

Практика №2: CSS

Web-программирование / ПГНИУ

CSS

- **Cascading Style Sheets**
- Каскадные таблицы стилей
- Формальный язык иерархических правил (каскадных таблиц), используемый для представления внешнего вида документа, написанного с использованием языка разметки

Синтаксис

- Состоит из **стилей** и **@-правил**
- Стили - множество правил

```
Правило ::= selectorlist {  
    property : value ;  
    [ property : value ; ]*  
}
```

- `selectorlist ::= selector [, selector]*`

Селекторы

Определяют множество элементов, на которые действует правило

Базовые селекторы

- Селектор по тегу: `tag`, `p`, `button`, `a`
- Универсальный: `*` - все элементы

HTML: атрибуты id, class

- **id** - универсальный HTML атрибут уникального идентификатора элемента
- **class** - универсальный HTML атрибут со списком имен классов (через пробел), к которым принадлежит элемент

```
<p class="paragraph line"></p>  
<p id="second-paragraph" class="paragraph"></p>  
<p class="last__line SomeClass__nam-e-5"></p>
```

Селекторы по `id` и `class`

- Селектор по классу: `.class`
- Селектор по идентификатору: `#id`
- Мультиклассы: `tag.class1.class2`, `#id.class1.class2`

Комбинационные селекторы

- Потомок: $x\ y$
Выбрать все y , являющиеся потомками x
- Смежный: $x\ \sim\ y$
Выбрать все y , являющиеся смежными (на одном уровне) с x
- Следующий: $x\ +\ y$
Выбрать все y , находящиеся непосредственно после x
- Непосредственно дочерний: $x\ >\ y$
Выбрать все y , являющиеся дочерними для x

Селекторы по атрибутам

- `[attr]` – имеет атрибут `attr`
- `[attr='value']` – имеет атрибут `attr` со значением `value`
- `[attr^='value']` – ... начинающийся на `value`
- `[attr$='value']` – ... заканчивающийся на `value`
- `[attr*='value']` – ... содержащий подстроку `value`
- `[attr~='value']` – ... содержащий `value` в списке
- `[attr|='value']` – ... содержащий `value` в списке с дефисом

Псевдоэлементы

- Определяют особые части элементов
- `::after`, `::before` – часть после/до
- `::backdrop` – верхний элемент по оси Z
- `::first-letter` – первая буква
- `::first-line` – первая строка
- `::placeholder` – плейсхолдер полей ввода
- `::selection` - выделение

Псевдоклассы

- Селекторы, обозначающие состояние элемента
- `:root`
- `:active`, `:focus`
- `:checked`, `:disabled`
- `:hover`
- `:not(selector)`
- `:empty`
- `:first-child`, `:last-child`
- `:nth-child(odd | even | <число> | <выражение>)`

Примеры селекторов

- `*`
 - Все элементы
- `p, div.paragraph`
 - Параграфы и элементы `div` с классом `paragraph`
- `#navbar a:active, #navbar a:focus`
 - Активные ссылки и ссылки в фокусе в эл. с `id="navbar"`
- `input[type="checkbox"]:checked + label`
 - `label`, который идёт сразу за выбранным чекбоксом

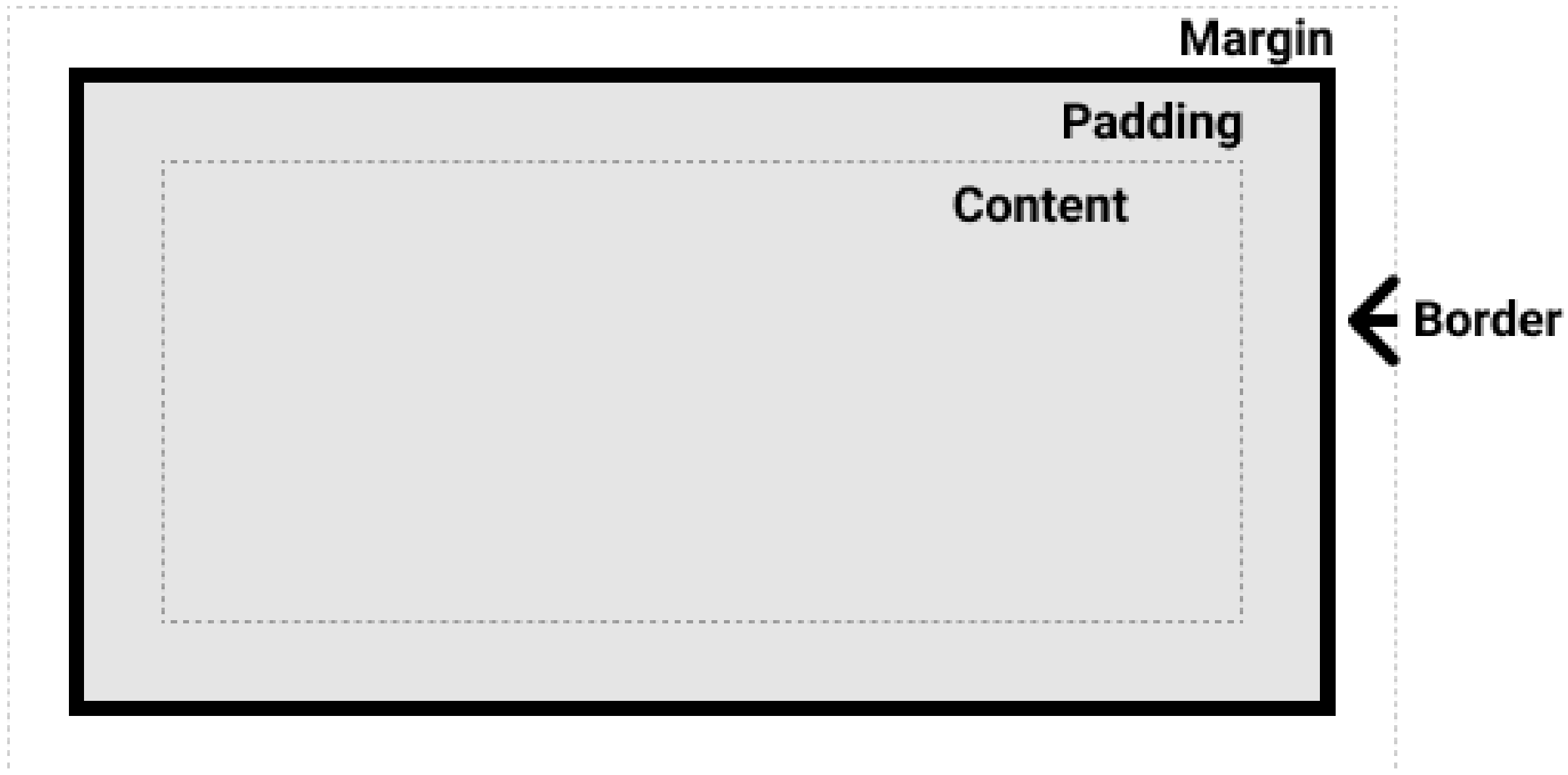
Стили

Стили определяют разные свойства элемента, которые влияют на его внешний вид и на поведение рендеринга (положение элемента и его влияние на положение других элементов).

Многие свойства наследуются и переопределяются **каскадно**.

Блочная модель

- Каждый элемент в HTML документе представлен прямоугольным блоком
- **Box = Content box + Padding box + Border box + Margin box**
- Тип блока определяет:
 - **внешний тип отображения**, т.е. расположение блока в потоке элементов
 - **внутренний тип отображения**, т.е. расположение дочерних элементов



Виды блоков

- Блочный (**block box**)
 - занимает всю доступную ширину
- Строчный (**inline box**)
 - часть текста
 - высота = высота строки текста
 - выравнивается относительно текста.
- Блочно-строчный (**inline-block**)
 - размер определяется содержимым, как у блока,
 - но обтекается окружающими элементами, как строчный

Виды блоков

- Таблица и её элементы (**table**, **table-row**, **table-cell...**)
- Список (**list-item**)
- Гибкий (**flexible box**, **flex layout**) - одномерный направленный макет с растяжимыми элементами
- Сетка (**grid layout**) - макет с двумерной сеткой

Основные группы свойств

- Блочная модель:
тип блока, контент, отступы, поля, границы, размер
- Свойства текста
- Свойства шрифта
- Свойства фона
- Свойства позиционирования
- Анимации
- Трансформации
- И многое др.

Размер

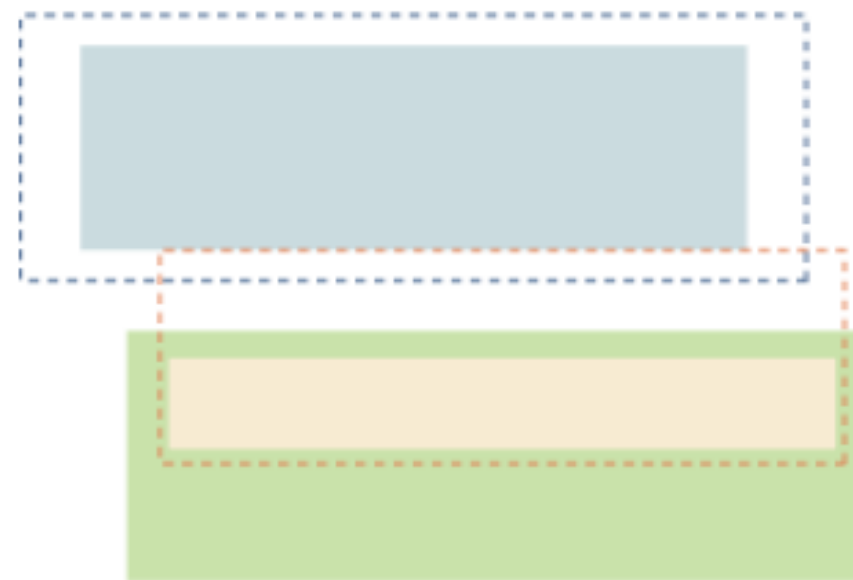
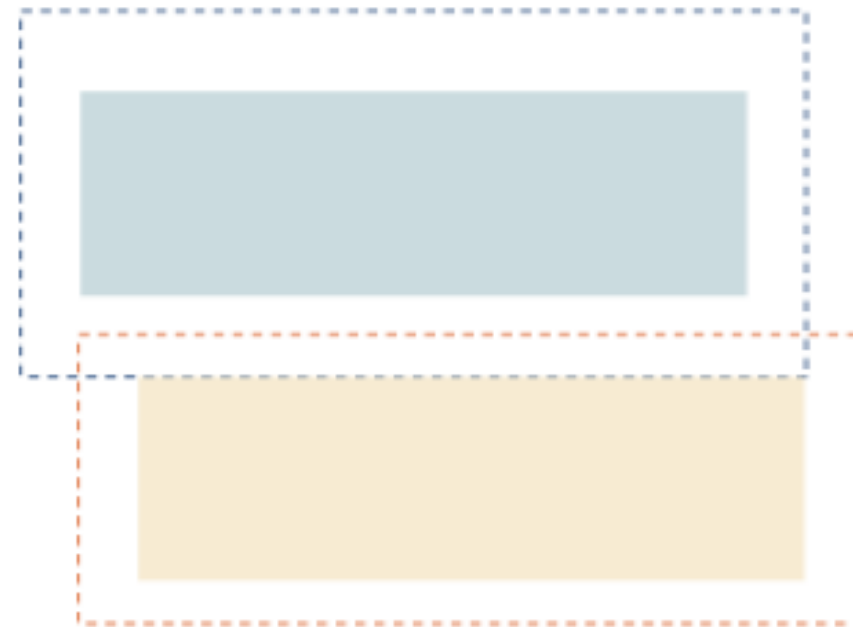
- `width`, `height` – ширина и высота
- `box-sizing` – определяет, какой части устанавливается размер
 - контенту
 - целиком с внутренним отступом и рамкой

Поля и отступы

- Отступ внутри блока
 - `padding-left: <value>; (-right, -top, -bottom)`
 - `padding: <top> <right> <bottom> <left>;`
 - `padding: <vertical> <horizontal>;`
 - `padding: <all>;`
- `margin` – аналогично отступ снаружи
- `margin` может быть отрицательным
- Особое значение - `auto`

Схлопывание отступов

- Наружные отступы могут перекрываться, кроме случаев:
 - есть внутренний отступ или граница
 - вне основного потока блоков



Граница

- Ширина: `border-width`
- Стил: `border-style` (solid, dashed, dotted)
- Цвет: `border-color`
- Скругление углов: `border-radius`
- `border: <width> <style> <color>`
- `-left, -right, -top, -bottom`

```
button {  
    border: 1px solid gray;  
    border-radius: 4px;  
}
```

Текст и шрифт

- Стили текста (выравнивание, подчёркивание, отступы, тень, переносы)
- `text-align`, `text-decoration`, `text-indent`, `text-shadow`, `word-break`, `white-space` ...
- Стили шрифта (семейство, размер, жирность, цвет и т.д.)
- `font-family`, `font-size`, `font-weight`, `color` ...
- Высота строки: `line-height`

Текст и шрифт

```
body {  
    font-family: "Open Sans", Tahoma, sans-serif;  
    font-weight: normal;  
    font-style: italic;  
    color: black;  
    line-height: 1.5;  
}  
  
a {  
    color: blue;  
    text-decoration: underline;  
}
```


Фон

- Цвет: `background-color`
- Изображение: `background-image`
- Позиция: `background-position`
- Повтор: `background-repeat`
- Область: `background-origin`
- Размер (заполнение): `background-size`
- `background: image position [/ size] repeat attachment origin clip`

Position

- `static` – обычная позиция в общем потоке рендеринга
- `relative` – позиция рендеринга сдвигается относительно обычной, при этом в общем потоке место остаётся занятым
- `absolute` – элемент выходит из основного потока рендеринга и рендерится поверх, относительно всего документа или ближайшего `relative`
- `fixed` – аналогично `absolute`, но относительно окна браузера
- `sticky` – фиксированный в определённой области (в родителе)
- За позицию отвечают свойства: `top, left, right, bottom`
- `z-index` – позиция по оси Z при накладывании друг на друга

Position

```
.overlay {  
  position: fixed;  
  top: 0;  
  left: 0;  
  right: 100%;  
  bottom: 100%;  
  z-index: 999;  
}
```

Тип блока

Свойство `display` позволяет явно изменить тип блока:

- `block`
- `inline`
- `inline-block`
- `flex`
- `grid`
- `table`
- И т.д.

Другие свойства

- `cursor` – курсор над элементом
- `opacity` – прозрачность
- `box-shadow` – тень
- `overflow` – что делать, если контент не помещается
- `transform` – различные трансформации
- `transition` – плавное изменение свойств
- `float` – плавающий блок в тексте
- `filter` - применение фильтра (размытие, яркость, контраст...)
- и многие другие...

Значения

- Относительные единицы измерения: `px`, `%`, `em`, `rem`, `vh`, `vw` (`10px`)
- Абсолютные единицы измерения: `in`, `cm`, `mm`, `pt`, `pc`
(фиксированные!)
- Числа, строки: `1`, `5.0`, `"string"`
- Для `0` не требуется единицы измерения
- Цвета: `#hex`, название (`red`, `blue`), `rgb(r, g, b)`, `rgba(r, b, g, a)`, `hsl(h, s, l)`, `transparent`
- Зарезервированные значения (`solid`, `cover`, `block`)
- `initial`, `inherit`, `none`

Функции

- Вычисление `calc(100% - 10px)`, `min(a, b)`, `max(a, b)`, `clamp(10px, 4em, 80px)`
- Значение пользовательского свойства `var(--variable)`
- Значение атрибута `attr(attr_name)`

Пользовательские свойства

- Ака **CSS-переменные**
- Любые свойства, начинающиеся на `--`
- Наследуется на все дочерние элементы
- Получить значение можно из функции `var()`

Пользовательские свойства

```
:root {  
  --base-size: 100px;  
}  
  
main {  
  --base-size: 200px;  
  --base-spacing: 10px;  
  --danger-color: red;  
}  
  
button {  
  color: var(--danger-color);  
  width: calc(var(--base-spacing) + var(--base-width));  
}
```

@-правила

- `@charset "utf-8";`
- `@media` – медиа запросы
- `@font-face` – подключение шрифта
- `@keyframes` – шаги анимации
- `@import` – импорт других таблиц стилей

@media

```
@media screen and (min-width: 1000px) {  
  main {  
    width: 1000px;  
  }  
}
```

Подключение стилей

- Подключение отдельного файла
- В элементе `style`
- Inline-стили в атрибуте `style`

```
<link href="path/to/style.css" rel="stylesheet" />
<style>/* Some css */</style>
<div style="color: red; font-size: 10px;"></div>
```

Специфичность (приоритет)

- Специфичность - вес правила
- Применяется правило наибольшего веса
- Значения:
 - 0 - селекторы типа элемента или псевдоклассы
 - 1 - селекторы по классу, атрибуту, псевдоклассу
 - 2 - селекторы по идентификатору
- Правило в `inline` стиле имеет больше приоритет, чем CSS
- `!important` - устанавливает наивысший приоритет
- При прочих равных - последний имеет больше приоритет

Ссылки

- Спецификация по CSS: <https://www.w3.org/Style/CSS>
- Гайд по CSS от MDN: <https://developer.mozilla.org/ru/docs/Learn/CSS>
- Учебник от web.dev: <https://web.dev/learn/css/>
- Документация по CSS от MDN:
<https://developer.mozilla.org/ru/docs/Web/CSS/Reference>
- CSS Layouts от MDN (про потоки рендеринга):
https://developer.mozilla.org/ru/docs/Learn/CSS/CSS_layout
- "Для чайников": <https://webref.ru/>

In the next episode

Flex-box, BEM и "как верстать"