**Знакомство**

|  |
| --- |
| [Поехали!](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/1) |
| [CSS в действии](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/2) |
| [Закрепление](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/3) |
| [Азы HTML](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/4) |
| [Одиночные HTML-теги](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/5) |
| [Атрибуты HTML-тегов](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/6) |
| [Ищем ошибки](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/7) |
| [Азы CSS](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/8) |
| [Другие способы подключения CSS](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/9) |
| [Селекторы в CSS](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/10) |
| [Классы в CSS](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/11) |
| [Свойства и значения CSS](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/12) |
| [Работа над ошибками](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/13) |
| [Первое испытание](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/14) |

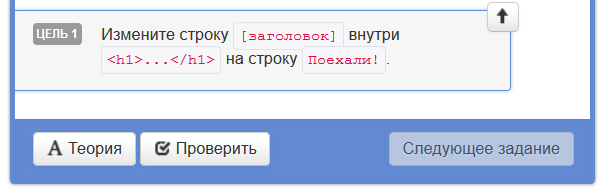
# Поехали! [1/14]

Наша академия посвящена изучению искусства по созданию веб-страниц. Веб-страницы (или HTML-страницы) — это простейшие кирпичики, из которых состоят все сайты, а значит, весь Интернет.

Для создания веб-страниц используются языки HTML и CSS. HTML отвечает за структуру и содержание страницы, CSS — за внешний вид.

Мы будем развивать ваши умения владения HTML и CSS. Начнём с простейших вещей, а затем доведём ваши навыки до совершенства с помощью сложнейших испытаний.

Перед собой вы видите учебный интерфейс. Он состоит из нескольких частей.

1. Слева находятся два редактора кода, в которых вам предстоит работать.
2. Справа — мини-браузер, где отображается страница, полученная из вашего кода в редакторах.
3. А в нижней части мини-браузера находятся цели и управляющие кнопки.  
   

Когда вы выполняете цель, она загорается жёлтым цветом. Чтобы пройти задание, нужно выполнить все цели.

# CSS в действии [2/14]

[Как уже говорилось](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/1), при создании веб-страниц используются два языка: HTML и CSS. HTML отвечает за структуру и содержание, а CSS — за оформление. Браузер объединяет HTML- и CSS-код и формирует внешний вид страницы.

В прошлом задании вы изменяли код в HTML-редакторе и убедились, что при этом меняется содержание страницы.

В этом задании вы поработаете с CSS-редактором и увидите, как с помощью нескольких строк кода можно изменить оформление страницы.

Сейчас мы не будем углубляться в значение каждого CSS-свойства. А просто посмотрим на CSS в действии! Чтобы включить CSS-код, нужно будет удалить символы комментариев, которые «выключают» его.

# Закрепление [3/14]

Первые задания были самыми простыми. Дальше придётся выполнять более сложные требования. Например, добавлять или удалять определённые теги или их атрибуты.

Чтобы выполнить это задание, нужно отредактировать подготовленный код. Попробуйте сами разобраться для чего предназначены теги и свойства в примере. Если не получится, то откройте подсказку.

Кстати, обратите внимание на кнопку Проверить в нижней части мини-браузера.

Обычно ваш код проверяется на лету в вашем же браузере, и в 95% случаев проверки работают правильно. Но иногда в некоторых браузерах проверки ломаются. Вот тогда и нужно пользоваться этой кнопкой: она отправляет код на наш сервер, где он проверяется в идеальных условиях. Если серверная проверка не прошла, то значит есть ошибка в коде.

# Азы HTML [4/14]

[В первых заданиях](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/3) вы познакомились с обучающими инструментами, а теперь мы приступаем к знакомству с HTML.

Язык HTML состоит из тегов. Теги — это те самые кирпичики, из которых построена каждая веб-страница.

Каждый тег начинается с символа < и заканчивается символом >, например: <p>.

Все теги можно разделить на парные и одиночные. Каждый парный тег состоит из двух частей: открывающего тега и закрывающего. Внутри закрывающего тега используется символ /.

Вот пример парного тега:

<p>Абзац</p>

Как вы видите, у тега <p> есть пара в виде закрывающего тега </p>.

# Одиночные HTML-теги [5/14]

С некоторыми [парными тегами](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/4) мы познакомились. А что же за одиночные теги?

Парные теги обычно нужны, чтобы оформить некоторый участок текста. Благодаря паре тегов вы можете указать начало и конец этого участка. Но ведь есть теги, которые не предназначены для оформления фрагментов текста.

Например, тег для вставки изображения или тег для вставки разделительной полосы. Такие теги добавляют на страницу одиночный объект, и им не нужно для этого заключать в себя какой-то текст. Поэтому их называют одиночными.

Примеры таких тегов: <br>, <hr>, <img>.

Кстати, в HTML-редакторе вы увидите такие фрагменты кода: <!-- текст -->. Они называются «комментарии», и браузер не отображает их на странице.

Раньше одиночные теги писались с закрывающим слешом перед закрывающей скобкой. Например: <br/>.  
В новом стандарте HTML5 использование закрывающего слеша в одиночных тегах необязательно.

# Атрибуты HTML-тегов [6/14]

[В предыдущем задании](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/5), как вы заметили, после вставки в код тега <img> ничего не произошло. Почему же так вышло?

Теги могут иметь атрибуты. Некоторые теги есть смысл использовать только с атрибутами. Наиболее яркий пример — тег <img>, обозначающий изображение. Для него обязательно нужно указывать атрибут src, который задаёт адрес картинки (иначе браузер не сможет загрузить её).

В общем случае тег записывается следующим образом:

<имя-тега атрибут1="значение1" атрибут2="значение2" ...>

Атрибутов может быть несколько, вот примеры:

<p class="important">...</p>

<a class="external" href="https://htmlacademy.ru">...</a>

<img class="avatar" src="keks.png">

В этом задании вы потренируетесь использовать атрибуты тегов.

Не забудьте пробелы между названием тега и атрибутом и между атрибутами

# Ищем ошибки [7/14]

Познакомимся с некоторыми часто встречающимися ошибками, чтобы в дальнейшем их не допускать.

HTML позволяет вкладывать теги друг в друга, и одна из самых частых ошибок заключается в неправильной вложенности, например:

<p>Текст <strong>выделен</p> полужирным</strong>

В этом примере тег <p> закрывается раньше, чем тег <strong>, и это ошибка.

Другой тип ошибок случается из-за досадных опечаток и невнимательности, когда забывают пробелы между атрибутами тега или неправильно пишут их названия.

Вот и инструктор Кекс решил обновить своё резюме поздно ночью. Захотелось сделать его лаконичнее, поиграться с оформлением. Но невнимательность его подвела. Зато вы можете потренироваться отыскивать и исправлять ошибки.

Если совсем потеряетесь, то внизу есть подсказка. Только чур сразу не смотреть!

# Азы CSS [8/14]

CSS — это язык для управления внешним видом HTML-документа. С помощью CSS можно задавать параметры отображения любого тега: ширину и высоту, отступы, цвет и размер шрифта, фон и так далее.

CSS это аббревиатура «Cascading Style Sheets» или «Каскадные Таблицы Стилей». Обычно CSS называют просто «стили».

Самый простой способ применить стили к тегу заключается в использовании атрибута style. Например:

<p style="color: red;">...</p>

В этом примере абзацу задан красный цвет шрифта. Такой способ задания стилей называют «инлайновые стили» или «встроенные стили».

Синтаксис таких стилей очень простой: свойство: значение;. Причём свойств может быть несколько.

Теперь давайте составим мини-конспект раздела и отметим зелёным цветом те темы, которые мы уже отработали на практике. Для стилизации используем инлайновые стили.

# Другие способы подключения CSS [9/14]

Задавать стили каждого тега с помощью атрибута style очень затратно и хлопотно. А ещё это приводит к засорению HTML-кода избыточными, повторяющимися кусками CSS.

К счастью, есть и другие способы подключения стилей. Первый — подключение внешнего файла с помощью тега <link>, а второй — использование специального тега <style>. Подробнее эти методы будут разобраны в последующих разделах.

А сейчас вы познакомитесь со вторым разделом редактора, помеченным как CSS. Код из CSS-редактора подставляется в HTML-документ так, как будто вы записали его в тег <style>.

Для начала мы почистим код в HTML-редакторе, а потом начнём пошагово стилизовать наш мини-конспект с помощью других возможностей CSS.

# Селекторы в CSS [10/14]

Поздравляем! Вы только что [подключили внешние CSS-стили](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/9) к своей HTML-страничке.

Когда вы задаёте стили тега с помощью атрибута style, браузер сразу же понимает, к какому именно тегу применить эти стили. Но когда стили подключаются с помощью внешнего файла или через тег <style>, браузер ищет стилизуемые теги с помощью «селекторов».

С селекторами вы уже немного знакомы: в предыдущем задании вы использовали селектор p, который находился перед фигурными скобочками в CSS-коде. В общем случае синтаксис CSS-правил выглядит так:

селектор {

свойство1: значение1;

свойство2: значение2;

...

}

Язык селекторов очень мощный и гибкий. Простейший тип селекторов — селекторы по имени тега: p, h1 и так далее. Когда браузер видит такой селектор, он применяет стили из правила ко всем подходящим тегам. Например, ко всем абзацам или ко всем заголовкам первого уровня.

Но, Хьюстон, у нас проблемы с конспектом! Селектор p подсветил все абзацы зелёным. А они должны быть разного цвета. Как быть?

Давайте попробуем добавить внутрь абзацев разные теги для разных цветов. Тогда в CSS можно будет использовать разные селекторы.

# Классы в CSS [11/14]

Что ж, [конспект мы снова раскрасили](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/10), но какой ценой. Давайте признаем, что добавлять внутрь абзацев дополнительные теги и раскрашивать текст с их помощью, не самое лучшее решение. Ну а какое решение хорошее?

Такое решение есть и оно называется «классы»!

Вспомните шаг, когда мы пытались стилизовать абзацы с помощью внешних стилей, но все они получились одинакового цвета.

Нужно было сделать следующее: добавить абзацам разные классы, а в CSS использовать селекторы не по тегу, а по классам.

Класс — это всего лишь один из атрибутов HTML-тегов, например:

<p class="important">...</p>

<p class="help">...</p>

В CSS можно задавать стили только для элементов с определённым классом. Для этого используется селектор по классу, который пишется так .имя-класса, например:

.important { color: red; } — выберет все теги с классом "important"

.help { color: green; } — выберет все теги с классом "help"

Классы гибкие, их можно создавать много и называть понятными именами. Например, можно создать класс, который отмечает раздел курса, который сейчас изучается.

Имя класса может содержать в себе латинские символы, цифры, символ дефиса - и подчёркивания \_ и начинаться оно должно с латинского символа.

# Свойства и значения CSS [12/14]

Итак, язык CSS состоит из [селекторов](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/10) и свойств. Селекторы указывают на то, к каким элементам применять стили, а свойства — на то, как именно отображать элементы.

Существует огромное количество CSS-свойств, которые влияют практически на все аспекты отображения элементов. Причём каждому свойству соответствует определённый набор значений.

Некоторые значения задаются с помощью текстовых констант, например red, bold, другие с помощью цифровых значений: 12px, 120% и так далее.

Мощь стилей заключается в том, что вы можете быстро и гибко менять внешний вид нужных элементов, особенно когда используете классы. Например, чтобы зачеркнуть текст всех изученных пунктов конспекта, нужно добавить всего лишь одну строчку в CSS:

.learned-ok {

color: green;

**text-decoration: line-through;**

}

И все теги с классом .learned-ok будут отображаться с перечёркнутым текстом. Теперь представьте, как долго делать то же самое через инлайновые стили, когда в конспекте больше сотни пунктов.

# Работа над ошибками [13/14]

Синтаксис CSS намного проще синтаксиса HTML, но это не мешает совершать досадные ошибки. [С ошибками в HTML](https://npoed.htmlacademy.ru/htmlcss110/course/1/run/7) мы уже разобрались, теперь посмотрим на ошибки в CSS.

Конечно, одни из самых распространённых ошибок — это опечатки в названиях свойств или селекторах.

К другим частым ошибкам относится отсутствие ; в списке CSS-свойств. Когда после пары «свойство-значение» забывают поставить точку с запятой, браузер не применяет ни это свойство, ни все последующие.

Когда вёрстка конспекта была почти завершена, к компьютеру шерстяной молнией подлетел инструктор Кекс и решил напоследок поиграться со стилями, чтобы сделать конспект «симпатичнее». Получилось неплохо, но в этот раз Кекс был спросонья и наделал досадных опечаток, которые сломали некоторые стили.

Ну и замечательно! Теперь есть возможность потренироваться в поиске ошибок в CSS!

# Первое испытание [14/14]

Испытания — это самые сложные задания разделов. Проходя испытание, вы доказываете, что хорошо усвоили материал.

Главная цель испытания — с помощью HTML и CSS получить в мини-браузере изображение, максимально похожее на образец. Чтобы система сравнила ваш результат с образцом, нужно нажать кнопку  Проверить.

Первое испытание достаточно простое. Вам необходимо разметить исходный HTML-код с помощью тегов и классов.

Подсказка: CSS менять не надо. Достаточно использовать правильные теги и классы.

Обратите внимание, что в учебном интерфейсе появилась панель со вкладками: «Образец», «Результат», «Наложение» и «Различия».

**Важно!** Во вкладке «Результат» отображается то, как ваш код видит наш сервер. Именно эту картинку он сравнивает с образцом.

Используйте эту вкладку чаще, ведь иногда изображения в вашем мини-браузере и на сервере могут отличаться. Особенно часто отличаются шрифты.

У вас есть 10 проверок испытания в день. Проверки снимаются и при перезагрузке страницы.

Если не получится пройти испытание сегодня, то можете переходить к следующим заданиям, а к испытанию вернуться завтра.