blue rectangular box with a thin black border. The word "Блочные" is written in black text inside the box. The box is positioned on a grid of dashed lines, with solid vertical lines marking the start and end of the element's width. The word "Блочные" is positioned at the beginning of the element, demonstrating how it occupies the full available horizontal space.

Блочные

# Блочная Модель

Строчные элементы располагаются друг за другом в одной строке, при необходимости строка переносится. Особенности строчных элементов:

1. До и после строчного элемента отсутствуют переносы строки.
2. Ширина и высота строчного элемента зависит только от его содержания, задать размеры с помощью CSS нельзя.
3. Можно задавать только горизонтальные отступы.

К строчным элементам относятся такие теги как: `<a>`, `<b>`, `<i>`, `<span>` и так далее.

Строчные элементы предназначены для оформления текста на уровне небольших фраз и отдельных слов. Блочные же элементы предназначены для разметки крупных блоков текста (заголовки, абзацы, списки) и создания сетки.

Строчные элементы не создают переносов строки до и после себя. Такие элементы располагаются в строке слева направо. Если строчный элемент не помещается в родительский контейнер, то он переносится на следующую строку.

# Ширина и высота

По умолчанию блочные элементы занимают всю доступную ширину, которая равна ширине родительского контейнера или окна браузера.

Высота по умолчанию блочных элементов зависит от их содержимого. Если задать блочному элементу ширину и высоту так, что содержимое элемента не будет в него помещаться, то оно как бы «выпадет» из него.

Строчные элементы не реагируют на задание ширины и высоты в CSS.

```
.selector {  
  width: 100px;  
  height: 100px;  
}
```

# Внутренние отступы (padding)

padding-top

padding-right

padding-bottom

padding-left

padding: 15px 25px 30px 15px;

padding: 15px 25px 30px;

padding: 35px 20px;

padding: 20px;

# Рамки (border)

border-width

border-style `none` | `dotted` | `solid` | `dashed`

border-color

border-radius

Универсальное свойство:

`border: 2px solid #000000;`

`border-right: 5px dotted #ff0000;`

# Внешние отступы (margin)

margin-top

margin-right

margin-bottom

margin-left

margin: 15px 25px 30px 15px;

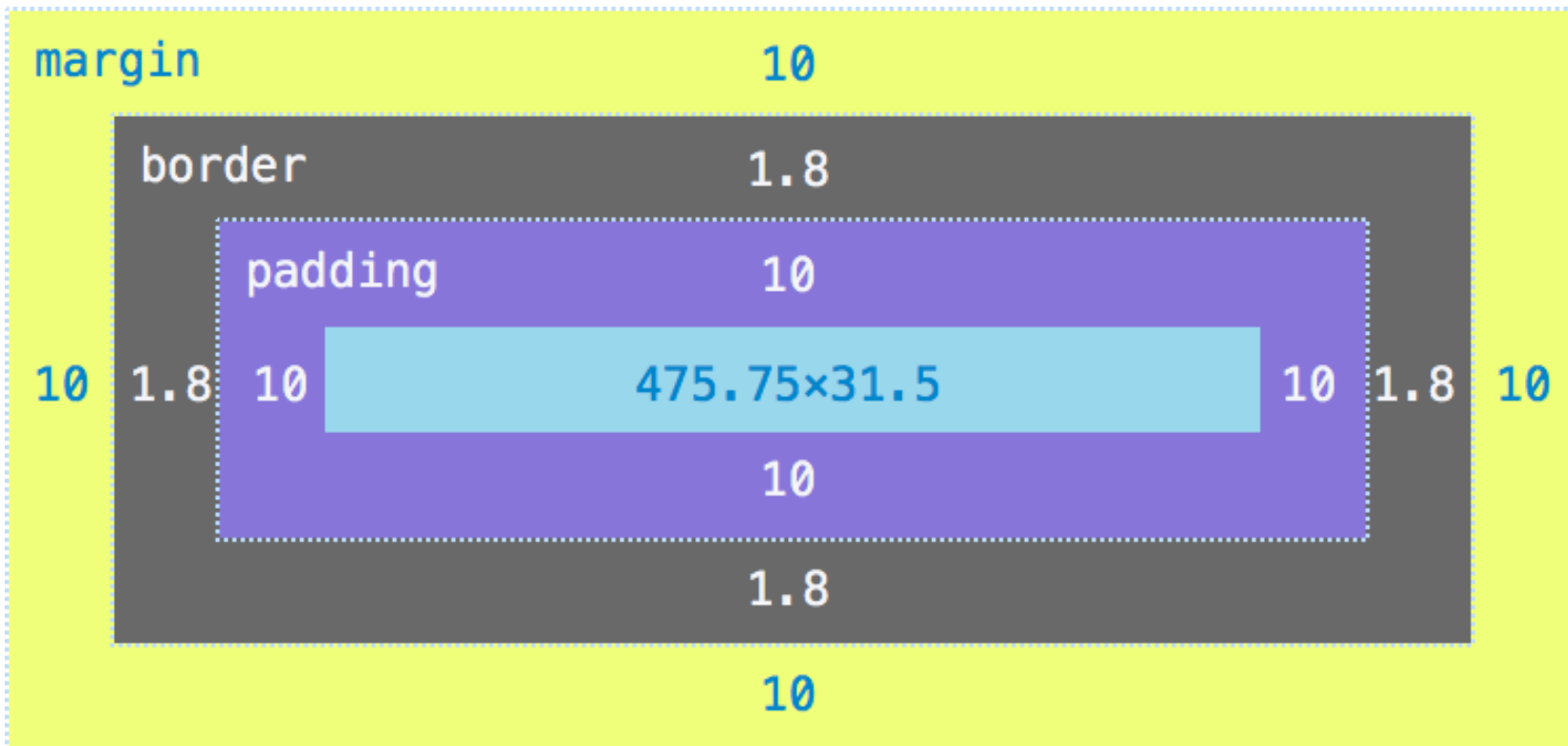
margin: 15px 25px 30px;

margin: 35px 20px; / margin: 10px auto; / margin: auto;

margin: 20px;

# Стандартная блочная модель

Область, занимаемая *блочным* элементом, складывается из его ширины и высоты содержания, внутренних и внешних отступов, ширины рамок



# Вертикальное выравнивание

`vertical-align:middle;`

Выравниванием текста по вертикали можно управлять с помощью свойства `vertical-align`. Его действие хорошо заметно в ячейках таблицы. Внутри текстовой строки «работа» этого свойства заметна, если в ней есть фрагменты разного размера.

У данного свойства много значений, но самые часто используемые:

- `top` — выравнивание по верхнему краю строки;
- `middle` — по середине;
- `bottom` — по нижнему краю;
- `baseline` — по базовой линии (значение по умолчанию).



# Схлопывание (margin)

*Вертикальный* отступ между двумя соседними элементами равен максимальному отступу между ними. Если отступ одного элемента равен 20px, а второго 40px, то отступ между ними будет 40px.

Этот эффект называется эффектом «схлопывания» внешних отступов или «схлопывания» маргинов.

*Горизонтальные* отступы между элементами просто складываются. Например, горизонтальный отступ между двумя элементами с отступами 30px будет равен 60px.

# Выпадение (margin)

«Выпадение» — это еще один эффект, связанный с вертикальными внешними отступами. Если внутри родительского блока расположить блок и задать ему отступ сверху, то внутренний блок прижмется к верхнему краю родительского, а у родительского элемента появится отступ сверху. То есть верхний отступ внутреннего элемента «выпадает» из родительского элемента.

Если у родительского элемента тоже был задан внешний отступ, то выберется максимальный отступ между собственным и «выпавшим» .

# Центрируем блочный элемент

1. Задать элементу ширину, которая меньше ширины родительского контейнера.
2. Задать для внешних отступов справа и слева значение auto.

```
selector {  
  width: 100px;  
  margin: 0 auto;  
}
```

```
selector {  
  width: 100px;  
  margin-left: auto;  
  margin-right: auto;  
}
```

# Блочная модель vs Строчный элементы

1. Не реагируют на CSS-свойства `width` и `height`.
2. Частично реагируют на `margin`, воспринимая только горизонтальные отступы.
3. Частично реагируют на `padding`, воспринимая только горизонтальные отступы.
4. При задании вертикальных `padding` визуально увеличиваются, но без увеличения занимаемого места (не отталкивают другие элементы).
5. Воспринимают рамки. Аналогично `padding` рамки сверху и снизу не увеличивают занимаемое элементом место.

# Проблемы стандартной блочной модели

## Ширина 100% и ширина по умолчанию

Поведение элемента может зависеть от того, как именно вы зададите его ширину.

**Первый вариант.** Вариант по умолчанию, когда ширина не задается, соответствует значению `width: auto;`. В этом случае блок занимает всю ширину родительского блока. Если у блока есть внутренние отступы или рамки, то его *ширина содержания* автоматически уменьшается, а общая ширина остается равной ширине родителя.

**Второй вариант.** Когда ширина блока задана явно, например, `width: 100%;`. В этом случае *ширина содержания* блока равна ширине родительского блока. Если блоку добавить внутренние отступы и рамки, то его общая ширина становится больше ширины родителя.

## Изменяем блочную модель, свойство box-sizing

Это свойство имеет два значения:

1. `content-box` - значение по умолчанию, соответствует стандартной блочной модели.
2. `border-box` - свойство `width` задает не ширину содержания, а общую ширину включает в себя `padding` и `border`

# Свойство display

Управление типом элемента

inline

inline-block

block

none

table / inline-table

list-item

flex / inline-flex

grid / inline-grid

# Display: inline-block;

Особенности блочно-строчных элементов:

- им можно задавать размеры, рамки и отступы, как и блочным элементам;
- их ширина по умолчанию зависит от содержания, а не растягивается на всю ширину контейнера;
- они не порождают принудительных переносов строк, поэтому могут располагаться на одной строке, пока помещаются в родительский контейнер;
- элементы в одной строке выравниваются вертикально подобно строчным элементам.

# Table

Table > caption > tr > th/td

Группировка разделов таблицы - <thead> / <tbody> / <tfoot>

## Table + css

```
table {  
    border: 1px solid grey;  
    border-collapse: separate / collapse;  
    border-spacing: px / rem / em / %;  
    empty-cell: hide;  
}
```



# Display: table;

```
table { display: table }
tr { display: table-row }
thead { display: table-header-group }
tbody { display: table-row-group }
tfoot { display: table-footer-group }
td, th { display: table-cell }
caption { display: table-caption }
```

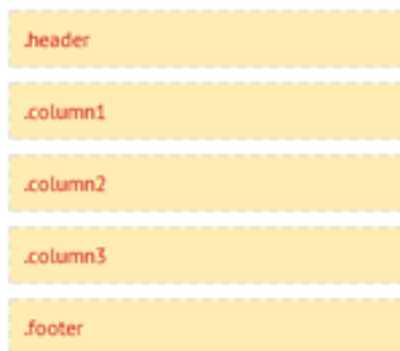
# Поток документа

Поток — это порядок отображения элементов на странице. По умолчанию блочные элементы отображаются как прямоугольные области, идущие друг за другом сверху вниз, а строчные элементы располагаются сверху вниз и слева направо и при необходимости переносятся на новую строку.

Исходный код

```
<div class="header"></div>  
<div class="column1"></div>  
<div class="column2"></div>  
<div class="column3"></div>  
<div class="footer"></div>
```

Обычный поток



Изменённый поток

