**Практическое занятие №17**

**Основы JavaScript**

# Цель: Освоить основы JavaScript

# Структура кода

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

**Написание команд:**

Например, можно вместо одного вызова alert сделать два:

<script>

alert('Привет'); alert('Мир');

</script>

<script>

alert('Привет');

alert('Мир');

</script>

Точку с запятой во многих случаях можно не ставить, если есть переход на новую строку.

<script>

alert('Привет') alert('Мир')

</script>

В этом случае JavaScript интерпретирует переход на новую строчку как разделитель команд и автоматически вставляет «виртуальную» точку с запятой между ними.

Однако, внутренние правила по вставке точки с запятой не идеальны. В примере выше они сработали, но в некоторых ситуациях JavaScript «забывает» вставить точку с запятой там, где она нужна. Таких ситуаций не так много, но они все же есть, и ошибки, которые при этом появляются, достаточно сложно исправлять.

# Поэтому рекомендуется точки с запятой ставить. Сейчас это, фактически, стандарт.

**Написание комментариев**

Комментарии могут находиться в любом месте программы и никак не влияют на ее выполнение. Интерпретатор JavaScript попросту игнорирует их.

*Однострочные комментарии* начинаются с двойного слэша //

<script>

alert('Мир'); // Второе сообщение выводим отдельно

</script>

*Многострочные комментарии* начинаются слешем-звездочкой "/\*" и заканчиваются звездочкой-слэшем "\*/".

Вложенные комментарии не поддерживаются! В этом коде будет ошибка:

<script>

/\*

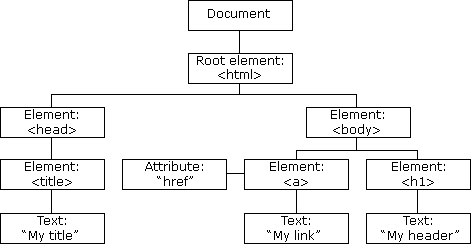
alert('Привет');

/\* вложенный комментарий ?!? \*/

\*/ alert('Мир');

</script>

# Структура DOM (Document Object Model) документа HTML и место Javascript в теле документа



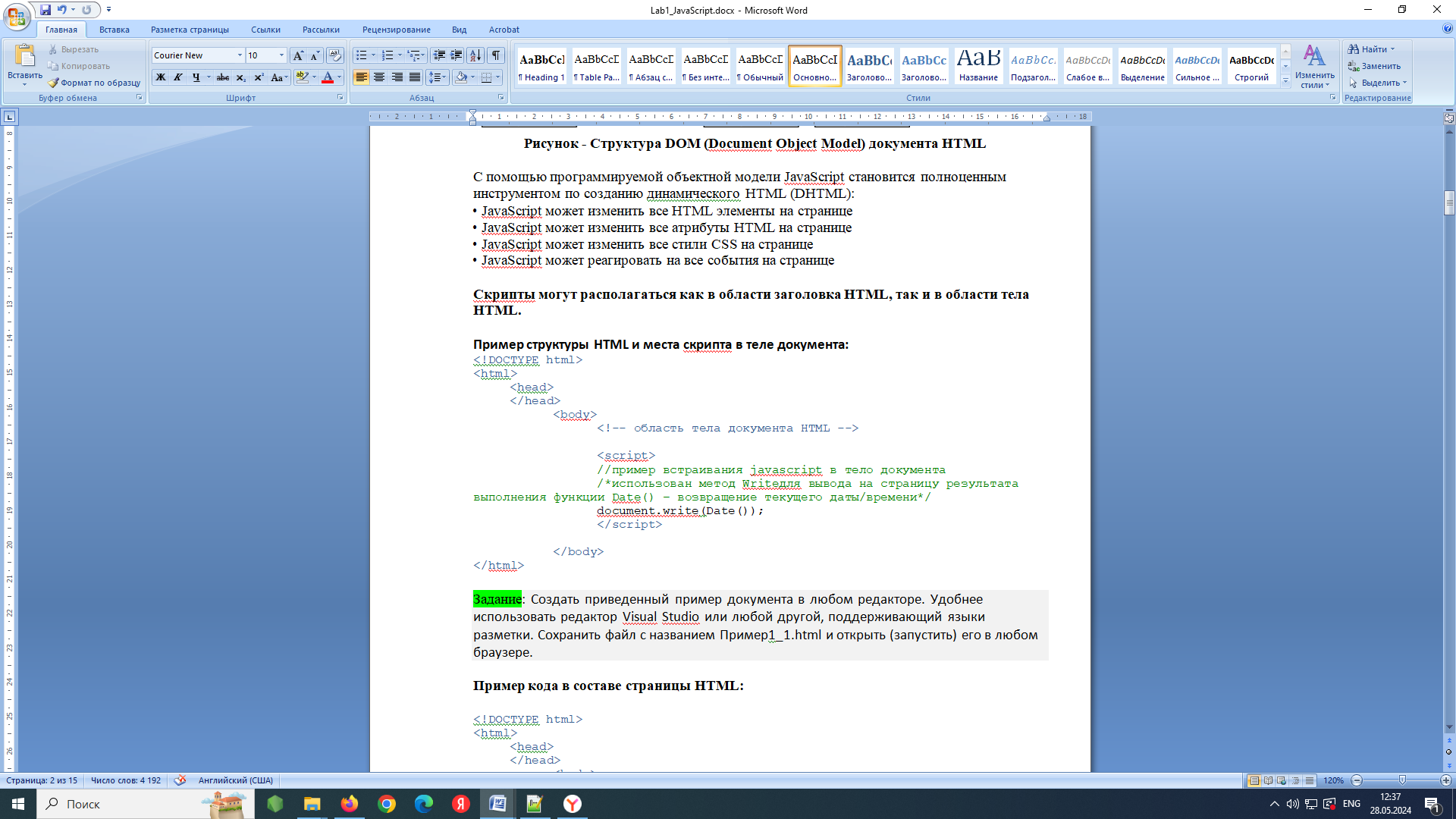
**Рисунок - Структура DOM (Document Object Model) документа HTML**

С помощью программируемой объектной модели JavaScript становится полноценным инструментом по созданию динамического HTML (DHTML):

* JavaScript может изменить все HTML элементы на странице
* JavaScript может изменить все атрибуты HTML на странице
* JavaScript может изменить все стили CSS на странице
* JavaScript может реагировать на все события на странице

# Скрипты могут располагаться как в области заголовка HTML, так и в области тела HTML.

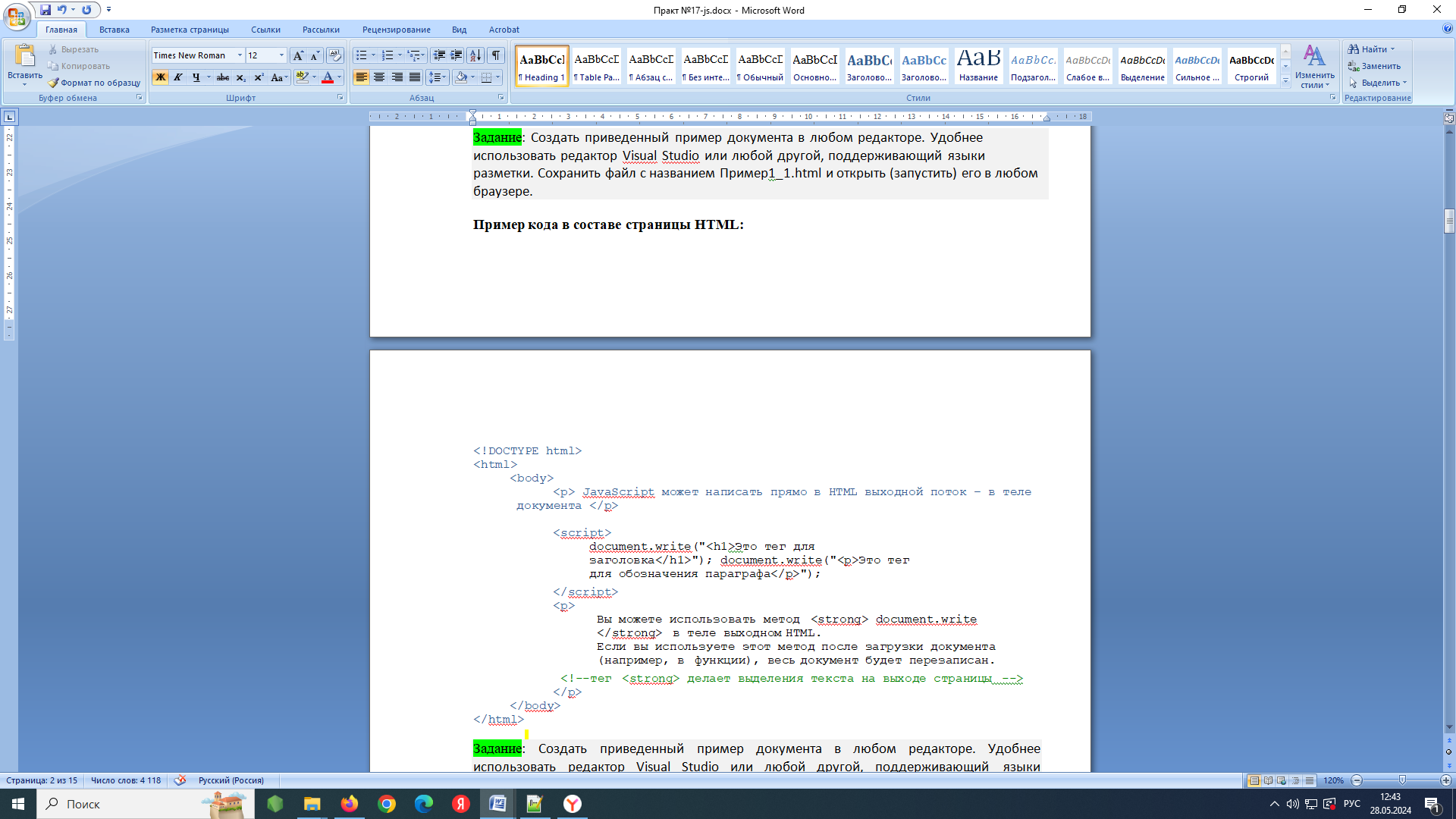
**Пример структуры HTML и места скрипта в теле документа:**



Задание: Создать приведенный пример документа в любом редакторе. Удобнее использовать редактор Visual Studio или любой другой, поддерживающий языки

разметки. Сохранить файл с названием Пример1\_1.html и открыть (запустить) его в любом браузере.

# Пример кода в составе страницы HTML:



Задание: Создать приведенный пример документа в любом редакторе. Удобнее использовать редактор Visual Studio или любой другой, поддерживающий языки разметки. Сохранить файл с названием Пример1\_2.html и открыть (запустить) его в любом браузере.

# Использование переменных

В зависимости от того, для чего создается скрипт, понадобится работать с информацией. ***Переменная* состоит из имени и выделенной области памяти, которая ему соответствует**.

Для *объявления* или, другими словами, *создания переменной* используется ключевое слово

var:

var message;

После объявления, можно записать в переменную данные:

Var message;

message = 'Привет'; // сохраним в переменной строку

Эти данные будут сохранены в соответствующей области памяти и в дальнейшем доступны при обращении по имени:

var message;

message = 'Привет';

alert(message); // выведет содержимое переменной

Для краткости можно совместить объявление переменной и запись данных:

var message = 'Привет';

При изменении значения старое содержимое переменной удаляется.

Переменные в JavaScript могут хранить не только строки, но и другие данные, например, числа.

Объявим две переменные, положим в одну - строку, а в другую - число. Как вы можете видеть, переменной без разницы, что хранить:

Var num = 100500;

var message = 'Привет';

# Значение можно копировать из одной переменной в другую.

var num = 100500;

var message = 'Привет'; message = num;

Значение из num перезаписывает текущее в message.

В JavaScript вы можете создать переменную и без var, достаточно просто присвоить ей значение:

x = "value"; // переменная создана

Технически, это не вызовет ошибки, но делать так все-таки не стоит.

**Всегда определяйте переменные через var**. Это хороший тон в программировании и помогает избежать ошибок.

Задание: Создайте документ HTML, в котором средствами JavaScript:

1. Объявите две переменные: admin и name.
2. Запишите в name строку "Василий".
3. Скопируйте значение из name в admin.
4. Выведите admin (должно вывести «Василий»).

# Объявление констант

*Константа* — это переменная, которая никогда не меняется. Как правило, их называют большими буквами, через подчёркивание. Например:

var COLOR\_BLUE = "#00F";

var COLOR\_RED = "#0F0";

var COLOR\_GREEN = "#F00";

var COLOR\_ORANGE = "#FF7F00";

alert(COLOR\_RED); // #0F0

Технически, константа является обычной переменной, то есть её можно изменить. Но мы договариваемся этого не делать. Зачем нужны константы? Почему бы просто не использовать "#F00" или "#0F0"?

1. Во-первых, константа — это понятное имя, в отличие от строки "#FF7F00".
2. Во-вторых, опечатка в строке может быть не замечена, а в имени константы её упустить невозможно — будет ошибка при выполнении.

# Константы используют вместо строк и цифр, чтобы сделать программу понятнее и избежать ошибок.

**На имя переменной наложены два ограничения:**

1. Имя может состоять из: букв, цифр, символов $ и \_
2. Первый символ не должен быть цифрой.

# !!! Регистр букв имеет значение

Переменные apple и AppLE - две разные переменные.

Существует список зарезервированных слов, которые нельзя использовать при именовании переменных, так как они используются самим языком, например: var, class, return, implements и др.

Некоторые слова, например, class, не используются в современном JavaScript, но они заняты на будущее. Некоторые браузеры позволяют их использовать, но это может привести к ошибкам.

# Типы данных в JavaScript

**Число number**:

var n = 123;

n = 12.345;

# Строка string:

var str = "Мама мыла раму";

str = 'Одинарные кавычки тоже подойдут';

1. **В JavaScript одинарные и двойные кавычки равноправны.** Можно использовать или те или другие.
2. Тип *символ* не существует, есть только *строка*
3. В некоторых языках программирования есть специальный тип данных для одного символа. Например, в языке С это char. В JavaScript есть только тип «строка» string..

**Булевый (логический) тип boolean**. У него всего два значения - true (истина) и

false (ложь).

Такой тип используется для хранения значения типа да/нет, например:

var checked = true; // поле формы помечено галочкой

checked = false; // поле формы не содержит галочки

Мы поговорим более подробно, когда будем обсуждать логические вычисления и условные операторы.

**null — специальное значение.** Оно имеет смысл «ничего». Значение null не относится ни к одному из типов выше, а образует свой отдельный тип, состоящий из единственного значения null:

var age = null;

1. В JavaScript null не является «ссылкой на несуществующий объект» или

«нулевым указателем», как в некоторых других языках. Это просто специальное значение, которое имеет смысл «ничего» или «значение неизвестно».

1. В частности, код выше говорит о том, что возраст age неизвестен.
2. **undefined** — специальное значение, которое, как и null, образует свой собственный тип. Оно имеет смысл «значение не присвоено».

Если переменная объявлена, но в неё ничего не записано, то ее значение как раз и есть undefined:

var u;

alert(u); // выведет "undefined"

Можно присвоить undefined и в явном виде, хотя это делается редко:

Var x = 123;

x = undefined;

В явном виде undefined обычно не присваивают, так как это противоречит его смыслу. Для записи в переменную «пустого значения» используется null.

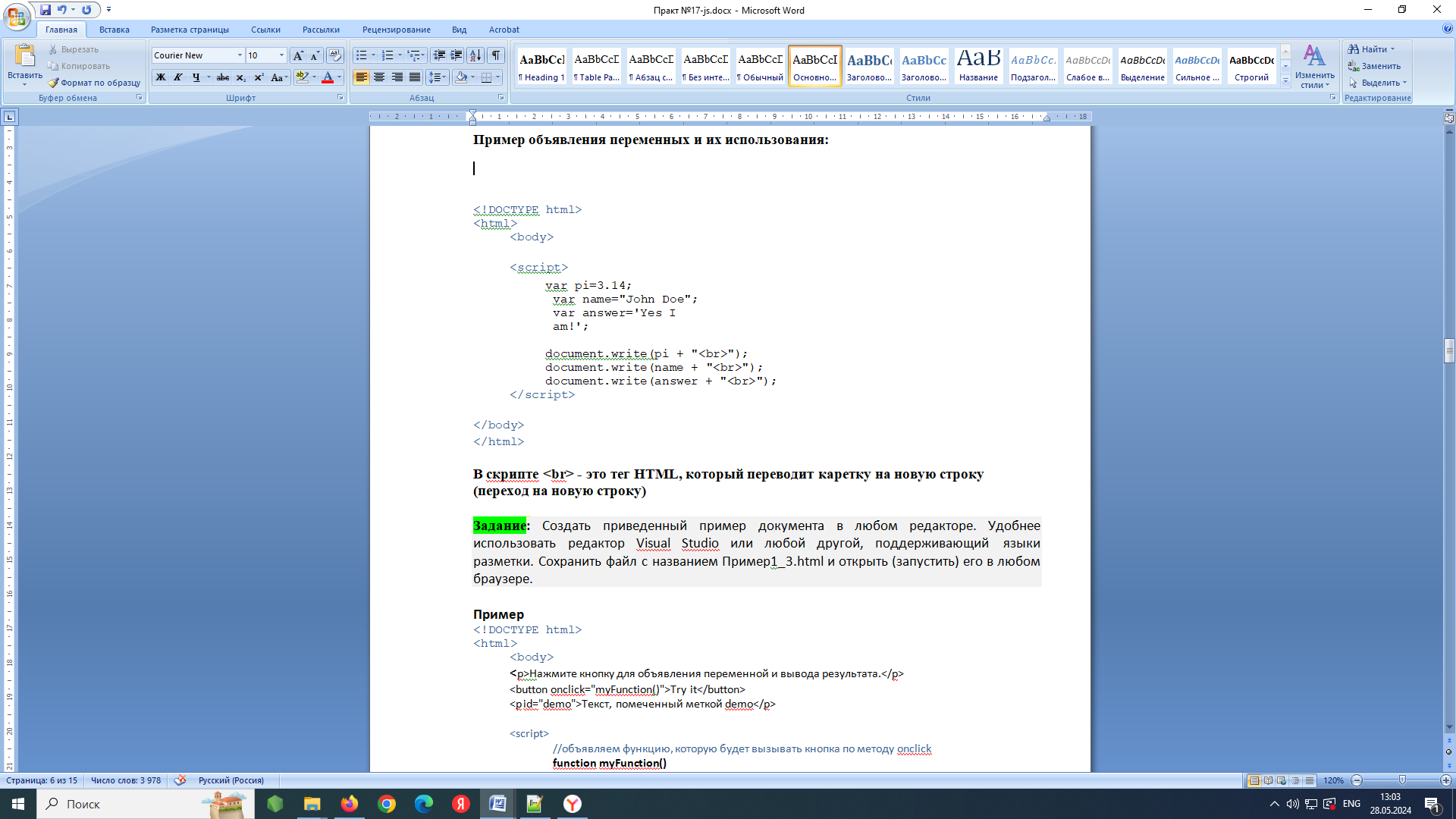
# Объекты object

Первые 5 типов называют *«примитивными»*.

Особняком стоит шестой тип: *«объекты»*. К нему относятся, например, даты, он используется для коллекций данных и для многого другого.

ИТОГО: Есть 5 «примитивных» типов: number, string, boolean, null, undefined и объекты object.

# Пример объявления переменных и их использования:



# В скрипте <br> - это тег HTML, который переводит каретку на новую строку (переход на новую строку)

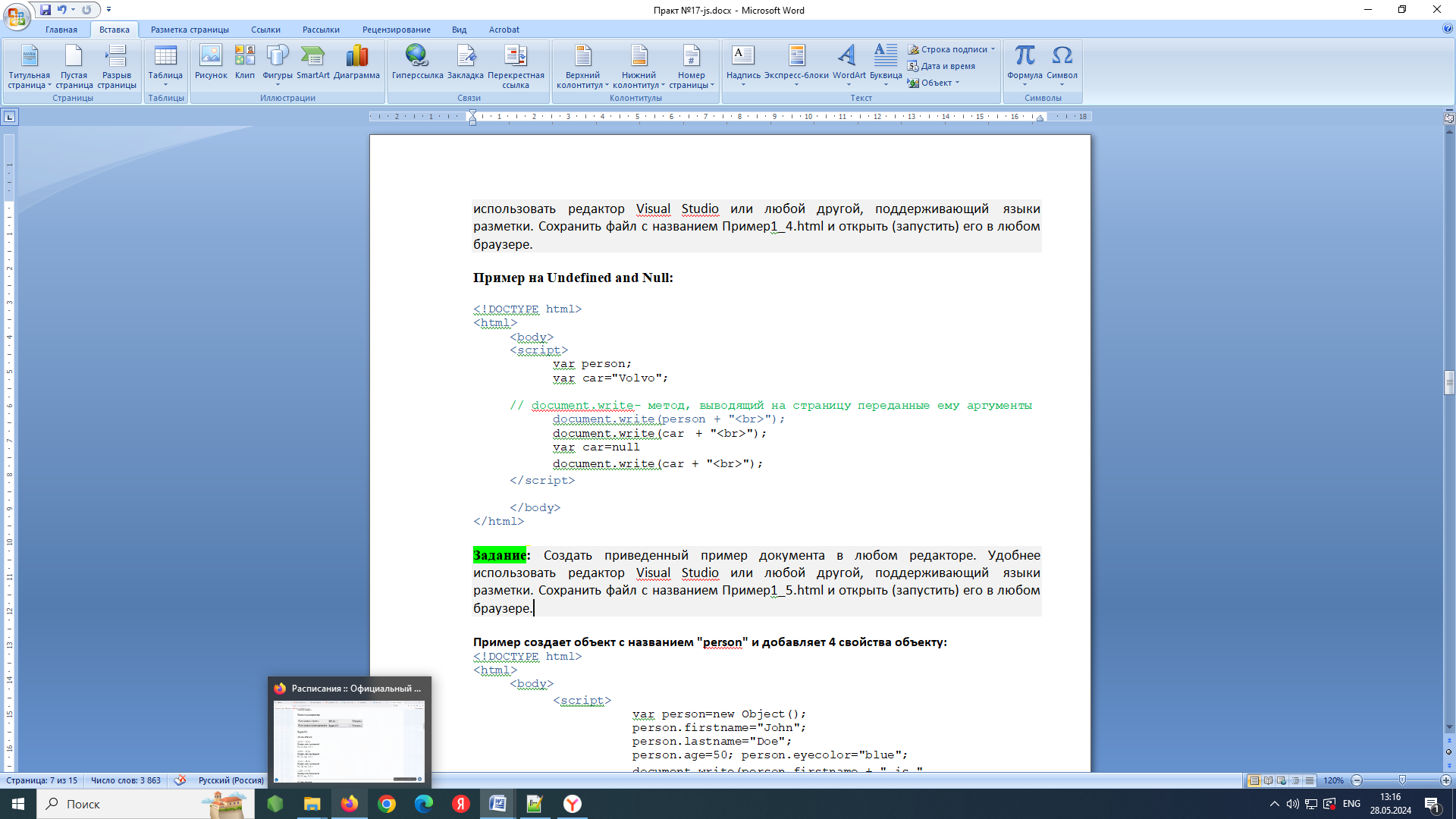
**Задание:** Создать приведенный пример документа в любом редакторе. Удобнее использовать редактор Visual Studio или любой другой, поддерживающий языки разметки. Сохранить файл с названием Пример1\_3.html и открыть (запустить) его в любом браузере.

# Пример

# 

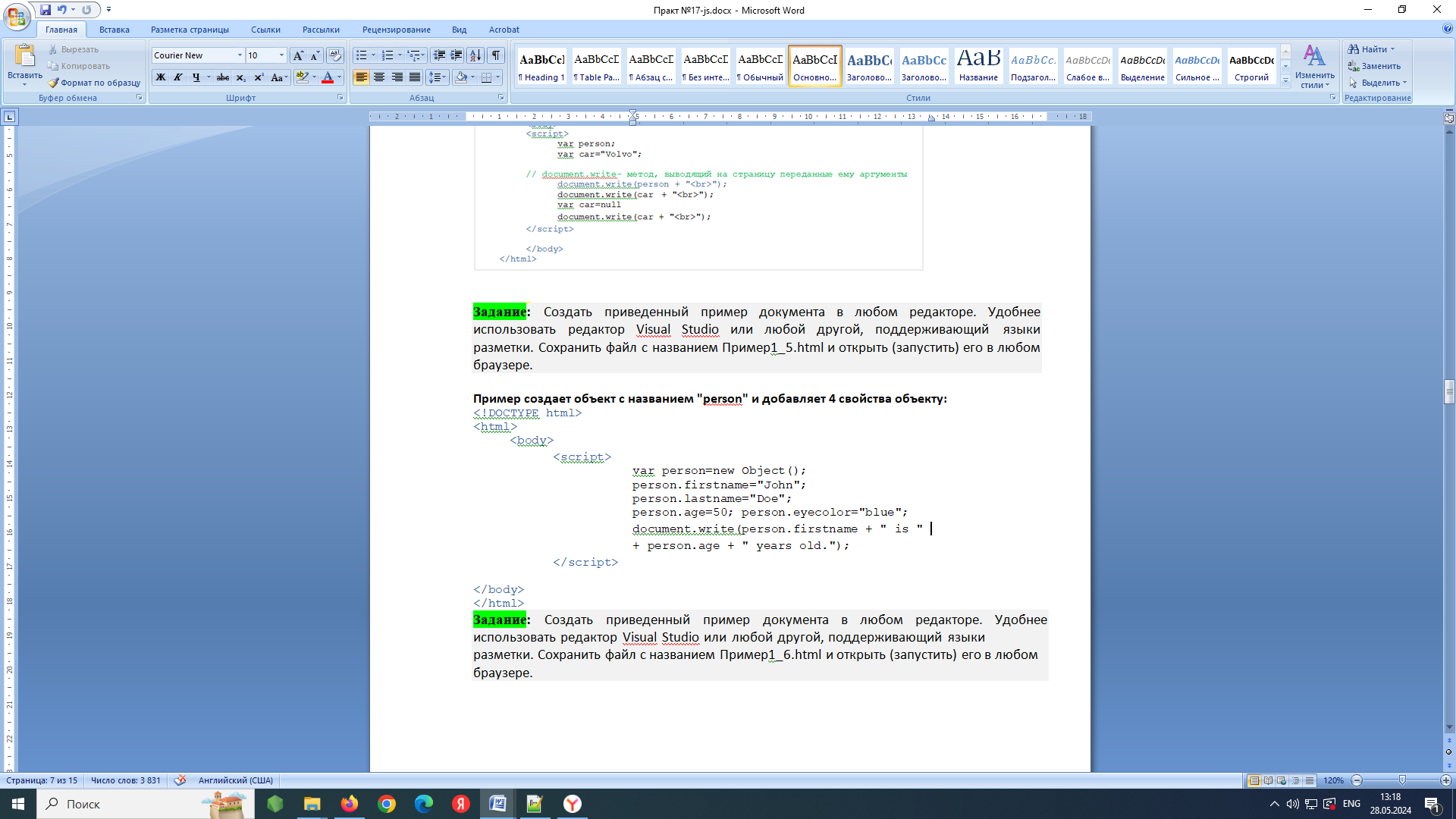
**Задание:** Создать приведенный пример документа в любом редакторе. Удобнее использовать редактор Visual Studio или любой другой, поддерживающий языки разметки. Сохранить файл с названием Пример1\_4.html и открыть (запустить) его в любом браузере.

# Пример на Undefined and Null:



**Задание:** Создать приведенный пример документа в любом редакторе. Удобнее использовать редактор Visual Studio или любой другой, поддерживающий языки разметки. Сохранить файл с названием Пример1\_5.html и открыть (запустить) его в любом браузере.

**Пример создает объект с названием "person" и добавляет 4 свойства объекту:**



**Задание:** Создать приведенный пример документа в любом редакторе. Удобнее использовать редактор Visual Studio или любой другой, поддерживающий языки

разметки. Сохранить файл с названием Пример1\_6.html и открыть (запустить) его в любом браузере.