Веб

История развития ИНТЕРНЕТ

Ученые под руководством которых была создана сеть ARPANet

Джозеф Карл Робнетт Ликлайдер

Роберт Вильям Тейлор

Ларри Робертс (79 лет)

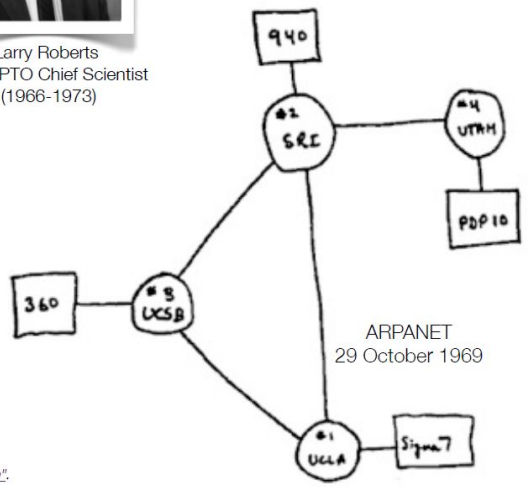
J.C.R. Licklider
ARPA IPTO Director
(1962-1964)

Robert Taylor
ARPA IPTO Director
(1966-1969)

Larry Roberts
ARPA IPTO Chief Scientist
(1966-1973)

"[...] we are entering a technological age in which we will be able to interact with the richness of living information - not merely in the passive way that we have been accustomed to using books and libraries, but as active participants in an ongoing process, bringing something to it through our interaction with it, and not simply receiving something from it by our connection to it."

J.C.R. Licklider, Robert Taylor: "The Computer as a Communication Device". Science and Technology 76, pp. 21-31 April 1968.



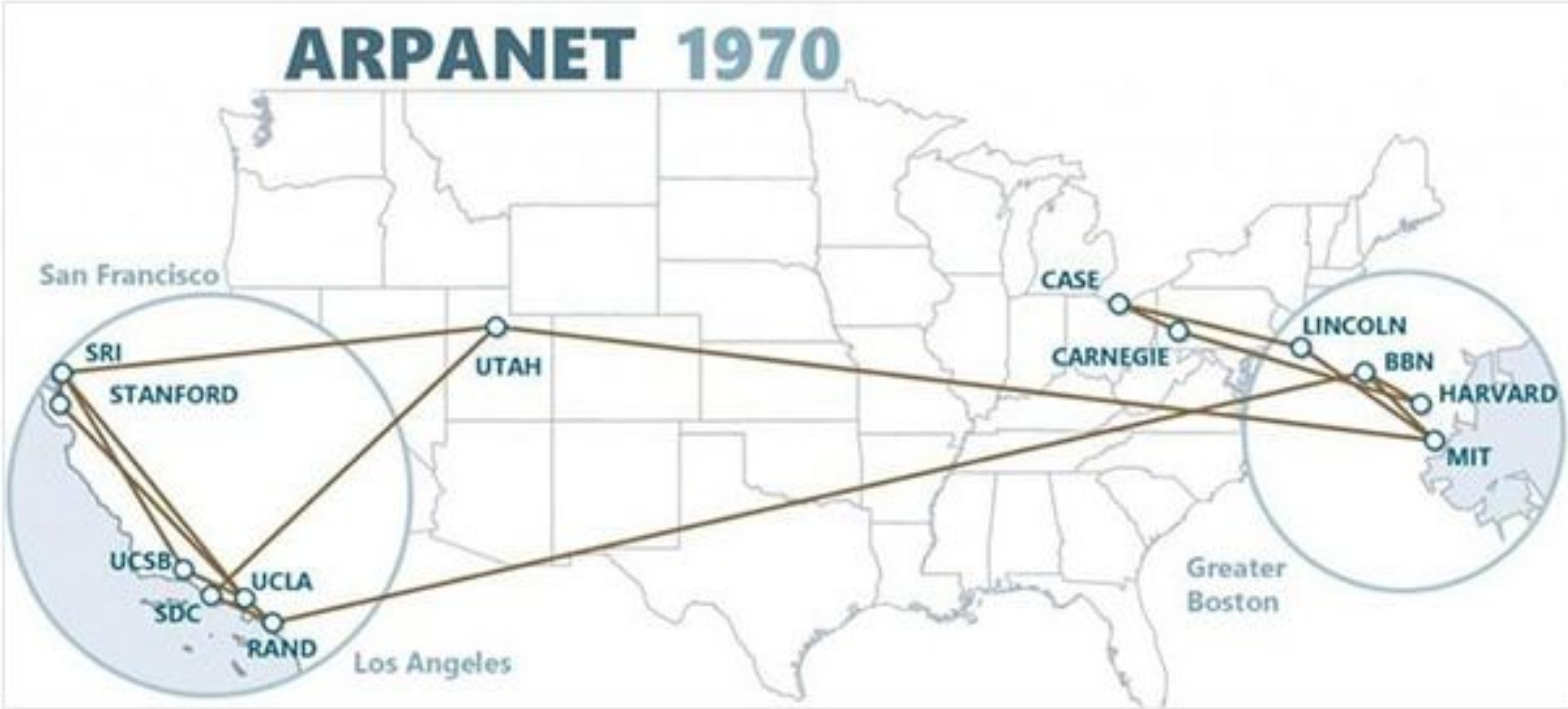
ARPANET
29 October 1969

Краткая история

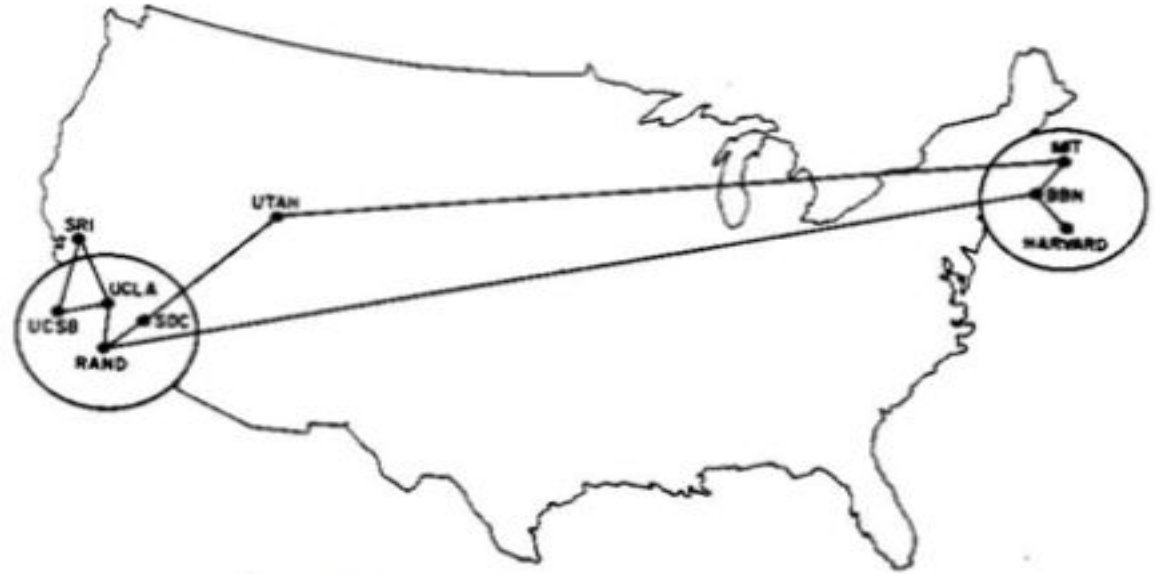
- 1962 Дж.К.Р. Ликлайдер озвучил мысль о так называемой "галактической сети" из взаимосвязанных компьютеров
- 1966 эксперименты с коммутацией пакетов управления ARPA
- 1967 Лоуренс Робертс из Массачусетского технологического института опубликовал план создания ARPANET
- 1969 первые работоспособные узлы сети ARPANET
- 1972 технология была впервые представлена широкой публике на Международной конференции по компьютерным коммуникациям. В этом же году было отправлено первое сообщение по электронной почте

Краткая история

- 1973 первые компьютеры, подключенные к сети ARPANET за пределами США
- 1975 сеть ARPANET передана в ведение управления связи министерства обороны США
- 1980 начинаются эксперименты с TCP/IP
- 1981 каждые 20 дней к сети добавляется новый хост
- 1983 завершен переход на TCP/IP
- 1986 создана магистраль NSFnet
- 1990 сеть ARPANET прекратила существование
- 1995 приватизация магистрали Интернета

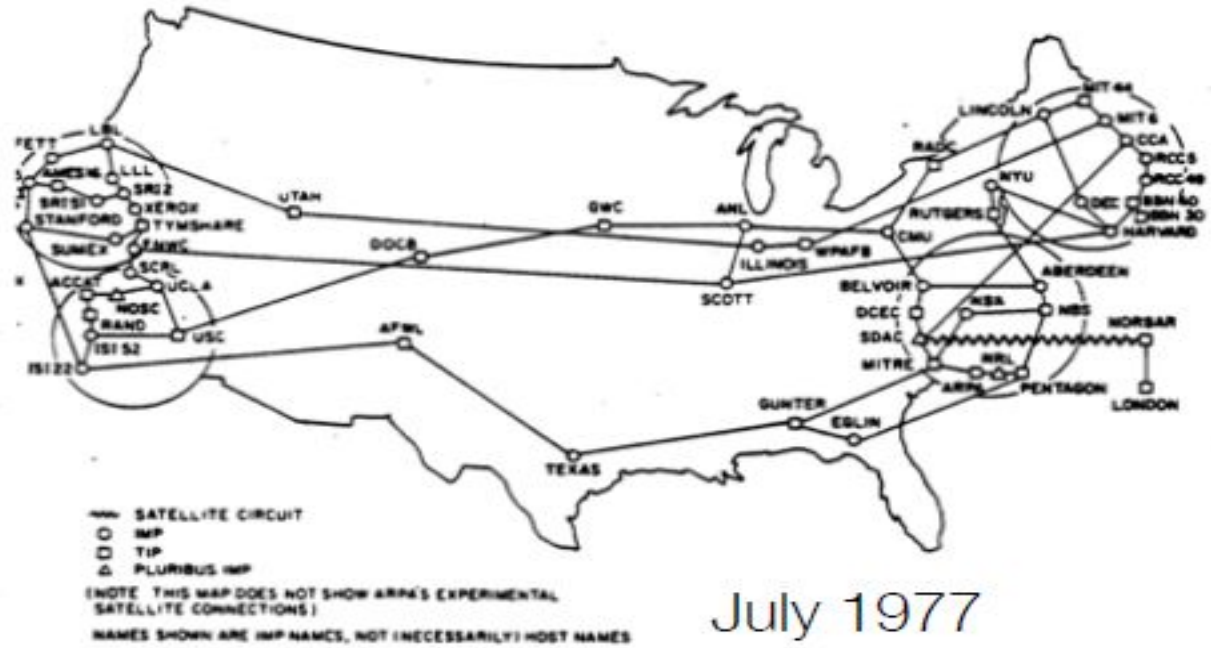

ARPANET 1970
San Francisco
SRI
STANFORD
UTAH
UCSB
UCLA
SDC
RAND
Los Angeles
CASE
CARNEGIE
LINCOLN
BBN
HARVARD
MIT
Greater
Boston

December 1969



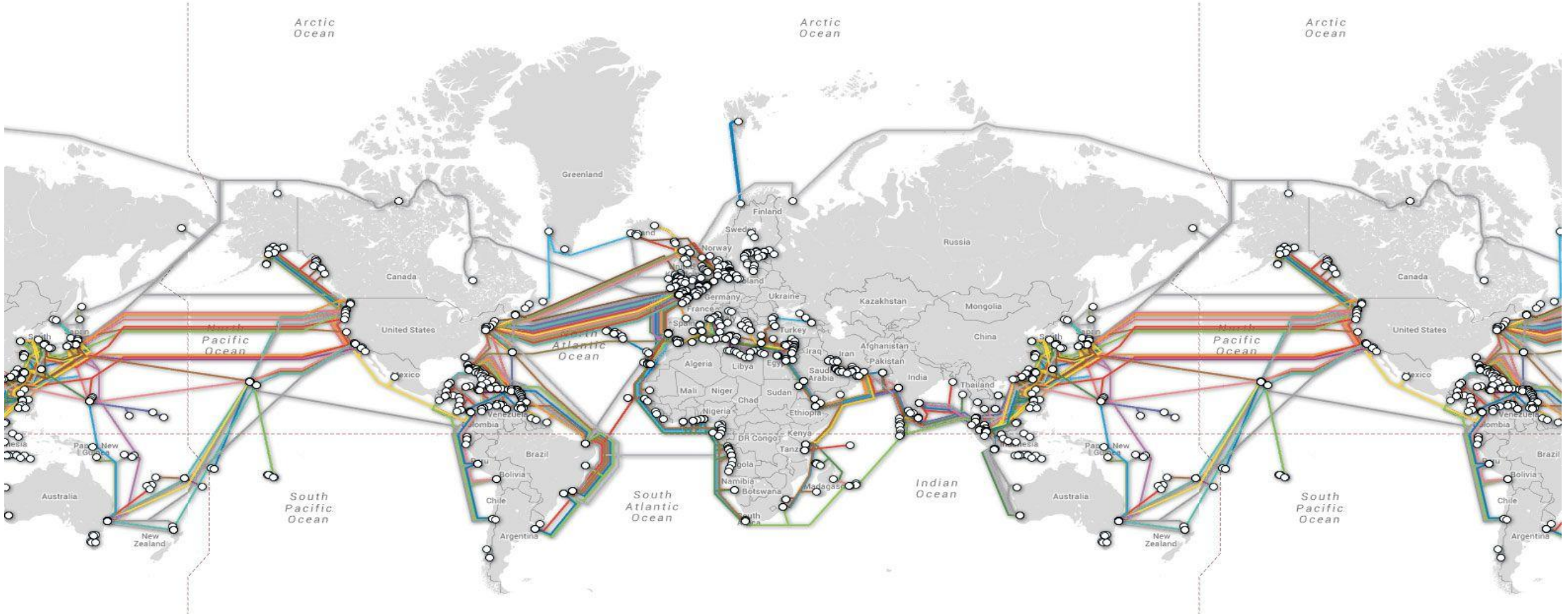
June 1970

March 1972

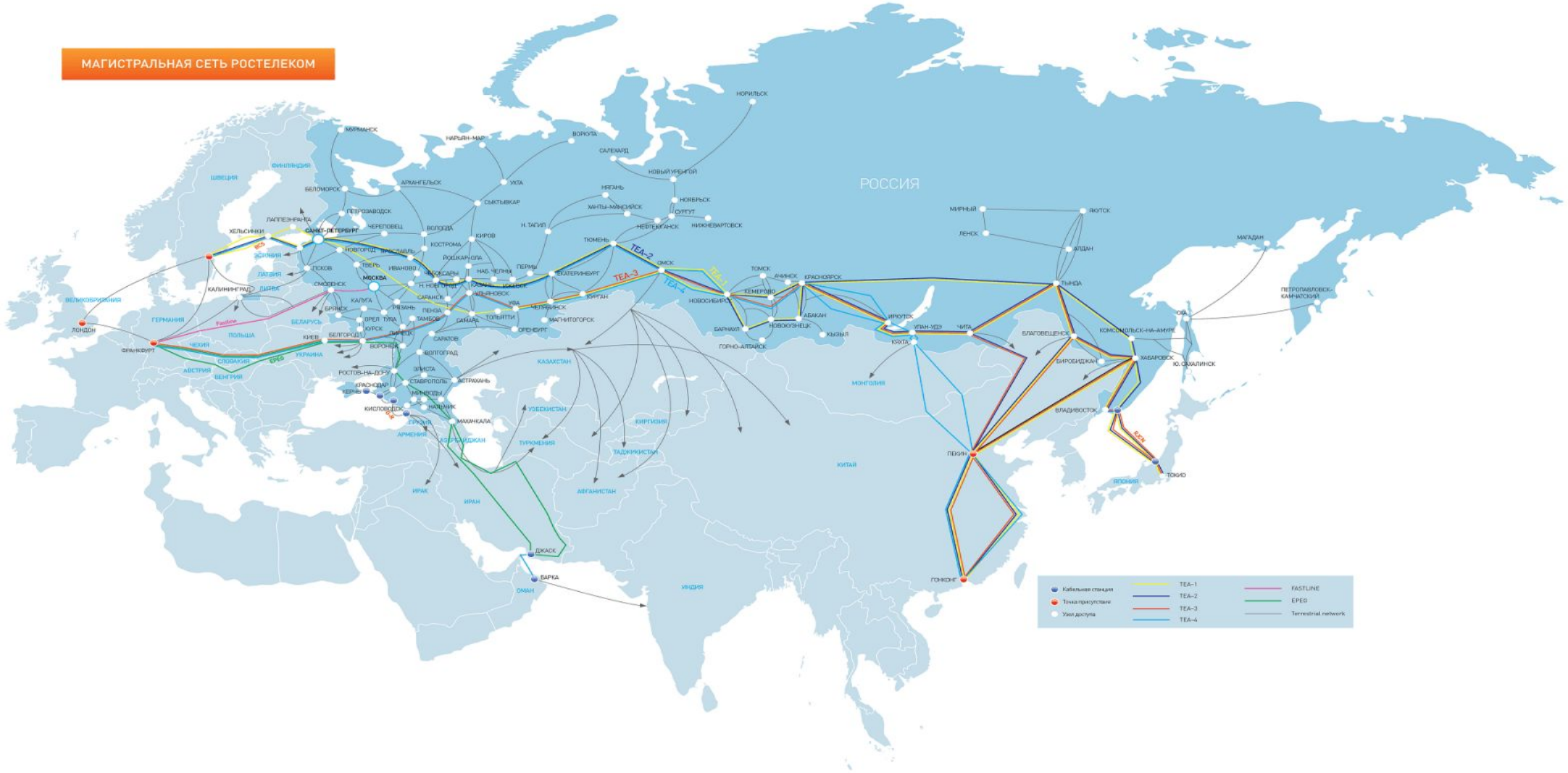


July 1977

Подводная магистральная сеть Интернет



Сеть РОСТЕЛЕКОМ



Число пользователей ИНТЕРНЕТ (2022 год)

Китай — 765 миллионов

Индия – 391 миллион

США – 245 миллионов

Россия – 130 миллионов

Бразилия – 126 миллионов

Япония – 116 миллионов

Мексика – 75,94 млн.

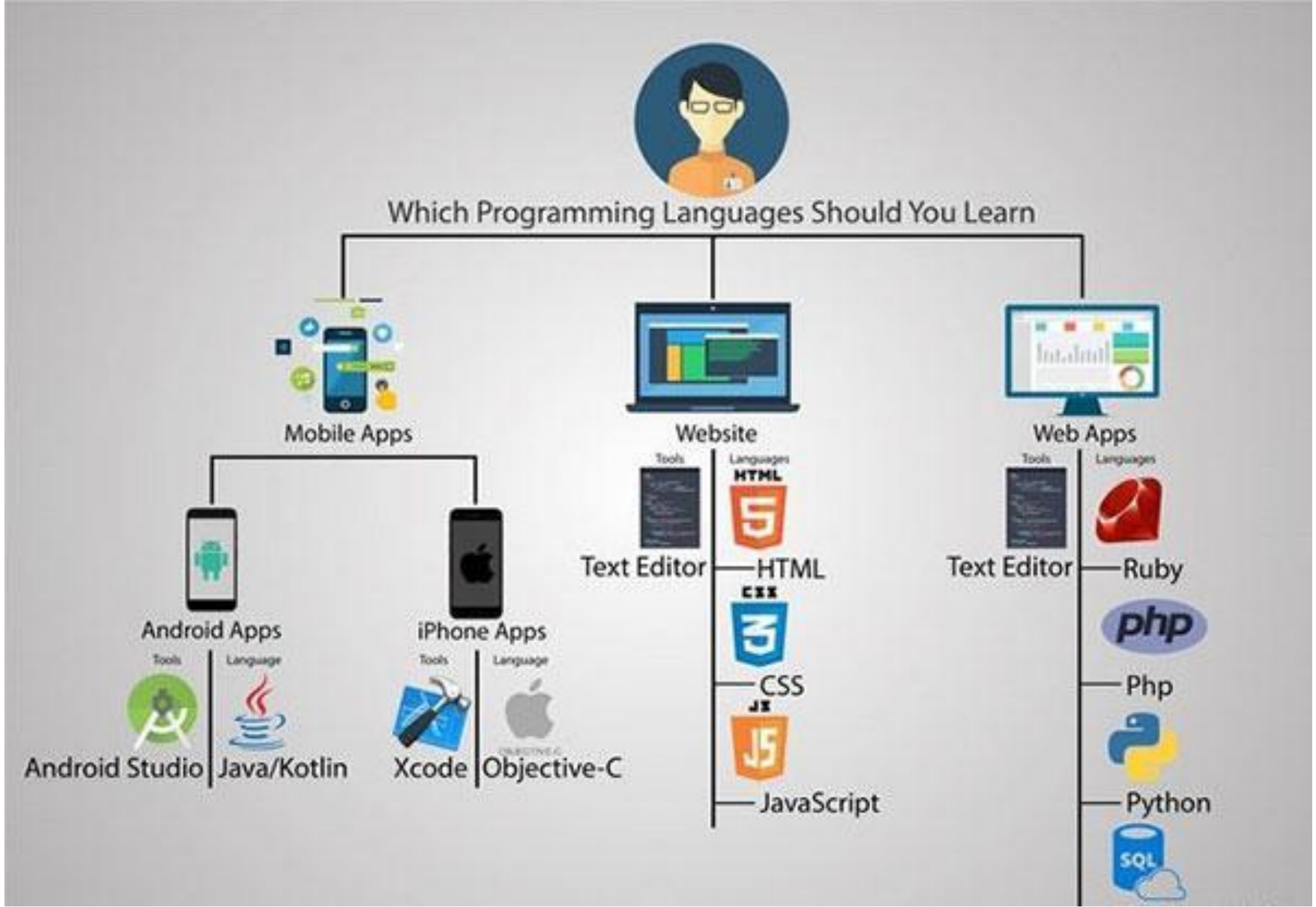
Германия – 73,43 млн.

Индонезия – 66,45 млн. Великобритания – 62,35 млн.

Скорость интернета по странам мира в 2022 году

1. Чили 197,59 2 мб/с
2. Сингапур 194.07 мб/с
3. Монако 188,66 4 мб/с
4. Таиланд 182,96 мб/с
5. Дания 164,14 мб/с
6. Гонконг (Китай) 163,37 мб/с
7. Макао (Китай) 159,89 мб/с
8. Китай 155,87 мб/с
9. Соединенные Штаты 146,17 мб/с
10. Румыния 136,46 мб/с

Веб-разработка


Which Programming Languages Should You Learn
Mobile Apps
Android Apps
Tools
Language
Android Studio
Java/Kotlin
iPhone Apps
Tools
Language
Xcode
Objective-C
Website
Tools
Languages
Text Editor
HTML
CSS
JavaScript
Web Apps
Tools
Languages
Text Editor
Ruby
Php
Python
SQL

HTML 5

Введение

HTML - Язык гипертекстовой разметки

HTML-документ — это обычный текстовый документ, может быть создан как в обычном текстовом редакторе (Блокнот), так и в специализированном, с подсветкой кода (Notepad++, Visual Studio Code и т.п.). HTML-документ имеет расширение .html

Структура парных тегов:

<Открывающий ТЕГ>Содержимое</Закрывающий ТЕГ>

Текст

Структура непарных тегов:

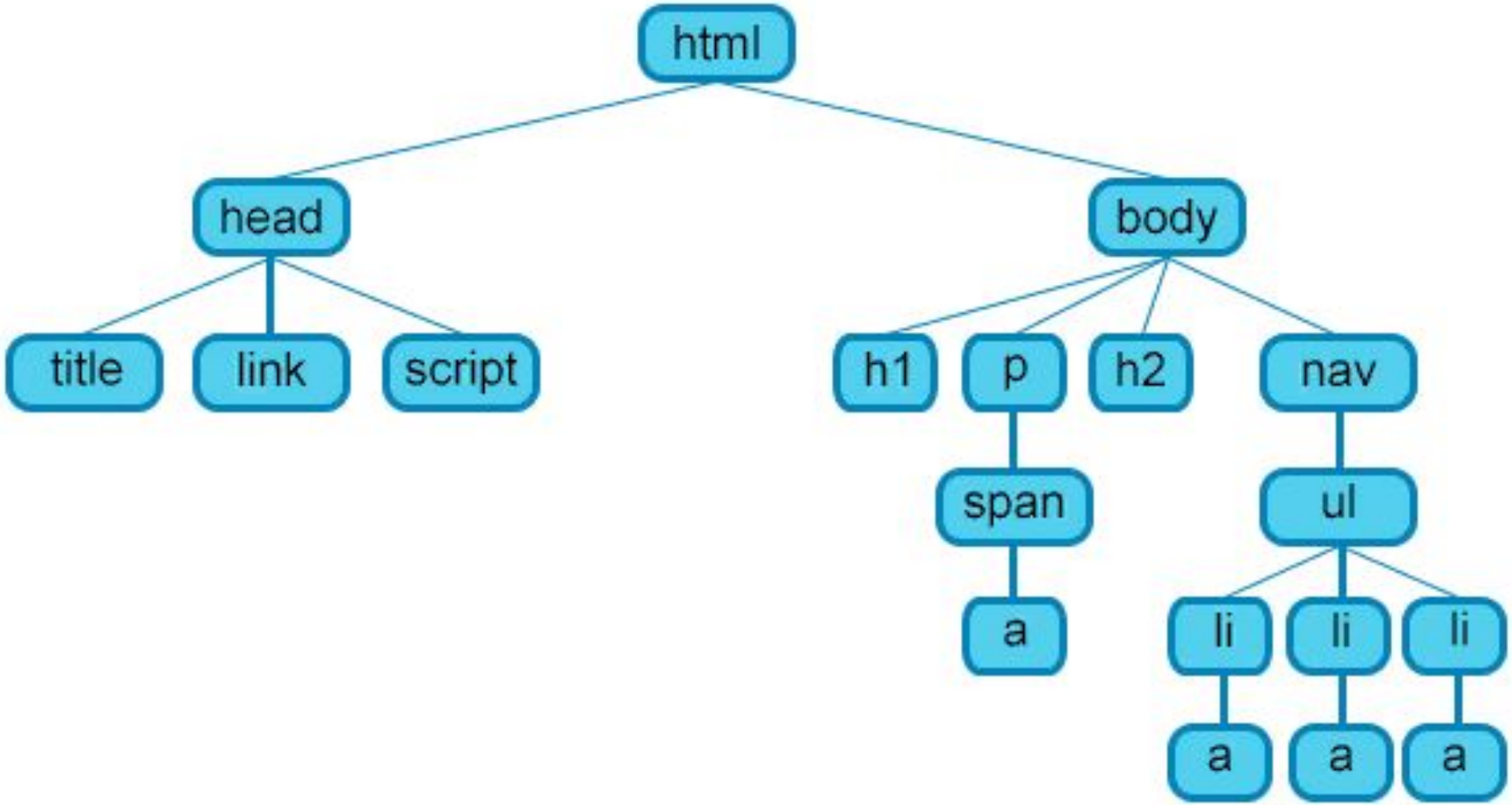
<Открывающий ТЕГ>

Введение. Возможности

- Вложенность элементов ** Текст **
- Наличие атрибутов: глобальные, собственные **имя атрибута=" значение"**
- Значения элементов: class (множественное использование) в рамках одного элемента, id (единичное использование) в рамках одного элемента

Введение. Структура HTML-документа

```
<!DOCTYPE html> <!-- Объявление формата документа -->
<html>
  <head> <!-- Техническая информация о документе -->
    <meta charset="UTF-8"> <!-- Определяем кодировку символов документа -->
    <title>...</title> <!-- Задаем заголовок документа -->
    <link rel="stylesheet" type="text/css" href="style.css"> <!-- Подключаем внешнюю таблицу стилей -->
    <script src="script.js"></script> <!-- Подключаем сценарии -->
  </head>
  <body> <!-- Основная часть документа -->
  </body>
</html>
```


html
head
body
title
link
script
h1
p
h2
nav
span
ul
a
li
li
li
a
a
a

HTML-элементы

- **пустые элементы** — <area>, <base>, <br>, <col>, <embed>, <hr>, <img>, <input>, <link>, <menuitem>, <meta>, <param>, <source>, <track>, <wbr>;
- **элементы с неформатированным текстом** — <script>, <style>;
- **элементы, выводящие неформатированный текст** — <textarea>, <title>;
- **элементы из другого пространства имён** — MathML и SVG;
- **обычные элементы** — все остальные элементы.

Метаданные документа		
<html></html>	корневой элемент html-документа	block
<head></head>	контейнер для метаданных html-документа	none
<title></title>	заголовок / имя html-документа	none
<base>	базовый url-адрес, относительно которого вычисляются относительные адреса	none
<link>	подключает внешние таблицы стилей	none
<meta>	мета-данные веб-страницы	none
<style></style>	подключает встраиваемые таблицы стилей	none

Секционные элементы		
`<body></body>`	тело html-документа	`block`
`<article></article>`	раздел контента, образующий независимую часть документа или сайта	`block`
`<section></section>`	логическая область (раздел) страницы, обычно с заголовком	`block`
`<nav></nav>`	раздел документа, содержащий навигационные ссылки по сайту	`block`
`<aside></aside>`	контент страницы, имеющий косвенное отношение к основному контенту	`block`
`<h1></h1> - <h6></h6>`	заголовки шести уровней	`block`
`<header></header>`	секция для вводной информации сайта или группы навигационных ссылок	`block`
`<footer></footer>`	секция для нижнего колонтитула документа или раздела	`block`

HTML-атрибуты

HTML-атрибуты это специальные слова, которые управляют поведением HTML-элемента. Они добавляют дополнительную функциональность

Атрибуты могут быть указаны четырьмя различными способами:

- имя атрибута, например, disabled
- значение атрибута без кавычек, например, autocapitalize=sentences
- значение атрибута в одинарных кавычках, например, type='checkbox'
- значение атрибута в двойных кавычках, например, class="external icon-link"

HTML-текст

- HTML-элементы для заголовков <h1>...<h6>
- Элементы для форматирования текста
  - **<b>** полужирное начертания шрифта
  - **** отображает шрифт курсивом
  - **<i>** отображает шрифт курсивом
  - **<small>** уменьшает размер шрифта на единицу
  - **** полужирное начертание шрифта
  - **<sub>** создание нижних индексов
  - **<sup>** создания степеней

HTML-текст

- Элементы для ввода «компьютерного» текста
  - <code> выделение фрагментов программного кода
  - <kbd> отмечает фрагмент как вводимый пользователем с клавиатуры
  - <samp> выделения результата, полученного в ходе выполнения программы
  - <var> Выделяет имена переменных, отображая текст курсивом
- Элементы для оформления цитат и определений
  - <abbr> применяется для форматирования аббревиатур
  - <bdo> используется для изменения текущего направления текста
  - <dfn> позволяет выделить текст как определение

HTML-текст

- Абзацы, средства переноса текста
  - <p> разбивает текст на отдельные абзацы
  -
 переносит текст на следующую строку
  - <hr> разделение контента на веб-странице в виде горизонтальной линии

HTML-ссылки

- HTML-ссылки создаются с помощью элементов
  - <a>,
  - <area>
  - <link> - ссылки на внешние ресурсы

HTML-ссылки

- Структура ссылки указатель ссылки
- URI метод доступа://имя сервера:порт/путь
  - file:/gallery/pictures/summer.html
  - http://site.ru/
  - https://site.ru/
  - ftp://pgu/directory/library
  - mailto: nika@gmail.com

HTML-ссылки

Номер порта TCP, на котором функционирует веб-сервер. Представляет собой число, которое необходимо указывать, если метод требует номер порта (отдельные сервера могут иметь свой отличительный номер порта). Если порт не указан, по умолчанию используется порт 80. Стандартными портами являются:

21 — FTP

23 — Telnet

70 — Gopher

80 — HTTP

HTML-ссылки

Абсолютный путь указывает точное местоположение файла в пределах всей структуры папок на компьютере (сервере).

- протокол, например, http (опционально);
- домен (доменное имя или IP-адрес компьютера);
- папка (имя папки, указывающей путь к файлу);
- файл (имя файла).
  - http://site.ru/pages/tips/tips1.html - с указанием протокола
  - //site.ru/pages/tips/tips1.html - без указания протокола

HTML-ссылки

Относительный путь описывает путь к указанному документу относительно текущего.

Относительный путь содержит следующие компоненты:

- папка (имя папки, указывающей путь к файлу);
- файл (имя файла).

Путь для относительных ссылок имеет три специальных обозначения:

- / указывает на корневую директорию и говорит о том, что нужно начать путь от корневого каталога документов и идти вниз до следующей папки
- ./ указывает на текущую папку
- ../ подняться на одну папку (директорию) выше

HTML-ссылки

Якоря, или внутренние ссылки, создают переходы на различные разделы текущей веб-страницы, позволяя быстро перемещаться между разделами

- <h1>Времена года</h1>
- <h2>Оглавление</h2>
- Лето <!--создаём якорь, указав #id элемента-->
- Осень
- Зима
- Весна
- ... <!--добавляем соответствующий id элементу-->
- ...
- ...
- ...

HTML-ссылки

Изображения-ссылки.

Чтобы сделать кликабельное изображение, необходимо поместить элемент <img> внутрь элемента <a>. Чтобы ссылка открывалась в другом окне, нужно добавить атрибут target="_blank" для ссылки.

- <img src="//html5book.ru/wp-content/uploads/2014/07/Fringe.jpg" alt="Fringe">

HTML-ссылки

- **ссылка на телефонный номер** +7 (495) 123-45-67
- **ссылка на адрес электронной почты** example@mail.ru
- **ссылка на скайп (позвонить)** Skype
- **ссылка на скайп (открыть чат)** Skype
- **ссылка на скайп (добавить в список контактов)** Skype
- **ссылка на скайп (отправить файл)** Skype

HTML-изображения

<img> - строчный элемент, рекомендуется располагать внутри блочного для правильного позиционирования

- <img src="image.png" alt="Пример кода">
- Адрес изображения: абсолютный или относительный
  - url(http://anysite.ru/images/anyphoto.png)
  - url(../images/anyphoto.png)
- Размеры изображения. Без задания размеров изображения рисунок отображается на странице в реальном размере
  - атрибуты width и height
- Форматы графических файлов: JPEG, GIF, PNG, APNG, SVG, BMP, ICO

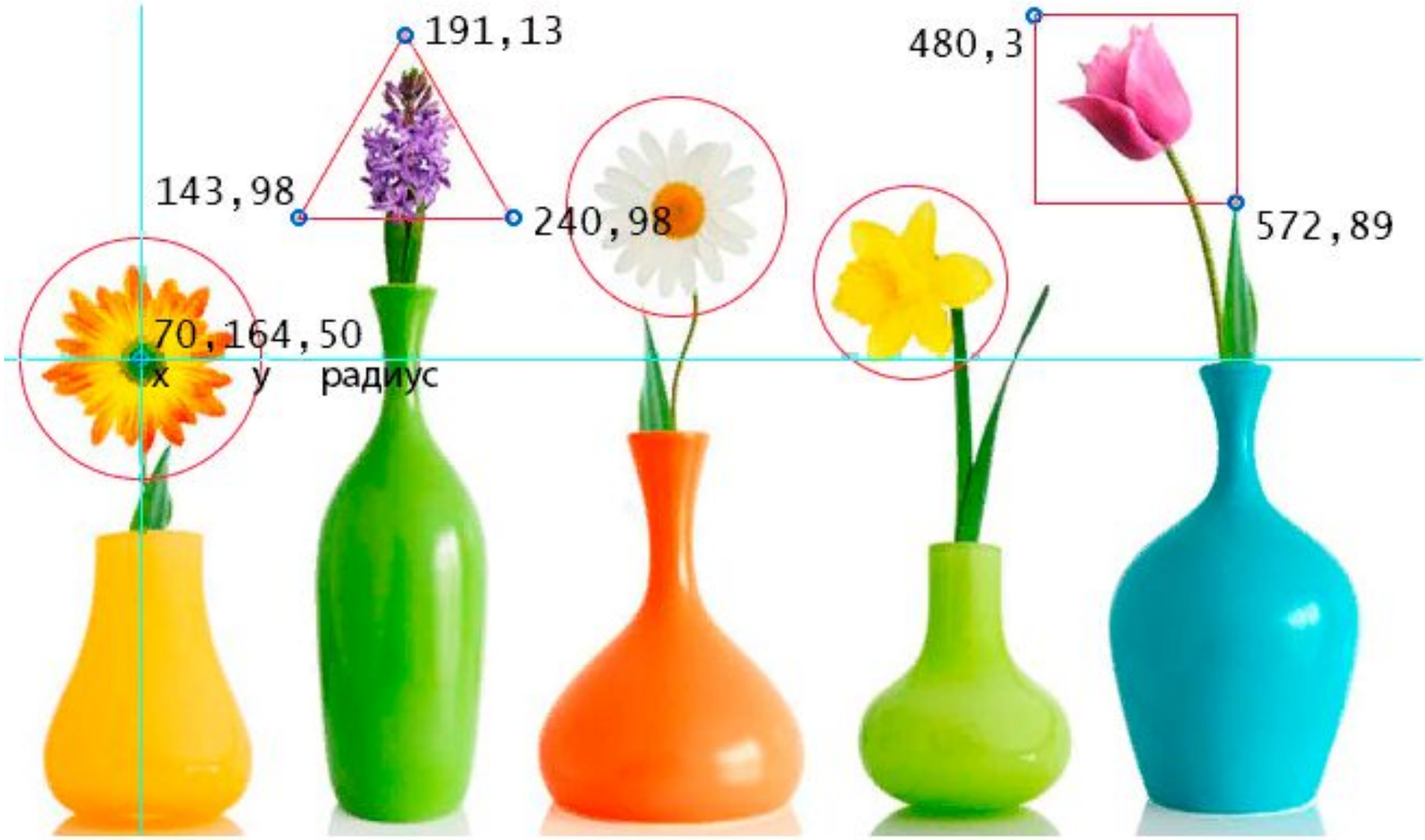
HTML-изображения

<map> - служит для представления графического изображения в виде карты с активными областями

<area> - описывает только одну активную область в составе карты изображений на стороне клиента

```
<img src="url" usemap="#имя_карты">
<map name="имя_карты">
<area атрибуты>
</map>
```

HTML-изображения



Пример создания карты-изображения

- Размечаем исходное изображение на активные области нужной формы. Координаты областей можно вычислить с помощью программы для обработки фотографий, например, Adobe Photoshop или Paint.
- Задаем имя карты, добавив ее в элемент <map> с помощью атрибута name. Это же значение присваиваем атрибуту usemap элемента <img>.
- Добавляем ссылки на веб-страницы или части веб-документа для каждой активной области, по которым пользователь будет переходить при нажатии курсором мыши на активную область изображения.

Пример создания карты-изображения

```
<img src="https://html5book.ru/wp-content/uploads/2014/12/flowers_foto.jpg" alt="flowers_foto" width="680" height="383" usemap="#flowers">

<map name="flowers">

<area shape="circle" coords="70,164,50" href="https://ru.wikipedia.org/wiki/Гербера" alt="gerbera" target="_blank">

<area shape="poly" coords="191,13,240,98,143,98,191,13" href="https://ru.wikipedia.org/wiki/%C3%E8%E0%F6%E8%ED%F2" alt="hyacinth" target="_blank">

<area shape="circle" coords="318,93,50" href="https://ru.wikipedia.org/wiki/Ромашка" alt="camomiles" target="_blank">

<area shape="circle" coords="425,129,45" href="https://ru.wikipedia.org/wiki/Нарцисс_(растение)" alt="narcissus" target="_blank">

<area shape="rect" coords="480,3,572,89" href="https://ru.wikipedia.org/wiki/Тюльпан" alt="tulip" target="_blank">

</map>
```

HTML-таблицы

HTML-таблицы упорядочивают и выводят на экран данные с помощью строк или столбцов. Таблицы состоят из ячеек, образующихся при пересечении строк и столбцов.

HTML-таблицы

Создание таблицы

- <table>

```
<table>
    <tr><th>текст заголовка</th><th>текст заголовка</th></tr> <!--ряд с ячейками заголовков-->
    <tr><td>данные</td><td>данные</td></tr> <!--ряд с ячейками тела таблицы-->
</table>
```

- строки (ряды) таблицы: <tr>
- ячейка тела таблицы: <td>
- подпись к таблице: <caption> (используется после тега<table>)

Группирование строк и столбцов таблицы

- <colgroup> - создает структурную группу столбцов
- <col> - формирует группы столбцов, которые делят таблицу на разделы, не относящиеся к общей структуре

```
<table>
  <colgroup>
    <col span="2" style="background:Khaki"><
    <col style="background-color:LightCyan">
  </colgroup>
  <tr>
    <th>№ п/п</th>
    <th>Наименование</th>
    <th>Цена, руб.</th>
  </tr>
  <tr>
    <td>1</td>
    <td>Карандаш цветной</td>
    <td>20,00</td>
  </tr>
  <tr>
    <td>2</td>
    <td>Линейка 20 см</td>
    <td>30,00</td>
  </tr>
</table>
```

№ п/п	Наименование	Цена, руб.
1	Карандаш цветной	20,00
2	Линейка 20см	30,00

HTML-списки

HTML-списки используются для группировки связанных между собой фрагментов информации. Существует три вида списков:

- **маркированный список** — — каждый элемент списка <li> отмечается маркером,
- **нумерованный список** — — каждый элемент списка <li> отмечается цифрой,
- **список определений** — <dl> — состоит из пар термин <dt> — <dd> определение.

Спецсимволы HTML

Спецсимволы HTML, или символы-мнемоники, представляют собой конструкцию SGML (стандартный обобщённый язык разметки)

Вид	HTML-код	CSS-код	Описание
🂆	🖂	\1F582	Обратная сторона конверта
☃	&#9731;	\2603	Снеговик

HTML-генераторы

HTML-генераторы и полезные сервисы облегчат работу с HTML-кодом, протестируют отображение вашего сайта в разных браузерах, а также дадут информацию о поддержке браузерами тех или иных тегов и технологий.

- Mobirise
- FREE Banner Maker
- HTML minifier
- Mobile HTML5
- Initializr
- Browsershots
- HTML5Test
- HTML5 Please
- FormLinter

Семантические элементы HTML5

Семантические элементы HTML5 доступно описывают свой смысл или назначение как для браузеров, так и для веб-разработчиков.

<header>

CSS

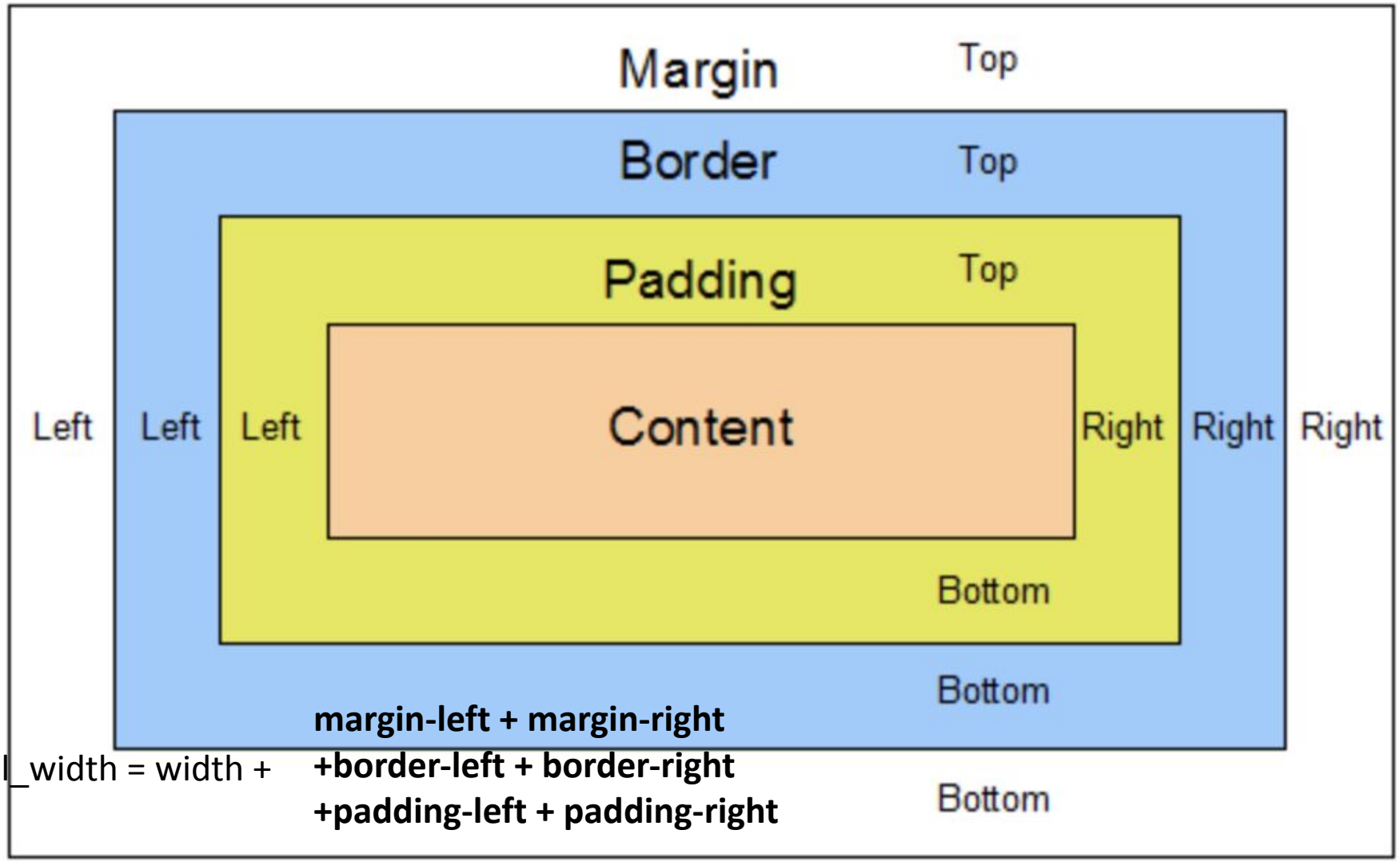
CSS блочная модель

Модуль CSS Box Model описывает свойства padding и margin, которые создают поля внутри и отступы снаружи CSS блока. Размеры блока также могут быть увеличены за счет рамки.

CSS блочная модель

Каждый блок имеет область содержимого, в которой находится текст, дочерние элементы, изображение и т.п., и необязательные окружающие ее padding, border и margin. Размер каждой области определяется соответствующими свойствами и может быть нулевым, или, в случае margin, отрицательным.

Расчёт размеров элементов



real_width = width + **margin-left + margin-right**
+border-left + border-right
+padding-left + padding-right

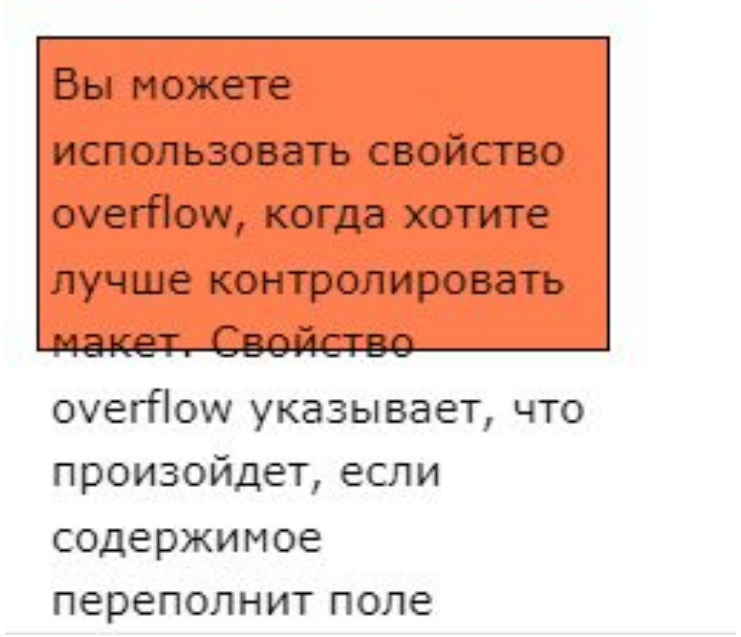
CSS блочная модель

Отступы элемента

Отступы окружают край рамки элемента, обеспечивая расстояние между соседними блоками. Свойства отступов определяют их толщину. Применяются ко всем элементам, кроме внутренних элементов таблицы. Сокращенное свойство margin задает отступы для всех четырех сторон, а его подсвойства задают отступ только для соответствующей стороны.

Расчёт размеров элементов

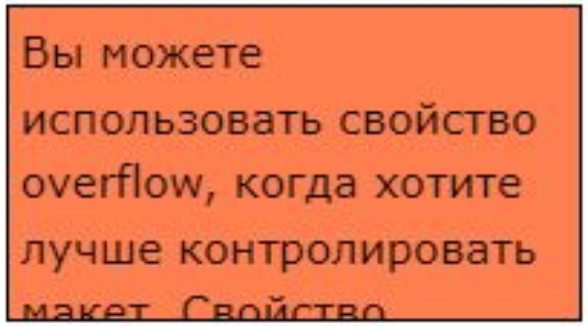
visible - По умолчанию. Переполнение не обрезается. Содержимое отображается за пределами поля элемента




```
div {
width: 200px;
height: 65px;
background-color: coral;
overflow: visible;
}
```

Расчёт размеров элементов

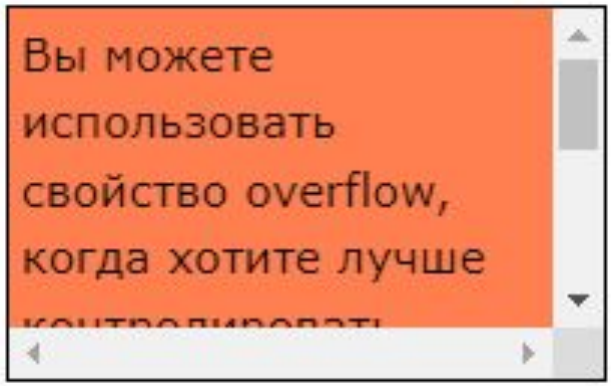
hidden - Переполнение обрезается, а остальное содержимое будет невидимым




```
div {
overflow: hidden;
}
```

Расчёт размеров элементов

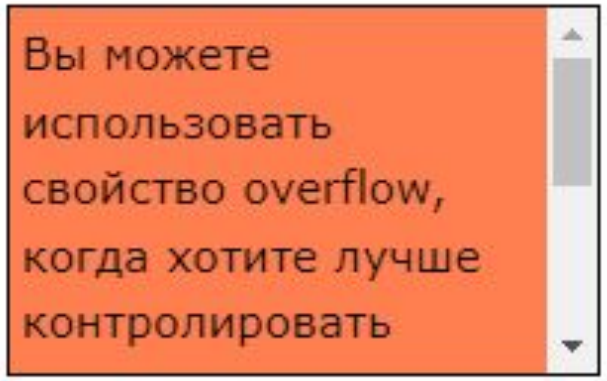
scroll - Переполнение обрезается, и добавляется полоса прокрутки для просмотра остального содержимого




```
div {
overflow: scroll;
}
```

Расчёт размеров элементов

auto - Аналогично scroll, но добавляет полосы прокрутки только при необходимости



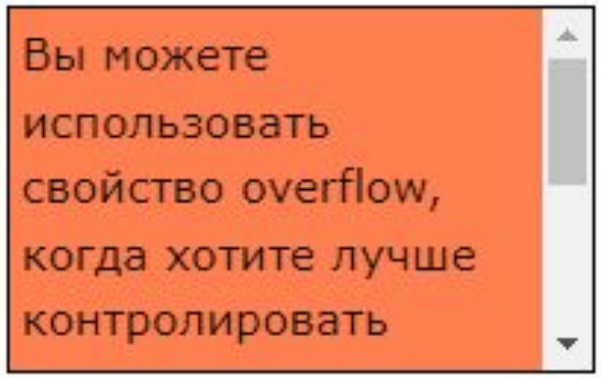

```
div {
overflow: auto;
}
```

Расчёт размеров элементов

overflow-x и overflow-y

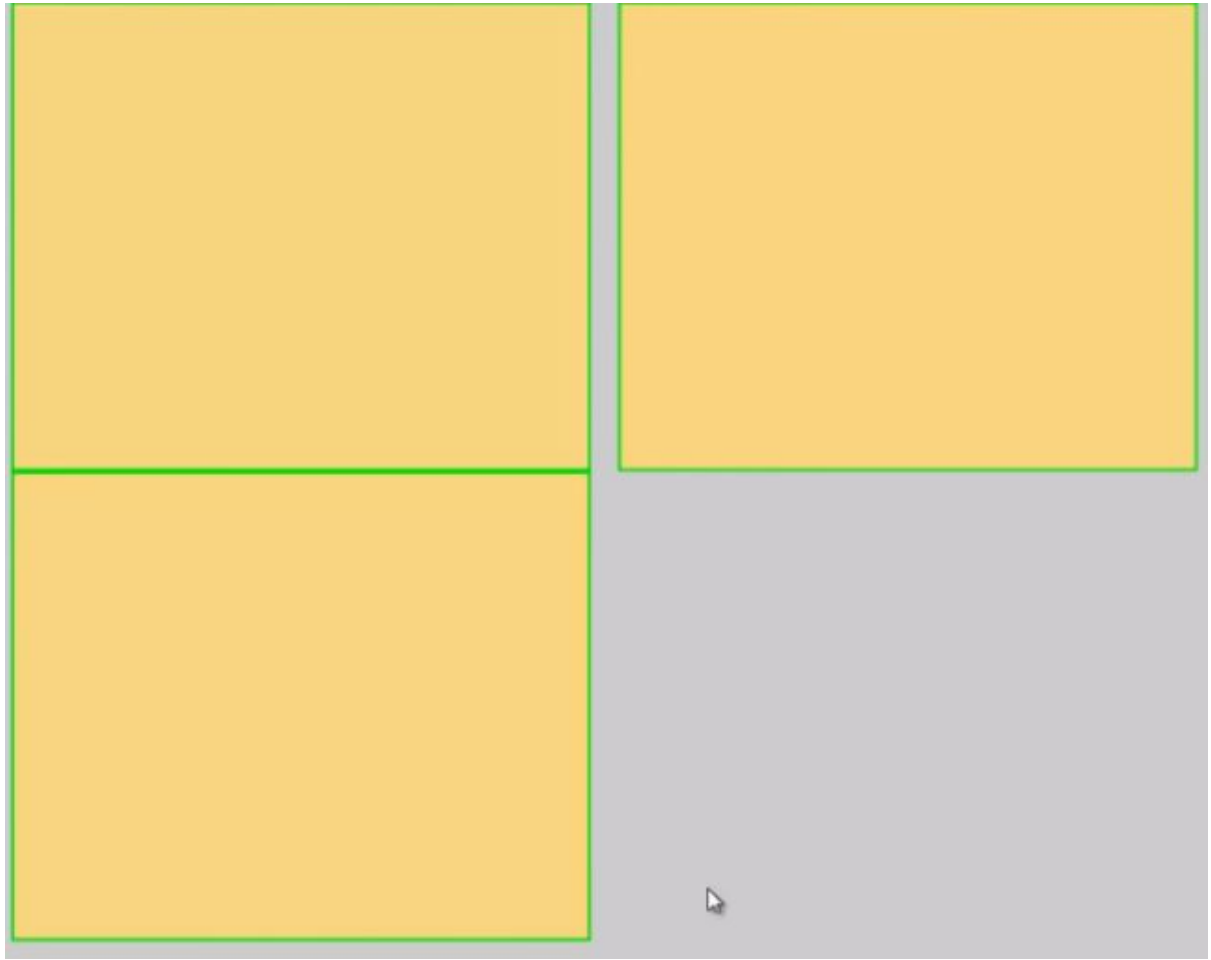
В overflow-x и overflow-y свойства указывает, следует ли изменять переполнение содержимого только по горизонтали или вертикали (или и то, и другое):

overflow-x указывает, что делать с левым / правым краями содержимого. overflow-y указывает, что делать с верхним / нижним краями содержимого.




```
div {
overflow-x: hidden;
overflow-y: scroll;
}
```

Потоки обтекания



html, body
- {margin: 0;background: #ccc;}

.sizes
- {background: #f5d781;width: 400px;height: 300px;padding: 20px 10px;margin: 0 20px;border: 2px solid #0f0;}

.sizes: first-child
- {float: left;}

.sizes:nth-child(n+ 1)
- {overflow: hidden;}

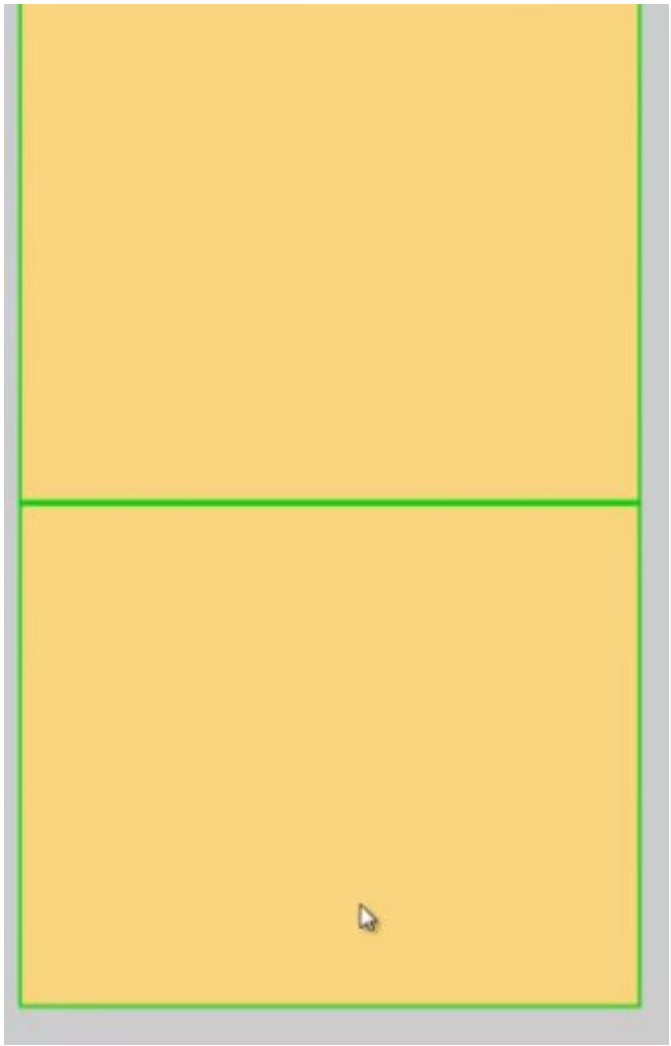
Потоки обтекания

html, body
- {margin: 0;background: #ccc;}

.sizes
- {background: #f5d781;width: 400px;height: 300px;padding: 20px 10px;margin: 0 20px;border: 2px solid #0f0;}

.sizes: first-child
- {float: left;}

.sizes: nth-child(n+ 1)
- {}



Правило вертикальных margin

Вертикальные margin всегда схлопываются!!!

- Если margin-top 20px, а margin-bottom 30px они нессумируются!

Правило вертикальных margin

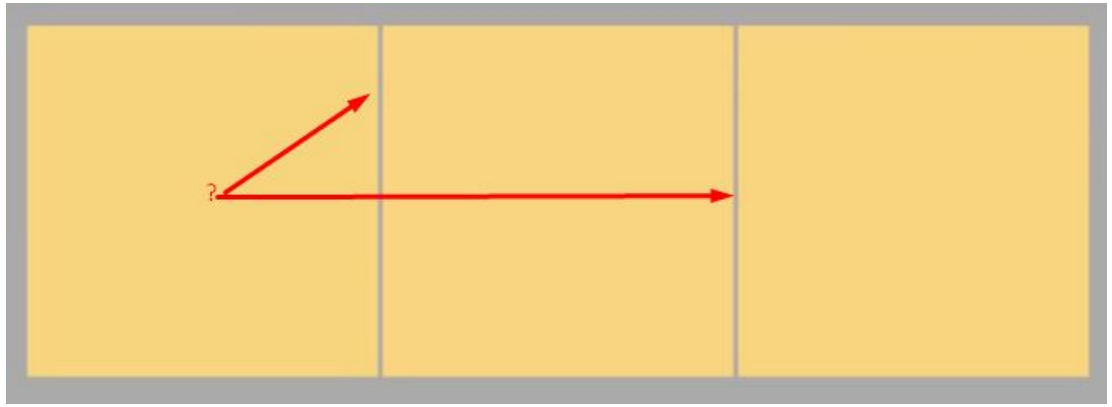
```
<Docktype>
<html>
<head>
<meta charset="utf-8">
<title>Вебинар</title>
<link href="styles.css" rel="stylesheet">
</head>
<body>
<div class="parent">
<div class="sizes"></div>
<div class="sizes"></div>
<div class="sizes"></div></div>
</body>
</html>
```

- html, body{margin: 0;background: #ccc;}
- Sizes{background: #f5d781;width: 400px;height: 30px;margin: 20px 0 30px;}

Потоки обтекания. Особенность переноса (display)

```
<doctype
<html>
<html>
<head>
<meta charset="utf-8">
<title>Вебинар</title>
<link href="styles.css" rel="stylesheet">
</head>
<body>
<div class="parent">
<div class="sizes"></div>
<div class="sizes"></div>
<div class="sizes"></div>
</div>
</body>
</html>
```

- Html, body{margin: 0;background: #ccc;}
- .parent{width: 1024px;margin: 50px auto 0;padding: 40px;background: #aaa;}
- .sizes{background: #f5d781;width: 300px;height: 300px;display: inline-block;}

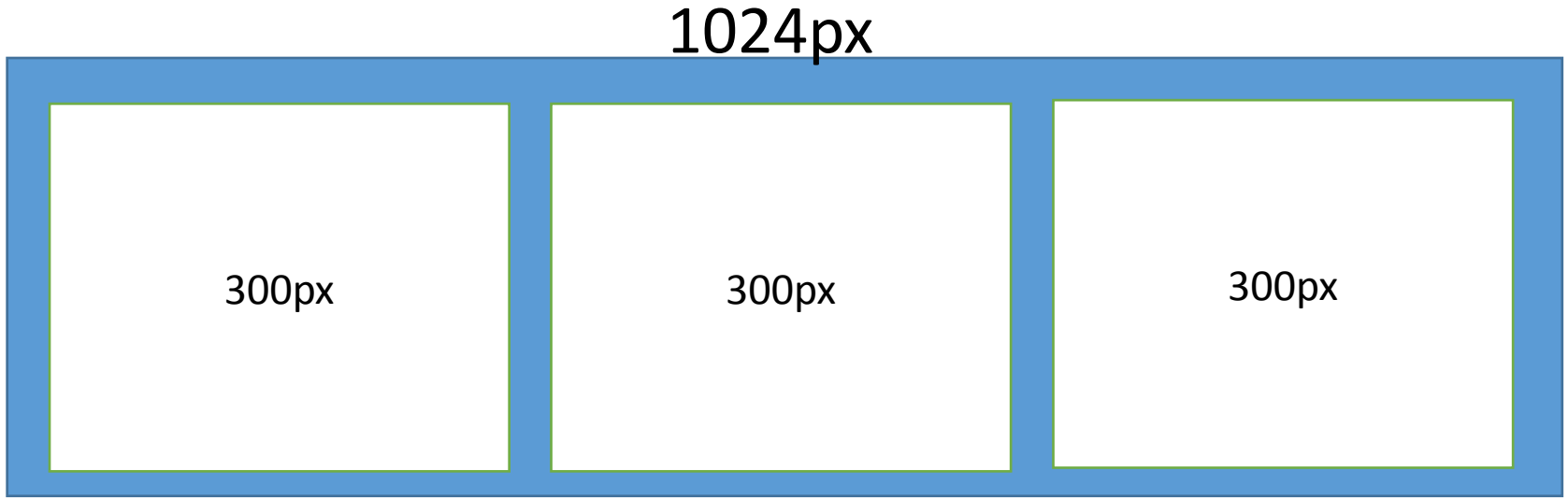


Потоки обтекания. Особенность переноса (display)

```
<doctype
<html>
<html>
<head>
<meta charset="utf-8">
<title>Вебинар</title>
<link href="styles.css" rel="stylesheet">
</head>
<body>
<div class="parent">
<div class="sizes"></div><
</div>
</body>
</html>
```

- Html, body{margin: 0;background: #ccc;}
- .parent{width: 1024px;margin: 50px auto 0;padding: 40px;background: #aaa;}
- .sizes{background: #f5d781;width: 300px;height: 300px;display: inline-block;}

Свойство box-sizing. Задача

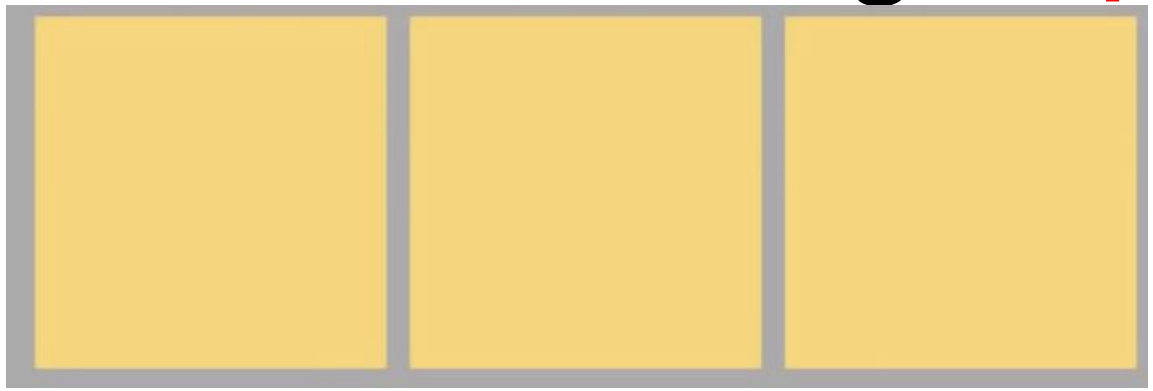


Свойство box-sizing. Задача

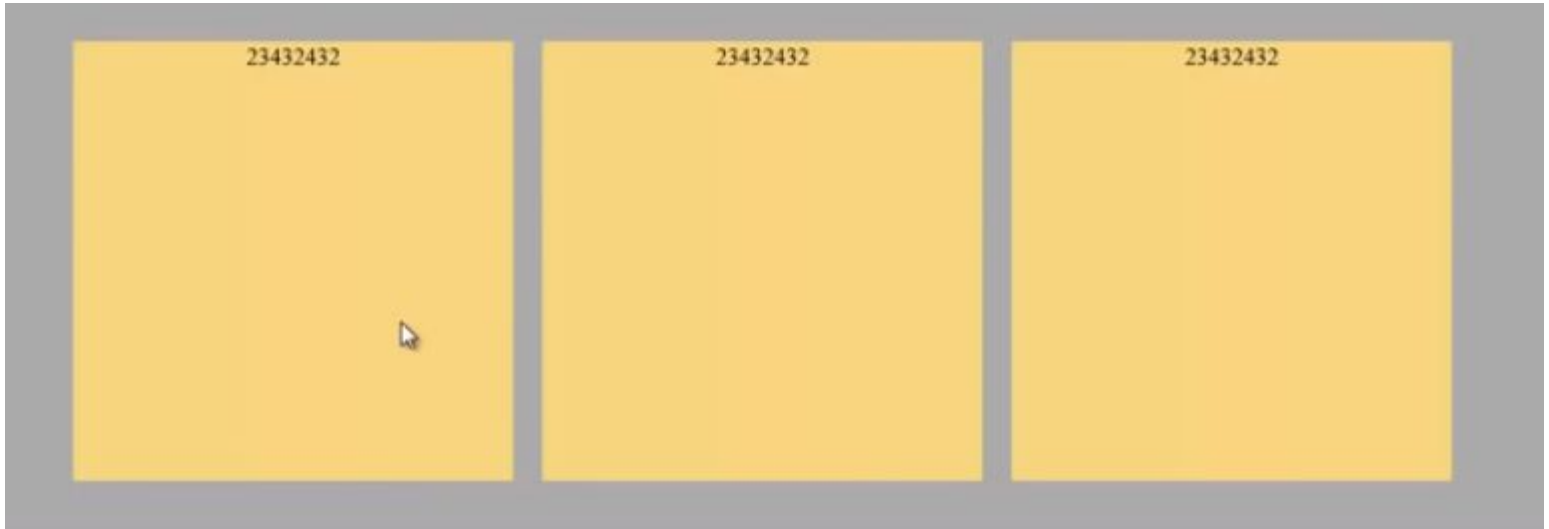
```
<doctype
<html>
<html>
<head>
<meta charset="utf-8">
<title>Вебинар</title>
<link href="styles.css" rel="stylesheet">
</head>
<body>
<div class="parent">
<div class="sizes"></div><div class="sizes"></div><div class="sizes"></div>
</div>
</body>
</html>
```

- Html, body{margin: 0;background: #ccc;}
- .parent{width: 1024px;margin: 50px auto 0;padding: 40px;background: #aaa;}
- .sizes{background: #f5d781;width: 300px;height: 300px;display: inline-block;}

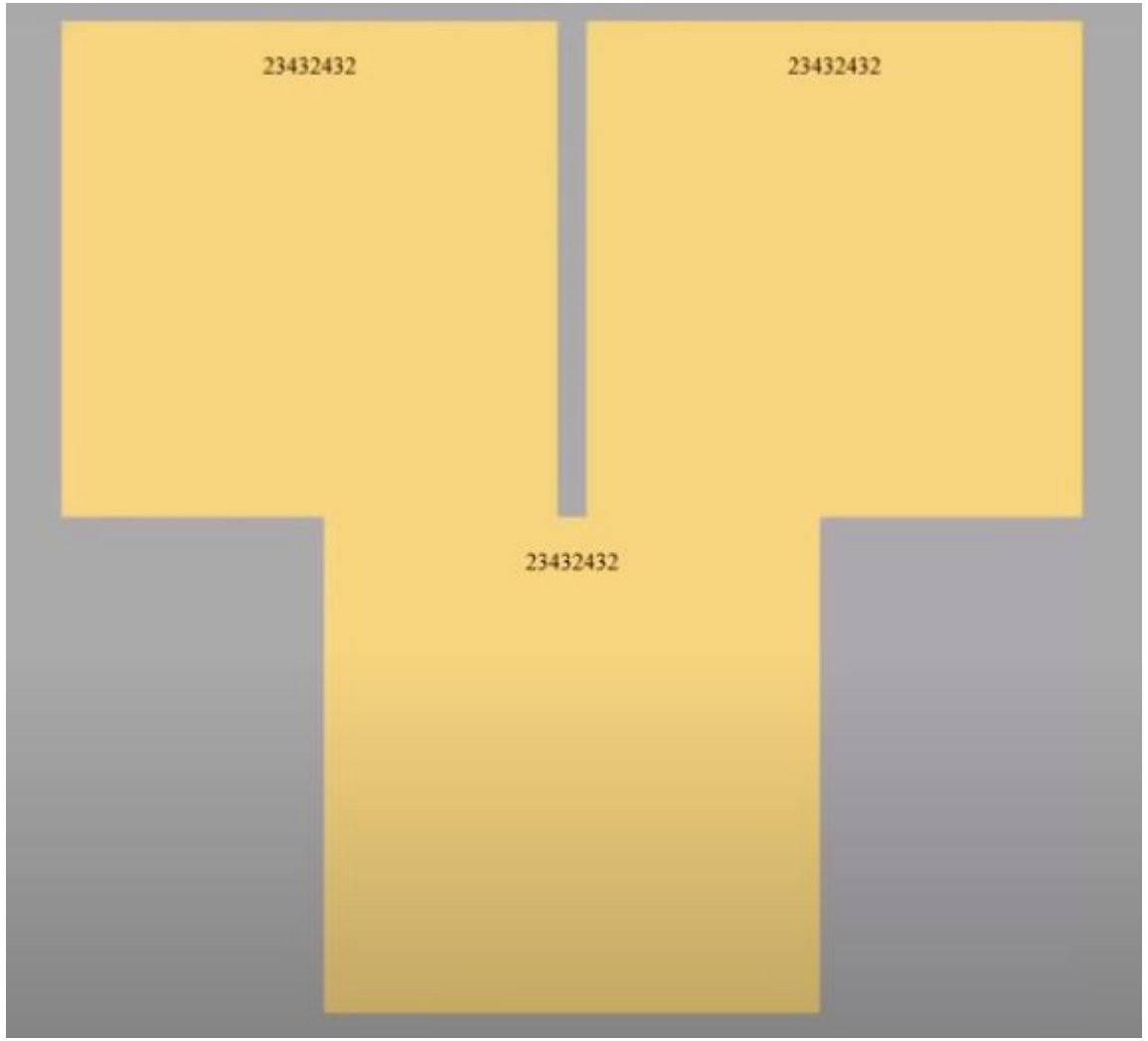
Свойство box-sizing. Задача



После добавления контента текст прилип к верхнему краю. Необходимо прописать padding



Свойство box-sizing. Задача



Свойство box-sizing. Задача

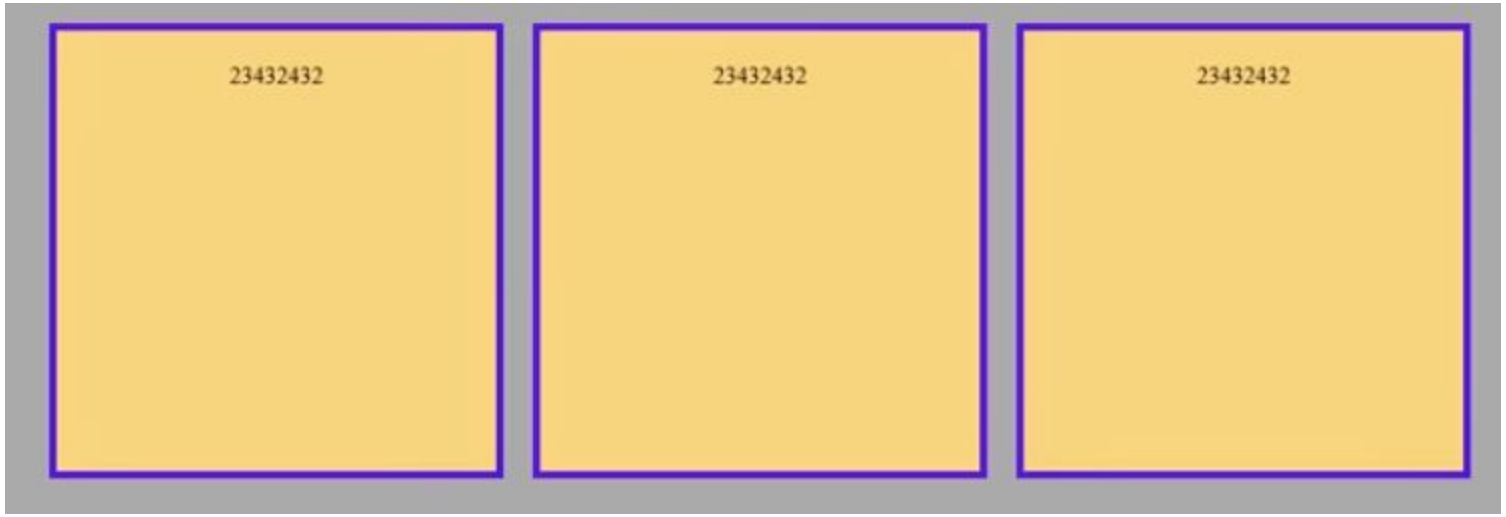
```
Html, body{margin: 0;background: #ccc;}
.parent{width: 1024px;margin: 50px auto 0;padding: 40px;background:
   #aaa;}
.sizes{background: #f5d781;width: 260px;height:
   260px;padding:20px;display: inline-block;}
```

Свойство box-sizing. Задача

Если хотим добавить рамку к каждому из элементов:

```
Html, body{margin: 0;background: #ccc;}
.parent{width: 1024px;margin: 50px auto 0;padding: 40px;background:
   #aaa;}
.sizes{background: #f5d781;width: 250px;height: 250px;padding:20px;
   border:5px solid #fg5; display: inline-block;}
```

Свойство box-sizing. Задача



Свойство box-sizing. Задача

.sizes{background: #f5d781;width: 300px;height: 300px;margin: 0 10px;padding: 20px;border: 5px solid #50d;display: inline-block;

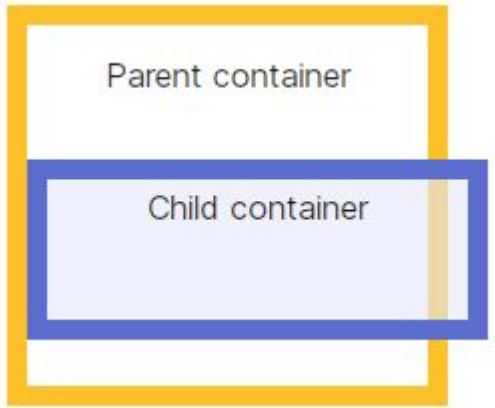
box-sizing: content-box; margin + border + padding + width *

box-sizing: border-box; margin + width

box-sizing: padding-box; margin + padding + width

Свойство box-sizing. Задача

```
box-sizing: content-box;
width: 100%;
border: solid #5B6DCD 10px;
padding: 5px;
```




```
box-sizing: border-box;
width: 100%;
border: solid #5B6DCD 10px;
padding: 5px;
```

Вертикальные margin (Правило2)

Вертикальный margin

- Для display: inline – не работает
- Для display: inline-block – работает хорошо
- Для display: block – выходит за пределы родительского элемента

Если задать border для внешнего элемента

margin будет работать для внутреннего элемента.

Вертикальные margin (Правило2)

Margin – схлопывается без border и *display: inline-block;*

```
.wrapper{background: #f5d781; border: 1px solid #fff;}
. Inner{margin: 10px 0;
 display: block;
display: inline-block;
display: inline; }
```

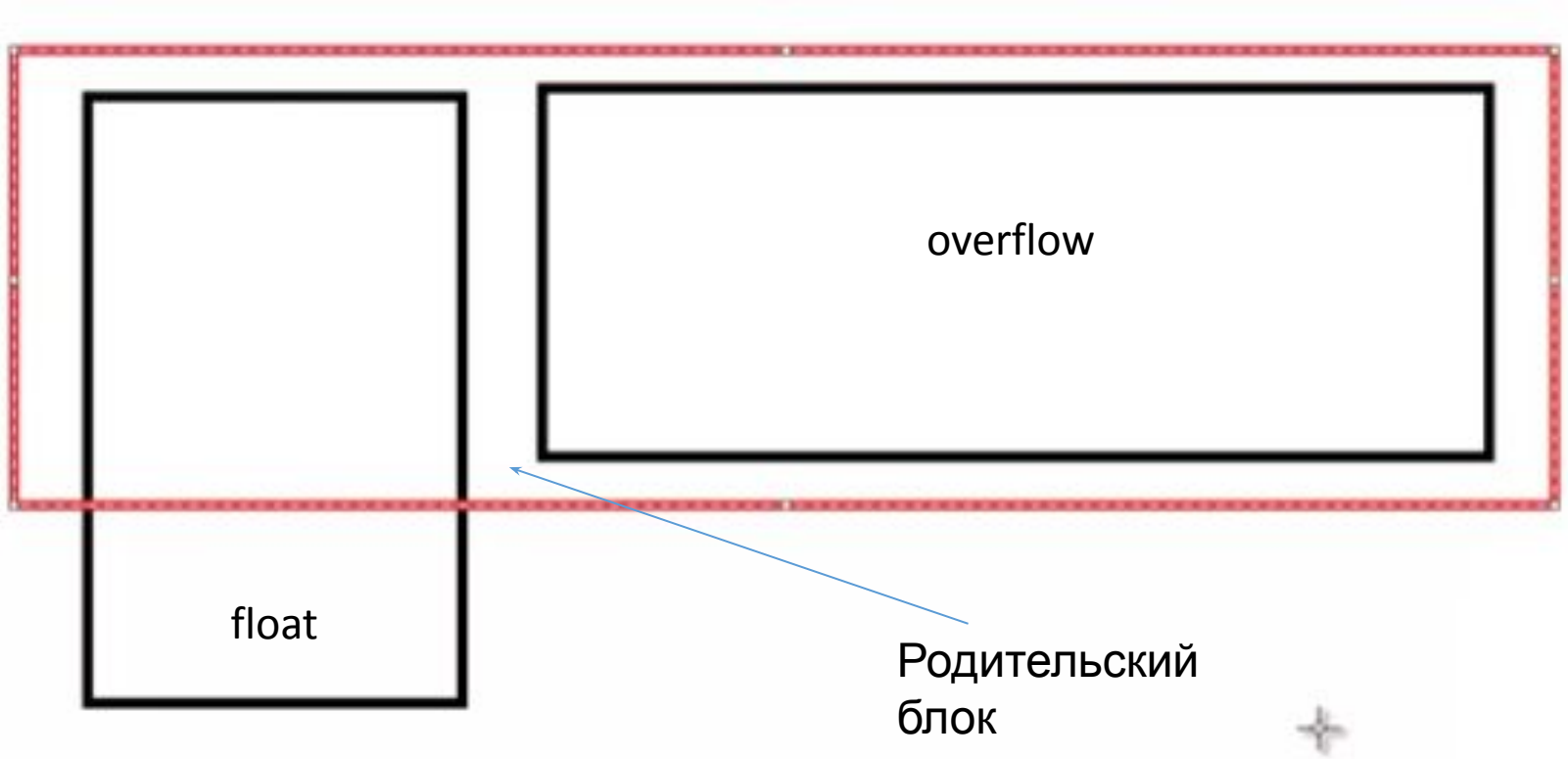
Высота блока при обтекании

```
Html, body{margin: 0;background: #cce;}

.sizes{background: #f5d781;width: 300px;height: 200px;padding: 20px 10px;margin: 0 20px;border: 2px solid #0f0;float: left;}

.parent{?}
```

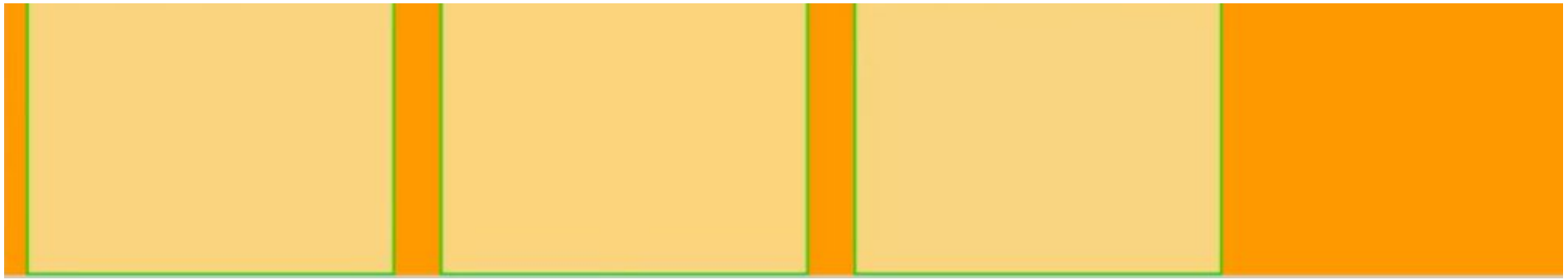
```
<div class="parent"><div class="sizes"></div><div class="sizes"></div><div class="sizes"></div></div>
```

Высота блока при обтекании

Размер родительского элемента рассчитывается только по «нормальным» дочерним элементам.



Высота блока при обтекании

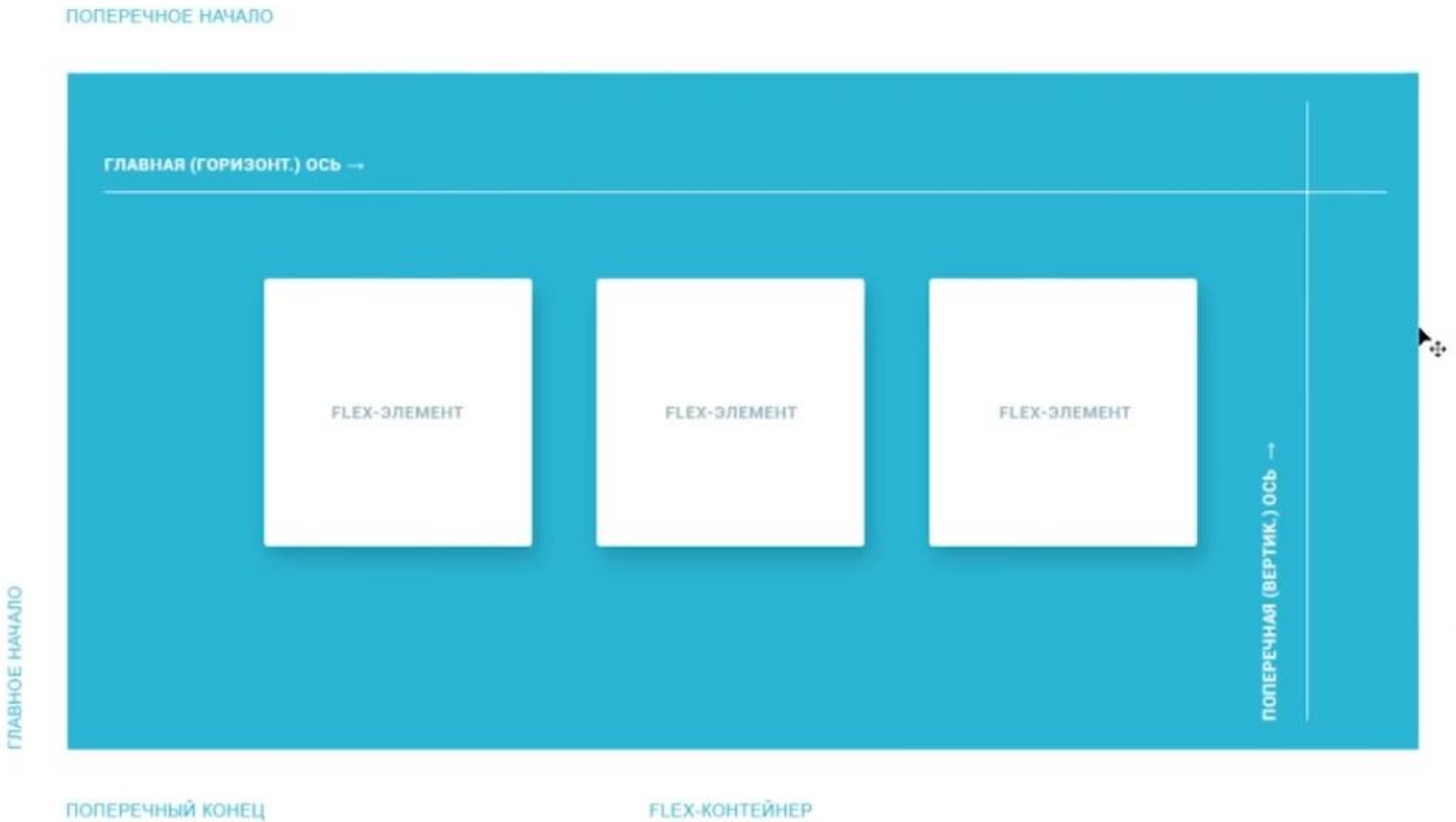


Высота блока при обтекании

Решение:

- Overflow: hidden – в случае если все должно быть выровнено, без выбивающихся участков
- Через псевдо элемент after
  - .parent:after{content: "" ;display: block;clear: both;}
- Через clearfix. Чтобы работало необходимо прописать двойной класс в html ( )
  - clearfix:after{content: "";display: block;clear: both;}

Flexbox


ПОПЕРЕЧНОЕ НАЧАЛО
ГЛАВНАЯ (ГОРИЗОНТ.) ОСЬ →
FLEX-ЭЛЕМЕНТ
FLEX-ЭЛЕМЕНТ
FLEX-ЭЛЕМЕНТ
ПОПЕРЕЧНАЯ (ВЕРТИК.) ОСЬ →
ГЛАВНОЕ НАЧАЛО
ГЛАВНЫЙ КОНЕЦ
ПОПЕРЕЧНЫЙ КОНЕЦ
FLEX-КОНТЕЙНЕР

Flexbox

Все свойства Flexbox разделены на 2 группы: Flexbox-контйнер и Flexbox-элемент

Flexbox. Задача 3

Расположить 3 блока с 4 равными расстояниями между ними.

Flexbox. Задача 3

html, body{margin: 0;background: #ccc;}

.parent {

- width:1280px;
- margin: 50px auto 0;
- padding: 40px 0;
- background: #aaa;
- text-align: center;}

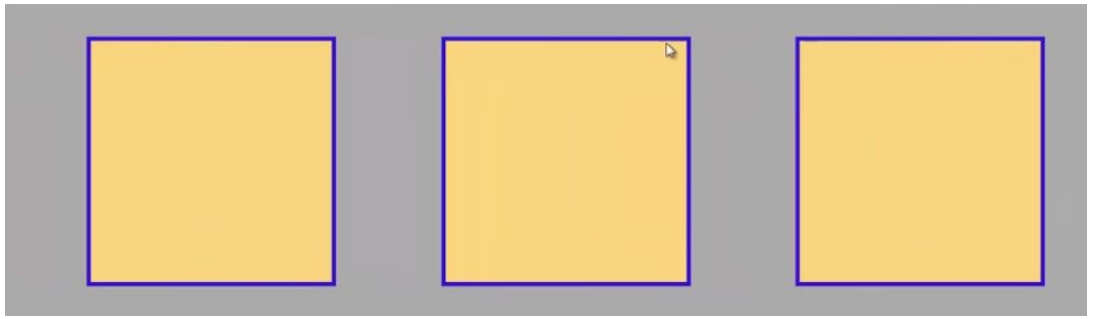
.sizes{

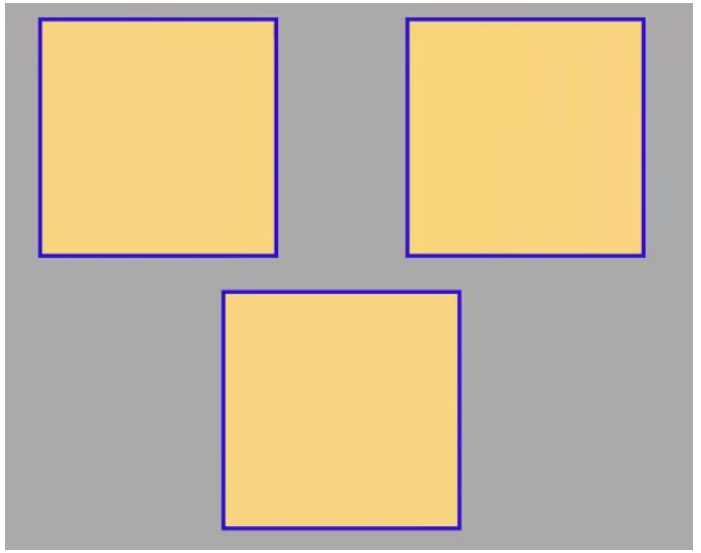
- background: #f5d781;
- width: 300px;
- height: 300px;
- border: 5px solid #30c;
- padding: 20px;
- margin: 0 10px;
- display: inline-block;
- box-sizing: border-box;}

- html, body{margin: 0;background: #ccc;}
- .parent {
  - width:1280px;
  - margin: 50px auto 0;
  - padding: 40px;
  - background: #aaa;
  - **display:flex;**
  - **justify-content: space-around;**}
- .sizes{
  - background: #f5d781;
  - width: 300px;
  - height: 300px;
  - border: 5px solid #30c;
  - padding: 20px;
  - margin: 0 10px;
  - box-sizing: border-box;}

Flexbox. Позиционирование

- html, body{margin: 0;background: #ccc;}
- .parent {
  - max-width:1280px;
  - margin: 50px auto 0;
  - padding: 20px 40px;
  - background: #aaa;
  - **display:flex;**
  - **justify-content: space-around;**
  - **Flex-wrap:wrap;**}
- .sizes{
  - background: #f5d781;
  - width: 300px;
  - height: 300px;
  - border: 5px solid #30c;
  - padding: 20px;
  - margin: 20 10px;
  - box-sizing: border-box;}





Методология BEM (Блок Элемент Модификатор)

Методология BEM

Блок – независимый компонент страницы, который может быть использован множество раз и на разных страницах.

Когда задаем блоку класс необходимо ответить на вопрос «Что это?», а не вопросы «Какой?» и «Как выглядит?».

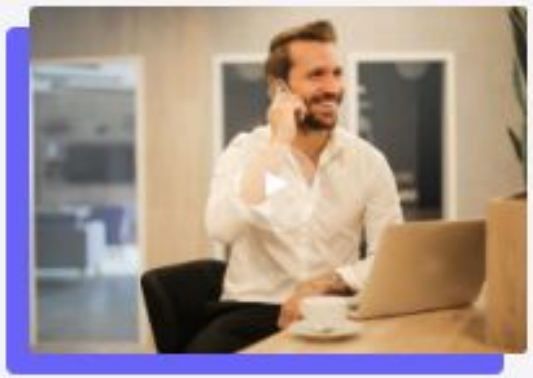
Стилю блока не рекомендуется задавать внешнюю геометрию, например отступы, что обеспечивает независимость блока при повторном использовании.

Блоки можно вкладывать друг в друга.

Допустима любая вложенность.

Категории

Sed do eiusmod tempor incididunt ut labore et dolore magna aliqua



ОНЛАЙН-КЛАССЫ И УЧЕБА

Онлайн-дистанционные занятия для всех курсов

Многие настольные издательские пакеты и редакторы веб-страниц теперь используют Lorem Ipsum в качестве модели по умолчанию

Последние курсы

Sed do eiusmod tempor incididunt ut labore et dolore magna aliqua

Методология BEM

Элемент – неотрывная составная часть блока, которая не может использоваться в отрыве от блока

Когда задаем элементу класс необходимо ответить на вопрос «Что это?», а не вопросы «Какой?» и «Как выглядит?». Синтаксис записи элемента: **имяклассаблока__имяэлемента**

Элементы можно вкладывать друг в друга.

Допустима любая вложенность.

Элемент всегда часть блока, а не другого элемента, поэтому неверный вид иерархии имеет вид: **имяклассаблока__имяэлемента __имяэлемента**

Элемент не обязательные компонент блока

```
<!--·Верно·-->
<!--·Блок·`row`·-->
<div·class="row">
——<!--·Элемент·`column`·-->
——<div·class="row__column·">
————<!--·Элемент·`item`·-->
————<div·class="row__item"></div>
——</div>
</div>
```

```
<!--·Неверно·-->
<!--·Блок·`row`·-->
<div·class="row">
——<!--·Элемент·`column`·-->
——<div·class="row__column·">
————<!--·Элемент·`item`·-->
————<div·class="row__column__item"></div>
——</div>
</div>
```

```
<!--·Верный·пример·-->
<!--·Блок·`about`·-->
<div·class="·about">
——<!--·Элемент·`title`·-->
——<div·class="about__title"></div>
——<!--·Элемент·`subtitle`·-->
——<div·class="·about__subtitle"></div>
</div>
```

```
<!--·Неверный·пример·-->
<!--·Блок·`about`·-->
<div·class="·about">
——<!--·Элемент·`title`·-->
——<div·class="about__title"></div>
</div>
<!--·Элемент·`subtitle`·-->
<div·class="about__subtitle"></div>
```

Категории

Sed do eiusmod tempor incididunt ut labore et dolore magna aliqua

ОНЛАЙН-КЛАССЫ И УЧЕБА

Онлайн-дистанционные занятия для всех курсов

Многие настольные издательские пакеты и редакторы веб-страниц теперь используют Lorem Ipsum в качестве модели по умолчанию

Последние курсы

Sed do eiusmod tempor incididunt ut labore et dolore magna aliqua

Методология BEM

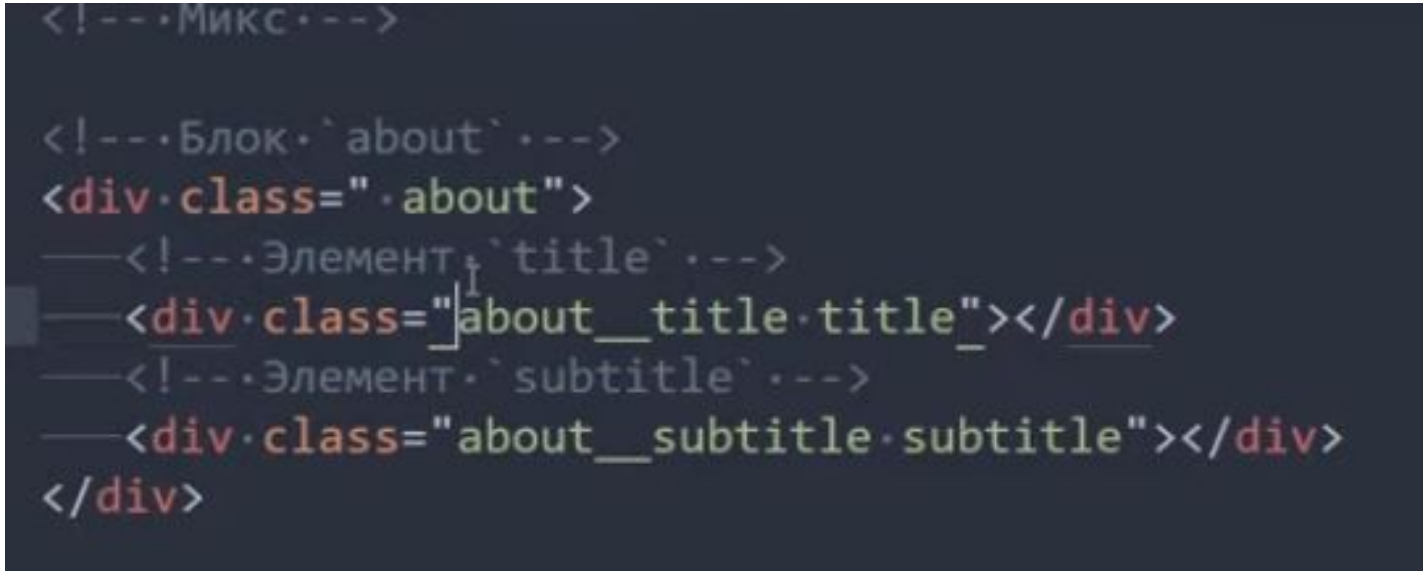
Модификатор – применяется для дополнения или уточнения внешнего вида или состояния блока или элемента

Когда задаем модификатору класс необходимо ответить на вопросы «Какой?» и «Как выглядит?». Синтаксис записи элемента:
имяклассаблока__имяэлемента_имямодификатора,
имяклассаблока_имямодификатора

Методология BEM

Прием МИКС – позволяет использовать и блоки и элементы в одном объекте

МИКС – позволяет совмещать поведение нескольких объектов без дублирования кода, создавать семантически новые компоненты интерфейса на основе существующих.



Блочные и строчные элементы

Блочные элементы — элементы высшего уровня, которые форматируются визуально как блоки, располагаясь на странице в окне браузера вертикально. Значения свойства **display,** такие как block, list-item и table делают элементы блочными. Блочные элементы генерируют основной блок, который содержит только блок элемента. Элементы со значением **display: list-item** генерируют дополнительные блоки для маркеров, которые позиционируются относительно основного блока.

Блочные элементы

<address>, <article>, <aside>,

<blockquote>,

<dd>, , <dl>, <dt>, <details>,

<fieldset>, <figcaption>, <figure>, <footer>, <form>,

<h1>-<h6>, <header>, <hr>,

, <legend>,

<nav>, <noscript>,

, <output>, <optgroup>, <option>,

<p>, <pre>,

<section>, <summary>,

<table>,

Блочные элементы

Элемент **<p>** относится к блочным элементам, но он не должен содержать внутри себя другой элемент **<p>**, а также любой другой блочный элемент.

Строчные элементы и строчные контейнеры

Встроенные (строчные) элементы генерируют внутристрочные контейнеры. Они не формируют новые блоки контента. Значения свойства **display**, такие как **inline** и inline-table делают элементы строчными.

Строчные элементы и строчные контейнеры

<a>, <area>,

<b>, <bdo>, <bdi>,

<cite>, <code>,

<dfn>, <del>,

,

<i>, <iframe>, <img>, <ins>,

<kbd>,

<label>,

<map>, <mark>,

<s>, <samp>, <small>, , , <sub>, <sup>,

<time>,

<q>,

<ruby>,

<u>,

<var>

**span {padding: 10px;
background: #c4c4c4;
border: 2px dashed grey}**

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

**span {padding: 10px;
margin: 30px;
background: #c4c4c4;
border: 2px dashed grey;}**

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

**span {display: inline-block;
padding: 10px;
background: #c4c4c4;
border: 2px dashed grey;}**

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Строчно-блочные элементы

Существует еще одна группа элементов, которые браузер обрабатывает как строчно-блочные {**display: inline-block;**}. Такие элементы являются встроенным, но для них можно задавать поля, отступы, ширину и высоту.

Строчно-блочные элементы

<audio>,

<button>,

<canvas>,

<embed>,

<input>,

<keygen>,

<meter>,

<object>,

<progress>,

<select>,

<textarea>,

<video>.

Ширина содержимого: свойство width

Свойство **width** определяет ширину содержимого блока.

Это свойство не применяется к незамещаемым строчным элементам **display: inline**;. Ширина содержимого встроенных блоков определяется шириной отображаемого содержимого внутри них. Встроенные блоки сливаются в линейные блоки. Ширина линейных блоков определяется шириной содержащего блока, но может быть уменьшена из-з
свойства **float**.

```
width: 100px;
width: 10em;
width: 50%;
width: auto;
width: inherit;
```

Минимальная и максимальная высота: свойства min-height и max-height

```
min-height: 100px;
min-height: 2em;
min-height: 50%;
min-height: inherit;
```

```
max-height: 500px;
max-height: 20em;
max-height: 80%;
max-height: none;
max-height: inherit;
```

Минимальная и максимальная ширина: свойства min-width и max-width

Свойства **min-width** и **max-width** позволяют ограничивать ширину содержимого до определенного диапазона. Значения не могут быть отрицательными. Для **min-width** значение по умолчанию 0, для max-**width — none**.

```
min-width: 100px;
min-width: 10em;
min-width: 50%;
min-width: inherit;
```

```
max-width: 500px;
max-width: 20em;
max-width: 80%;
max-width: none;
max-width: inherit;
```

Высота содержимого: свойство height

Свойство **height** определяет высоту содержимого блока. Это свойство не применяется к незамещаемым строчным элементам. Значения длины **не могут быть отрицательными**.

```
height: 100px;
height: 10em;
height: 50%;
height: auto;
width: inherit;
```

Расчет высоты строки: свойства line-height и vertical-align

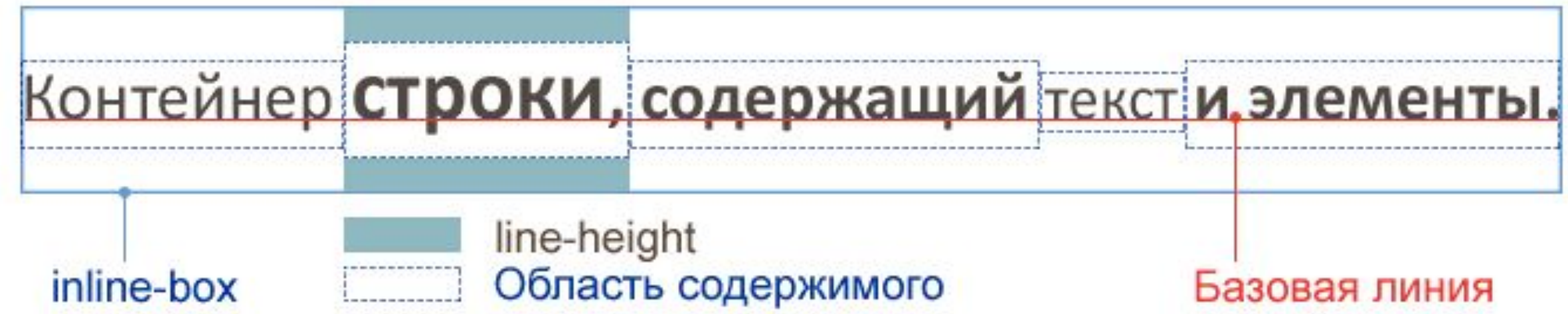
Высота линейного блока определяется следующим образом:

Высота каждого встроенного прямоугольника в линейном блоке вычисляется. Для замещаемых, **inline-block** и **inline-table** элементов это высота их области поля (margin box).

Блоки уровня строки выравниваются вертикально в соответствии со значением свойства **vertical-align**. Если они выровнены по верху или по низу, они должны быть выровнены так, чтобы минимизировать высоту линейного

```
line-height: normal;
line-height: 2em;
line-height: 1.5;
line-height: 50%;
line-height: inherit;
```

Расчет высоты строки: свойства line-height и vertical-align



Изменение блочной модели: свойство box-sizing

Свойство **box-sizing** переключает блочную модель с фиксированных размеров длины и ширины на **content-box** и **border-box**. Это влияет на интерпретацию всех свойств, определяющих размеры, включая **flex-basis**.

```
box-sizing: content-box;
box-sizing: border-box;
box-sizing: inherit;
box-sizing: initial;
```

CSS-позиционирование

CSS рассматривает макет html-документа как **дерево элементов**. Уникальный элемент, у которого нет родительского элемента, называется **корневым элементом**. Модуль CSS-позиционирование описывает, как любой из элементов может быть размещен **независимо от порядка документа** (т.е. извлечен из «потока»).

CSS-позиционирование

В CSS блок элемента может быть расположен в соответствии с тремя схемами позиционирования:

1. **Нормальный поток**

Нормальный поток включает блочный контекст форматирования (элементы с **display block, list-item или table**), строчный (встроенный) контекст форматирования (элементы с **display inline**, **inline-block** или **inline-table**), и относительное и «липкое» позиционирование элементов уровня блока и строки.

CSS-позиционирование

2. **Обтекание**

В обтекающей модели блок удаляется из нормального потока и позиционируется влево или вправо. Содержимое обтекает правую сторону элемента с **float: left** и левую сторону элемента с **float: right**.

3. **Абсолютное позиционирование**

В модели абсолютного позиционирования блок полностью удаляется из нормального потока и ему присваивается позиция относительно содержащего блока. Абсолютное позиционирование реализуется с помощью значений **position: absolute**; и **position: fixed**;.

CSS-позиционирование
Выбор схемы позиционирования: свойство position

Свойство position определяет, какой из алгоритмов позиционирования используется для вычисления положения блока

```
position: static;
position: relative;
position: absolute;
position: sticky;
position: fixed;
position: initial;
position: inherit;
```

CSS-позиционирование
Смещение блока: свойства top, right, bottom, left

Элемент считается позиционированным, если свойство position имеет значение, отличное от **static**. Позиционированные элементы генерируют позиционированные блоки и могут быть расположены в соответствии со следующими четырьмя физическими

```
top: 10px;      right: -10px;    bottom: 50px;      left: 50px;
top: 2em;       right: .5em;     bottom: -3em;      left: 10em;
top: 50%;       right: -10%;     bottom: -50%;      left: 20%;
top: auto;      right: auto;     bottom: auto;      left: auto;
top: inherit;   right: inherit;  bottom: inherit;   left: inherit;
top: initial;   right: initial;  bottom: initial;   left: initial;
```

CSS-позиционирование
Обтекание: свойство float

Обтекание позволяет блокам смещаться влево или вправо на текущей строке. «Плавающий блок» смещается влево или вправо до тех пор, пока его внешний край не коснется края содержащего блока или внешнего края другого плавающего блока.

```
float: left;
float: right;
float: none;
float: inherit;
```

БЛОК 1. Этот блок отображается в нормальном нисходящем потоке.

БЛОК 2. Для этого блока задано обтекание по левому краю.

БЛОК 3. Этот блок также отображается в нормальном нисходящем потоке. Он игнорирует плавающий блок, а текст в блоке его обтекает.

РАЗМЕТКА:

```
<div style="height:150px; border:1px dashed #8bc63e">БЛОК 1.</div>
<div style="height:200px; width:300px; border:3px dashed #e03c32; float:left">БЛОК 2.</div>
<div style="height:150px; border:1px solid #888888">БЛОК 3.</div>
```

CSS-позиционирование
Управление потоком рядом с плавающими элементами: свойство clear

Свойство clear указывает, какие стороны блока/блоков элемента не должны прилегать к плавающим блокам, находящемся выше в исходном документе.

```
clear: none;
clear: left;
clear: right;
clear: both;
clear: inherit;
```

CSS-позиционирование
Определение контекста наложения: свойство z-index

Свойство z-index позволяет изменить порядок наложения позиционированных элементов в случае, когда они накладываются друг на друга.

`auto`	Вычисляется в `0`. Если для блока задано `position: fixed;` или это корневой элемент, значение `auto` также устанавливает новый контекст наложения. Значение по умолчанию.
целое число	Определяет положение блока в текущем контексте наложения. Также устанавливает новый локальный контекст наложения. Можно использовать любое целое число, включая отрицательные числа. Отрицательные значения помещают элемент вглубь экрана.
`inherit`	Наследует значение свойства от родительского элемента.
`initial`	Устанавливает значение свойства в значение по умолчанию.

CSS-шрифты. Насыщенность шрифта: свойство font-weight

`normal`	Значение по умолчанию, устанавливает нормальную насыщенность шрифта. Эквивалентно значению насыщенности, равной 400.
`bold`	Делает шрифт текста полужирным. Эквивалентно значению насыщенности, равной 700.
`bolder`	Насыщенность шрифта будет больше, чем у предка.
`lighter`	Насыщенность шрифта будет меньше, чем у предка.
`100, 200, 300, 400, 500, 600, 700, 800, 900`	Значение 100 соответствует самому легкому варианту начертания шрифта, а 900 — самому плотному. При этом, эти числа не определяют конкретной плотности, т.е. 100, 200, 300 и 400 могут соответствовать одному и тому же варианту слабой насыщенности начертания шрифта; 500 и 600 — средней насыщенности, а 700, 800 и 900 могут выводить одинаковое очень насыщенное начертание. Распределение плотности так же зависит от количества уровней насыщенности, определенных в конкретном семействе шрифтов.
`initial`	Устанавливает значение свойства в значение по умолчанию.
`inherit`	Наследует значение свойства от родительского элемента.