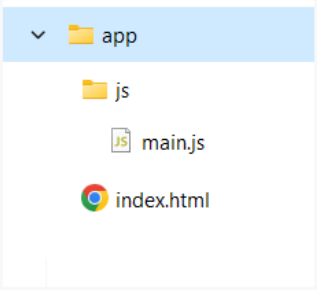
Подключение внешнего файла JavaScript

В прошлой статье код javascript непосредственно определялся на веб-странице. Но есть еще один способ подключения кода JavaScript, который представляет вынесение кода во внешние файлы и их подключение с помощью тега `<script>`

Итак, в прошлой теме мы создали html-страницу **index.html**, которая находится в каталоге **app**. Теперь создадим в этом каталоге новый подкаталог. Назовем его **js**. Он будет предназначен для хранения файлов с кодом javascript. В этом подкаталоге создадим новый текстовый файл, который назовем **main.js**. Файлы с кодом javascript имеют расширение **.js**. То есть у нас получится следующая структура проекта в папке **app**:



Откроем файл **main.js** в текстовом редакторе и определим в нем следующий код:

```
document.write("<h2>Первая программа на JavaScript</h2>"); // выводим заголовок
document.write("Привет мир!");                             // выводим обычный текст
```

Здесь для добавления на веб-страницу некоторого содержимого применяется метод `document.write`. Первый вызов метода `document.write` выводит заголовок `<h2>`, а второй вызов - обычный текст.

Для совместимости с кодировкой страницы index.html для файла с кодом javascript также желательно устанавливать кодировку utf-8. При работе в Visual Studio Code этот редактор автоматически устанавливает кодировку UTF-8.

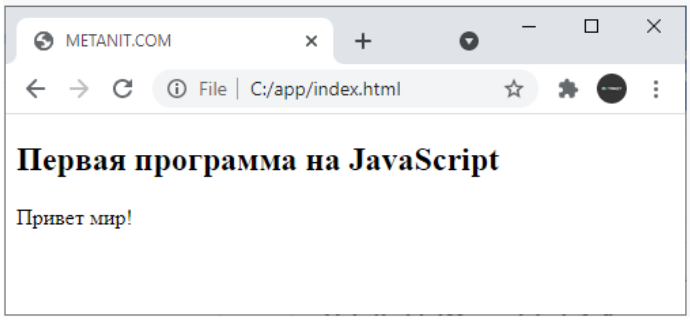
Теперь подключим этот файл на веб-страницу **index.html**:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>METANIT.COM</title>
</head>
<body>
    <script src="js/main.js"></script>
</body>
</html>
```

Чтобы подключить файл с кодом javascript на веб-страницу, применяется также тег **<script>**, у которого устанавливается атрибут **src**. Этот атрибу указывает на путь к файлу скрипта. В нашем случае используется относительный путь. Так как веб-страница находится в одной папке с каталогом js, то в качестве пути мы можем написать **js/main.js**.

Также надо учитывать, что за открывающим тегом script должен обязательно следовать закрывающий `</script>`

И после открытия файла index.html в браузере отобразится сообщение:



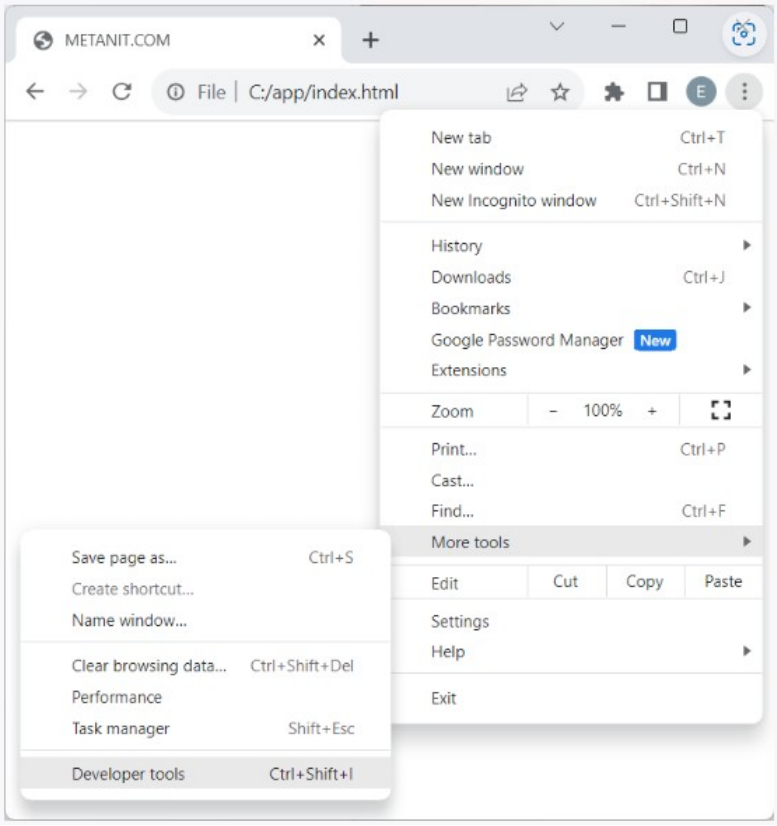
В отличие от определения кода javascript вынесение его во внешние файлы имеет ряд преимуществ:

- Мы можем повторно использовать один и тот же код на нескольких веб-страницах
- Внешние файлы javascript бразуер может кэшировать, за счет этого при следующем обращении к странице браузер снижает нагрузка на сервер, а браузеру надо загрузить меньший объем информации
- Код веб-страницы становится "чище". Он содержит только html-разметку, а код поведения хранится во внешних файлах. В итоге можно отделить работу по созданию кода html-страницы от написания кода javascript

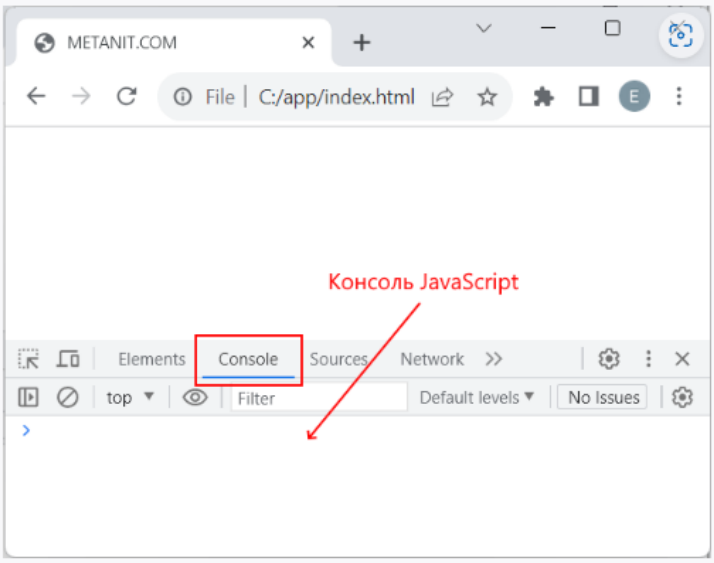
Поэтому, как правило, предпочтительнее использовать код javascript во внешних файлах, а не в прямых вставках на веб-страницу с помощью элемента script.

Консоль браузера и console.log

Незаменимым инструментом при работе с JavaScript является консоль браузера, которая позволяет производить отладку программы. Во многих современных браузерах есть подобная консоль. Например, чтобы открыть консоль в Google Chrome, нам надо перейти в меню **Дополнительные инструменты (More Tools) -> Инструменты разработчика (Developer tools)**:



После этого внизу браузера откроется консоль:

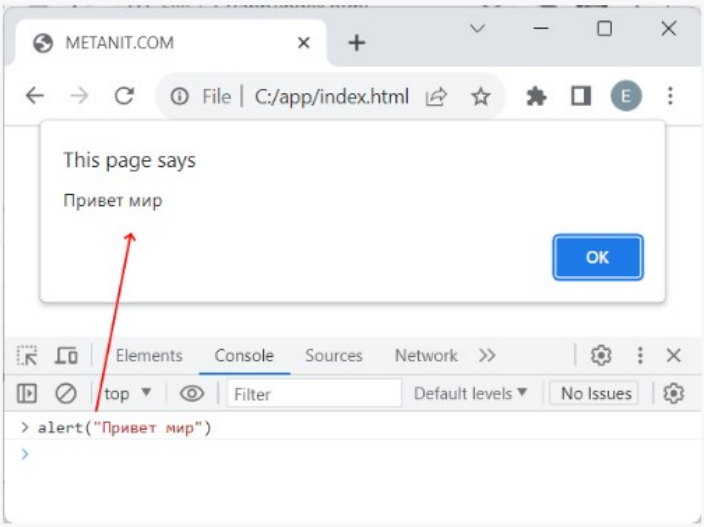


Мы можем напрямую вводить в консоль браузера выражения JavaScript, и они будут выполняться. Например, введем в консоли следующий текст:

```
1 alert("Привет мир");
```

И что замечательно при работе с консолью, при вводе она предлагает всплывающую подсказку с названиями, которые мы хотим ввести, что упрощает ввод, снижает количество возможных ошибок:

Функция **alert()** выводит в браузере окно с сообщением. В итоге после ввода этой команды и нажатия на клавишу Enter браузер выполнит эту функцию и отобразит нам окно с сообщением:



Таким образом, для написания и выполнения кода JavaScript нам в принципе не нужен ни текстовый редактор, ни файл веб-страницы, который содержит код javascript, достаточно одной консоли браузера. Однако набирать большой и сложный код JavaScript в консоли может быть не очень удобно, поэтому в дальнейшем для всех примеров по прежнему будет использовать код JavaScript, который встраивается на html-страницу.

Для вывода различного рода информации в консоли браузера используется специальная функция **console.log()**. Например, определим html-страницу со следующим кодом:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" >
    <title>METANIT.COM</title>
</head>
<body>
    <script>
        const sum = 5 + 8;
        console.log("Результат операции: ");
        console.log(sum);
    </script>
</body>
</html>
```

В коде javascript с помощью ключевого слова **const** объявляется константа `sum`, которой присваивается сумма двух чисел 5 и 8:

```
const sum = 5 + 8;
```

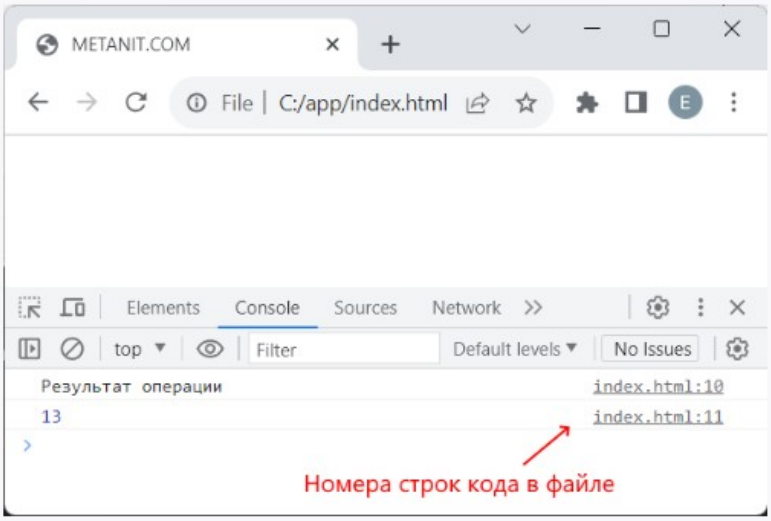
Далее с помощью метода `console.log()` на консоль браузера выводится некоторое сообщение

```
console.log("Результат операции");
```

И в конце также с помощью метода `console.log()` на консоль браузера выводится значение константы sum.

```
console.log(sum);
```

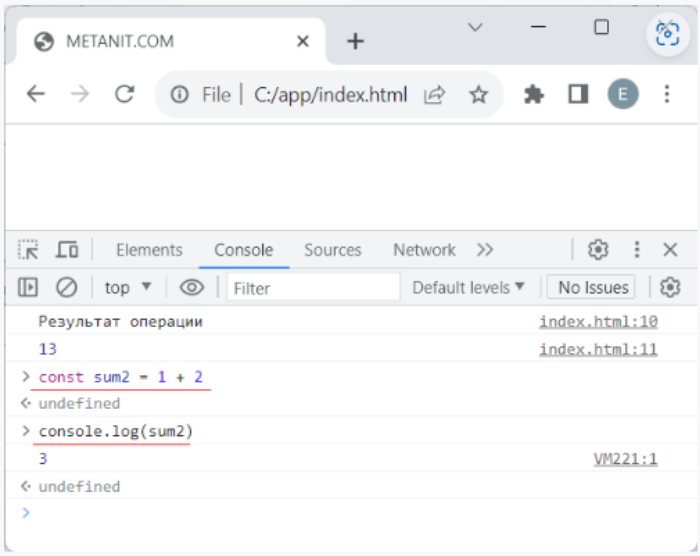
И после запуска веб-страницы в браузере мы увидим в консоли результат выполнения кода:



Что очень полезно, в консоли браузера вы также можете заметить номера строк кода, где именно выполнялся вывод на консоль.

В дальнейшем мы часто будем обращаться к консоли и использовать функцию console.log.

Причем подобный код мы могли бы ввести в самой консоли:



Также последовательно вводим инструкции и после ввода каждой инструкции нажимаем на Enter. В данном случае я ввел две инструкции:

```
const sum2 = 1 + 2      // определяем константу sum2, которая равна сумме 1 + 2
console.log(sum2)       // выводим значение константы sum2 на консоль
```

Если нам надо, чтобы код в консоли переносился на новую строку без выполнения, то в конце выражения javascript нажимаем на комбинацию клавиш **Shift + Enter**. После ввода последней инструкции для выполнения введенного кода javascript нажимаем на Enter.