Немного о синтаксисе JavaScript.

Рассмотрим 2 выражения кода JavaScript:
document.body.innerHTML = "Какой-то новый HTML-контент";
document.getElementById('clock').innerHTML = h + "час" + m + "мин" + s + "сек";
Они разделены точками на 3 части. Как называется каждая часть выражения?

У обьекта document есть методы и есть свойства. Например, в выражении document.body.innerHTML, в нём body – это свойство. Во втором коде getElementById('clock') – это метод.

Методы и свойства в JavaScript: в чём различия?
В JavaScript, как и во многих других языках программирования, 'свойства' и 'методы' - это два ключевых понятия, которые описывают, как мы взаимодействуем с объектами.

Свойства – это характеристики объекта, которые хранят информацию о нем. Это может быть текст, число, массив, другой объект или что-либо еще. Свойства обычно **не изменяют состояние объекта**, а лишь предоставляют доступ к хранимым данным. Обычно, но не всегда.

Например:
document.body.innerHTML; – так мы получаем содержимое тега '<body>'.
document.body.innerHTML = "Какой-то новый HTML-контент"; – так мы изменяем содержимое тега '<body>', меняя его состояние.
Некоторые свойства объекта действительно могут изменять его состояние, особенно в контексте DOM (Document Object Model).

Почему так?

- Свойства в JavaScript – это не просто данные, а функции. Они могут быть getter (получают значение) или setter (устанавливают значение).
- innerHTML – это не просто свойство, а специальный 'getter/setter', который работает с HTML-контентом.

Объяснение:
1. document.body.innerHTML – getter – читает HTML-код, находящийся в теле документа.
2. document.body.innerHTML = "Новый текст" – setter – записывает новый HTML-код в тело документа, заменяя старый.

Важно понимать:

- Не все свойства объектов ведут себя как 'innerHTML'.
- Многие свойства просто предоставляют доступ к хранимым данным и не меняют состояние объекта.
- Свойства DOM (document.body, document.createElement и так далее) часто

оснащены getter/setter методами, которые могут как получать, так и изменять состояние.

Методы – это действия, которые можно выполнить с объектом. Они могут **изменять состояние объекта** или возвращать какое-либо значение. Методы представляют собой функции, которые связаны с конкретным объектом.

Пример (без innerHTML):

- document.body: 'body' – это свойство объекта 'document'. Оно содержит ссылку на HTML-элемент <body>.
- document.getElementById('clock'): 'getElementById' - это метод объекта 'document'. Он ищет элемент по заданному 'id' и изменяет его.

Как их правильно называть?

- Свойства обычно описываются именем характеристики, которую они представляют: 'innerHTML', 'tagName', 'id', 'style' и так далее.
- Методы обычно описываются глаголом, который отражает действие, которое они выполняют: 'getElementById', 'addEventListener', 'createElement', 'appendChild' и так далее.

Важно:

- Свойства обычно получаются с помощью точечной нотации: 'document.body'
- Методы вызываются с помощью круглых скобок: 'document.getElementById('clock')'

Дополнительные примеры:

- Свойства объекта 'window': 'innerWidth', 'innerHeight', 'location', 'navigator'
- Методы объекта 'string': 'toUpperCase', 'toLowerCase', 'trim', 'split', 'replace'

Методы и свойства являются основными строительными блоками для работы с объектами в JavaScript. Понимание их различий – ключ к созданию функционального и эффективного кода.

Получается, что в выражении JavaScript: document.body.innerHTML; у одного обьекта 2 свойства? Это свойство 'body' и свойство 'innerHTML'. В коде изменения цвета фона документа: document.body.style.backgroundColor = "yellow"; у одного обьекта 3 свойства?

Нет, не так. Давайте разберемся, что происходит с объектами в JavaScript.

Пример 1: 'document.body.innerHTML;'

- 'document': Это глобальный объект, который представляет собой сам HTML-документ. В нем хранится вся информация о структуре и содержимом страницы.
- 'body': Это свойство объекта 'document', которое ссылается на HTML-элемент

'<body>'. Он является корневым элементом для всего содержимого веб-страницы.
- 'innerHTML': Это свойство объекта 'body', которое представляет собой строку HTML-кода, содержащуюся внутри элемента '<body>'.

То есть, в данном случае у нас не два свойства одного объекта. У нас цепочка:
1. Объект 'document'.
2. Его свойство 'body' (одновременно объект HTML-элемента '<body>').
3. Свойство 'innerHTML' этого объекта 'body'.

Пример 2: 'document.body.style.backgroundColor = "yellow";'
- 'document': Опять же, глобальный объект, представляющий документ.
- 'body': Свойство 'document', ссылающееся на HTML-элемент '<body>'.
- 'style': Свойство объекта 'body', которое содержит все стили, примененные к элементу '<body>'. Оно, в свою очередь, является объектом, содержащим разные стили (например, 'backgroundColor).
- 'backgroundColor': Это свойство объекта 'style', которое задает цвет фона элемента.

И снова, мы не имеем дело с тремя свойствами одного объекта. Это цепочка:
1. Объект 'document'.
2. Его свойство 'body' (объект HTML-элемента '<body>').
3. Свойство 'style' этого объекта (объект стилей).
4. Свойство 'backgroundColor' объекта 'style'.

Важный момент: JavaScript позволяет работать с объектами, которые содержат другие объекты. Это очень удобный инструмент, позволяющий организовать данные иерархически.

Теперь о другом.
В коде JavaScript можно оставлять пустую строку внутри скрипта? Это является ошибкой или нет?
Например, так:

```
<script>
function clock() {
const today = new Date();
let h = today.getHours();
let m = today.getMinutes();

let s = today.getSeconds();
m = checkTime(m);
s = checkTime (s);
document.getElementById('txt').innerHTML = h + ”:” + m + ”:” + s;

setTimeout(startTime, 1000);
```

```
}
</script>
```

Пустые строки в коде JavaScript допустимы и не являются ошибкой. Они используются для разделения блоков кода и улучшения читаемости.

Ещё одно простое утверждение, которого нужно придерживаться.
Метод document.write() используется только для тестирования, а не для применения в самом коде. Применяйте другие механизмы вывода результата.