



Конспект «Циклы». Раздел 1

Цикл for

Синтаксис

```
for (let i = 0; i < 10; i++) {  
  // Повторяющиеся команды  
}
```

В круглых скобках записывается код управления циклом. Он состоит из трёх частей, разделённых `;`.

1. Первая часть — подготовительная. Команды отсюда запускаются *один раз* перед началом работы цикла. Обычно здесь задаётся исходное значение для переменной-счётчика. Обратите внимание, что в цикле мы создаём переменную-счётчик с помощью `let`, как в случае с любой другой переменной.

```
for (let i = 0; i < 5; i = i + 1) { }
```

2. Вторая часть — проверочная. Она содержит условие и запускается *перед* каждым новым витком цикла. Если условие возвращает `true`, цикл делает ещё один виток, иначе цикл завершает свою работу.

```
for (let i = 0; i < 5; i = i + 1) { }
```

3. Третья часть — дополняющая, или «закон изменения». Код третьей части запускается *после* каждого витка цикла. Обычно там изменяется переменная-счётчик.

```
for (let i = 0; i < 5; i = i + 1) { }
```

Накопление значений в цикле:

Внутри циклов можно использовать обычные математические операции. Например, сложение:

```
let sum = 0;

for (let i = 1; i <= 5; i++) {
  sum += 2;
  console.log(sum);
}
```

Программа выведет:

```
LOG: 2 (number)
LOG: 4 (number)
LOG: 6 (number)
LOG: 8 (number)
LOG: 10 (number)
```

Проверки в теле цикла

Если добавить условие внутри цикла, то оно будет проверяться на каждой итерации.

```
let sum = 0;

for (let i = 1; i <= 5; i++) {
  if (i > 2) {
    sum += 1;
  }
}
```

Поиск чётного числа

Оператор `%`, или «остаток от деления», возвращает остаток от деления.

```
10 % 5; // Вернёт 0
12 % 5; // Вернёт 2
7 % 3;  // Вернёт 1
5.5 % 2; // Вернёт 1.5
```

Если остаток от деления числа на `2` равен `0` — число чётное, иначе нечётное.

Сокращённые операторы

В JavaScript есть несколько удобных операторов, которые позволяют сократить код:

Название	Пример	Аналог
Инкремент (увеличение на единицу)	<code>i++</code>	<code>i = i + 1</code>
Декремент (уменьшение на единицу)	<code>i--</code>	<code>i = i - 1</code>
К-к-комбо!	<code>i += 2</code>	<code>i = i + 2</code>

Комбинировать можно не только сложение, но и остальные математические операции: вычитание `-=`, умножение `*=`, деление `/=` и нахождение остатка `%=`.

Продолжить

Хотите верстать адаптивно и по методологии, использовать препроцессоры и автоматизацию? Записывайтесь на профессиональный курс [«HTML и CSS. Адаптивная вёрстка и автоматизация»](#). Цена **12 000 ₽**.



Практикум

Курсы для новичков

Подписка

Профессии

Фронтенд-разработчик

JavaScript-разработчик

Фулстек-разработчик

Услуги

Работа наставником

Для вузов и колледжей

Для учителей

Журнал

Справочник по HTML

Учебник по Git

Учебник по PHP

Соглашение

Конфиденциальность

Сведения об образовательной организации

Лицензия № ЛО35-01271-78/00176657



© ООО «Интерактивные обучающие технологии», 2013–2025

Курсы

HTML и CSS. Профессиональная вёрстка сайтов

HTML и CSS. Адаптивная вёрстка и автоматизация

JavaScript. Профессиональная разработка веб-интерфейсов

JavaScript. Архитектура клиентских приложений

React. Разработка сложных клиентских приложений

Node.js. Профессиональная разработка REST API

Node.js и Nest.js. Микросервисная архитектура

TypeScript. Теория типов

Алгоритмы и структуры данных

Паттерны проектирования

Webpack

Vite

Vue.js 3. Разработка клиентских приложений

Git и GitHub

Анимация для фронтендеров

Информация

Об Академии

О центре карьеры

Остальное

Написать нам

Мероприятия

Форум

Акции

Отзывы о курсах