

Раздел

[Основы JavaScript](#)

Навигация по уроку


Инструкции

Точка с запятой

Комментарии

Комментарии

Поделиться

[Редактировать на GitHub](#)[🏠 → Язык программирования JavaScript](#)  
[→ Основы JavaScript](#) 5-го сентября 2020

## Структура кода

Начнём изучение языка с рассмотрения основных «строительных блоков» кода.

### Инструкции

Инструкции – это синтаксические конструкции и команды, которые выполняют действия.

Мы уже видели инструкцию `alert('Привет, мир!')`, которая отображает сообщение «Привет, мир!».

В нашем коде может быть столько инструкций, сколько мы захотим. Инструкции могут отделяться точкой с запятой.

Например, здесь мы разделили сообщение «Привет Мир» на два вызова `alert`:

```
1 alert('Привет'); alert('Мир');
```



Обычно каждую инструкцию пишут на новой строке, чтобы код было легче читать:

```
1 alert('Привет');  
2 alert('Мир');
```



### Точка с запятой

В большинстве случаев точку с запятой можно не ставить, если есть переход на новую строку.

Так тоже будет работать:

```
1 alert('Привет')  
2 alert('Мир')
```



В этом случае JavaScript интерпретирует перенос строки как «неявную» точку с запятой. Это называется [автоматическая вставка точки с запятой](#).

**В большинстве случаев новая строка подразумевает точку с запятой. Но «в большинстве случаев» не значит «всегда»!**

В некоторых ситуациях новая строка всё же не означает точку с запятой. Например:

```
1 alert(3 +  
2 1  
3 + 2);
```



Код выведет 6, потому что JavaScript не вставляет здесь точку с запятой. Интуитивно очевидно, что, если строка заканчивается знаком "+", значит, это «незавершённое выражение», поэтому точка с запятой не требуется. И в этом случае всё работает, как задумано.

**Но есть ситуации, где JavaScript «забывает» вставить точку с запятой там, где она нужна.**

Ошибки, которые при этом появляются, достаточно сложно обнаруживать и исправлять.

Раздел

Основы JavaScript

Навигация по уроку

Инструкции

Точка с запятой

Комментарии

Комментарии

Поделиться



Редактировать на GitHub



### Пример ошибки

Если вы хотите увидеть конкретный пример такой ошибки, обратите внимание на этот код:

```
1 [1, 2].forEach(alert)
```



Пока нет необходимости знать значение скобок `[]` и `forEach`. Мы изучим их позже. Пока что просто запомните результат выполнения этого кода: выводится `1`, а затем `2`.

А теперь добавим `alert` перед кодом и не поставим в конце точку с запятой:

```
1 alert("Сейчас будет ошибка")
2
3 [1, 2].forEach(alert)
```



Теперь, если запустить код, выведется только первый `alert`, а затем мы получим ошибку!

Всё исправится, если мы поставим точку с запятой после `alert`:

```
1 alert("Теперь всё в порядке");
2
3 [1, 2].forEach(alert)
```



Теперь мы получим сообщение «Теперь всё в порядке», следом за которым будут `1` и `2`.

В первом примере без точки с запятой возникает ошибка, потому что JavaScript не вставляет точку с запятой перед квадратными скобками `[...]`. И поэтому код в первом примере выполняется, как одна инструкция. Вот как движок видит его:

```
1 alert("Сейчас будет ошибка")[1, 2].forEach(alert)
```



Но это должны быть две отдельные инструкции, а не одна. Такое слияние в данном случае неправильное, оттого и ошибка. Это может произойти и в некоторых других ситуациях.

Мы рекомендуем ставить точку с запятой между инструкциями, даже если они отделены переносами строк. Это правило широко используется в сообществе разработчиков. Стоит отметить ещё раз – в большинстве случаев *можно* не ставить точку с запятой. Но безопаснее, особенно для новичка, ставить её.

## Комментарии

Со временем программы становятся всё сложнее и сложнее. Возникает необходимость добавлять *комментарии*, которые бы описывали, что делает код и почему.

Комментарии могут находиться в любом месте скрипта. Они не влияют на его выполнение, поскольку движок просто игнорирует их.

**Однострочные комментарии начинаются с двойной косой черты `//`.**

Часть строки после `//` считается комментарием. Такой комментарий может как занимать строку целиком, так и находиться после инструкции.

Как здесь:

```
1 // Этот комментарий занимает всю строку
2 alert('Привет');
```



```
3
4 alert('Мир'); // Этот комментарий следует за инструкции
```



**Многострочные комментарии начинаются косой чертой со звёздочкой `/*` и заканчиваются звёздочкой с косой чертой `*/`.**



Как вот здесь:

```
1 /* Пример с двумя сообщениями.
2 Это - многострочный комментарий.
3 */
4 alert('Привет');
5 alert('Мир');
```



Содержимое комментария игнорируется, поэтому, если мы поместим код внутри `/* ... */`, он не будет исполняться.

Это бывает удобно для временного отключения участка кода:

```
1 /* Закомментировали код
2 alert('Привет');
3 */
4 alert('Мир');
```



#### **Используйте горячие клавиши!**

В большинстве редакторов строку кода можно закомментировать, нажав комбинацию клавиш `Ctrl+/*` для однострочного комментария и что-то вроде `Ctrl+Shift+/*` – для многострочных комментариев (выделите кусок кода и нажмите комбинацию клавиш). В системе Mac попробуйте `Cmd` вместо `Ctrl` и `Option` вместо `Shift`.

#### **Вложенные комментарии не поддерживаются!**

Не может быть `/*...*/` внутри `/*...*/`.

Такой код «умрёт» с ошибкой:

```
1 /*
2   /* вложенный комментарий ?!? */
3 */
4 alert('Мир');
```



Не стесняйтесь использовать комментарии в своём коде.

Комментарии увеличивают размер кода, но это не проблема. Есть множество инструментов, которые минифицируют код перед публикацией на рабочий сервер. Они убирают комментарии, так что они не содержатся в рабочих скриптах. Таким образом, комментарии никоим образом не вредят рабочему коду.

Позже в учебнике будет глава [Качество кода](#), которая объяснит, как лучше писать комментарии.

Проводим [курсы по JavaScript и фреймворкам](#).



Присоединиться к обсуждению...

войти с помощью

или через DISQUS ?

Имя

Загрузить ещё комментарии

✉ Подписаться

📌 Добавь Disqus на свой сайтДобавить DisqusДобавить

⚠ Do Not Sell My Data

Раздел

Основы JavaScript

Навигация по уроку

Инструкции

Точка с запятой

Комментарии

Комментарии

Поделиться



Редактировать на GitHub

