Lecture 1

Contents: Introduction; Editors for writing HTML program; Installing `notepad++` editor; Writing the first program; Adding numbers; Features of JavaScript; Current version of JavaScript; Exercises

1.1 Introduction

JavaScript is a very popular programming language of the *World Wide Web*, and it is easy to learn. HTML is used to define the content of web pages. The purpose of CSS is to specify the layout of web pages. But, JavaScript is used to program the behaviour of web pages.

It is not required to download JavaScript. JavaScript is already running in the browser on the computer, tablet, and smart-phone. It is free to use for everyone. JavaScript is to be placed within an HTML program.

Note: A script is a series of instructions that a computer can follow to achieve a goal.

1.2 Editors for writing HTML program

There are many editors by which an HTML program could be developed. A few names are given as follows:

- Atom
- Notepad ++
- Sublime Text
- Visual Studio Code
- Adobe Dreamweaver CC
- Froala
- CoffeeCup

We shall take `notepad++` as an editor for coding HTML programs. `notepad` also serves as an editor.

`notepad++` is a free editor and it supports several languages. Running in the MS Windows environment, its use is governed by GNU General Public License.

Based on the powerful editing component `Scintilla`, `notepad++` is written in C++ and uses pure Win32 API and STL which ensures a higher execution speed and smaller program size.

NOTE: GNU stands for GNU's Not Unix; API has full form application programming interface; STL is a file format native to the stereolithography CAD software created by 3D Systems. STL has several acronyms such as *standard triangle language* and *standard tessellation language*.

1.3 Installing `notepad++` editor

It is a free software. Visit website https://notepad-plus-plus.org/downloads/ to download `notepad++` software. Otherwise, you can give a search using "download notepad++" using Google search. Follow the instructions to install the software and choose option to create a shortcut at the desktop.

1.4 Writing the first program

Every HTML program has extension `.html`. One can execute an HTML program in different ways.

⊳ Firstly, one can just double click the icon of the file containing the HTML program.

⊳ One could write or copy the entire path of the file containing HTML program in the address bar of a browser and then press the "Enter" key of the keyboard. Then the program gets executed.

HTML programs are tag-based. Here, we include JavaScript in the HTML program. An example program is given in Fig. 1.1.

```
<html>
<body>
<h3>First JavaScript Program</h3>
<button type="button"
onclick="document.getElementById('demo').innerHTML = Date()">
Click here to display date and time.</button>
<p id = "demo"></p>
</body>
</html>
```

Figure 1.1: An HTML program that uses JavaScript to display date and time

The output (screenshot) of the program is given in Fig. 1.2.

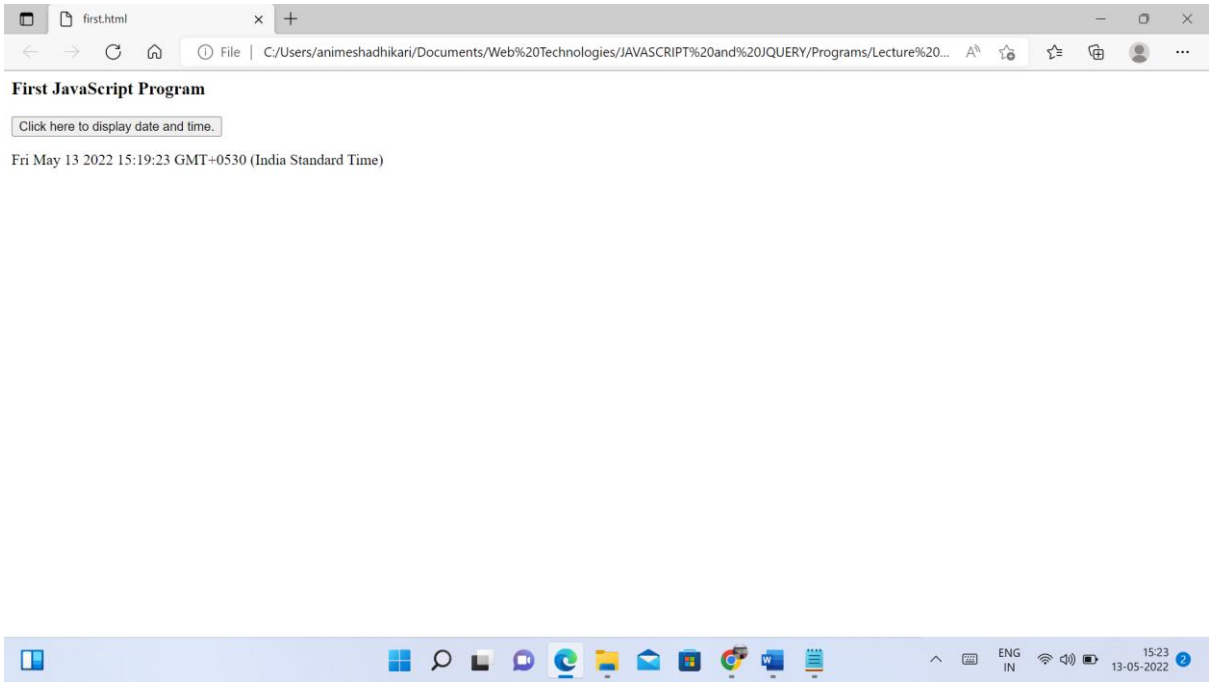

Figure 1.2: The output of program given in Fig. 1.1

1.5 Adding numbers

Suppose we wish to add three numbers. Fig. 1.3 illustrates how to add three numbers using JavaScript.

```
<html>
<body>
<h2>Add Numbers</h2>
<p>Paragraph for addition of numbers</p>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML = 5 + 6 + 7;
</script>
</body>
</html>
```

Figure 1.3: Addition of three numbers using JavaScript

The output (screenshot) of the program is given in Fig. 1.4.

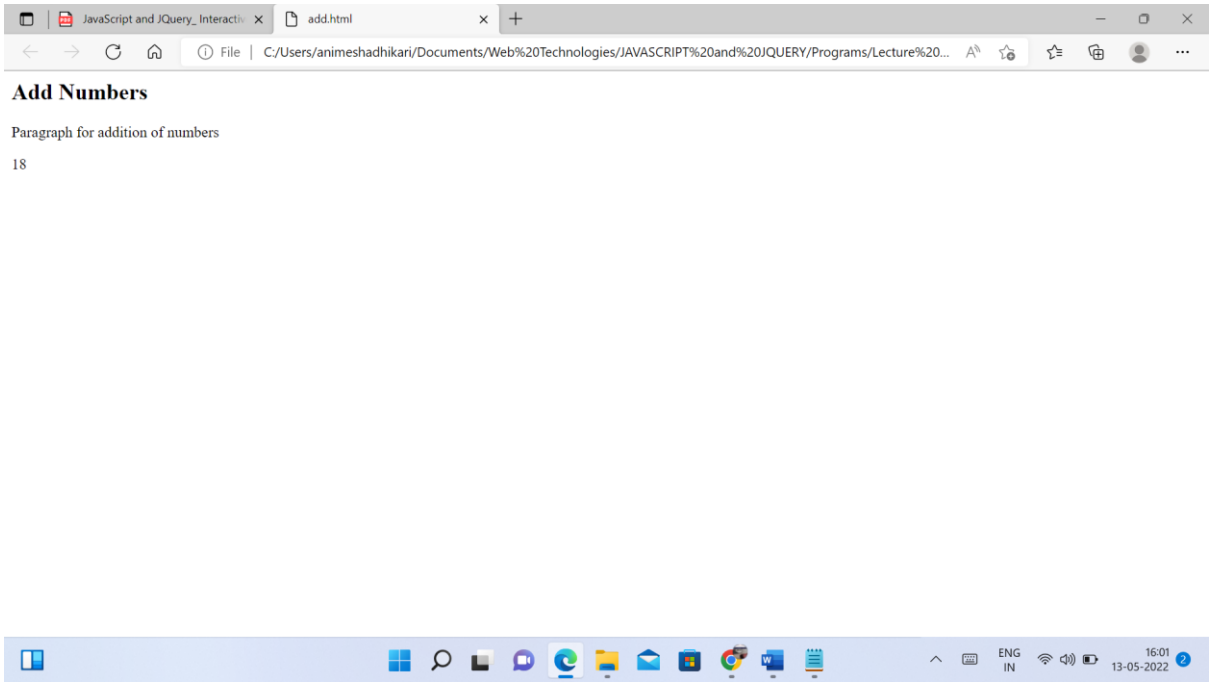

Figure 1.4: The output of program given in Fig. 1.3

1.6 Features of JavaScript

JavaScript is a client edge script language. This is a object-centered script language which is commonly used for designing web pages. Features of JavaScript include the following.

▻ *Object-centered script language*

Object centered languages has a feature of built-in objects such as a `window` object. A few examples of object centered languages are Java Script and Visual Basic. The object-centered languages mostly use features, like polymorphism, i.e., an ability of taking an object in many forms.

Lecture 2

Contents: Common uses; Change of text; Change of style; Change of attribute; Hiding elements; Exercises

2.1 Common uses

Common uses of JavaScript are image manipulation, form validation, and dynamic changes of content. In many cases, to select an HTML element, JavaScript often uses the `document.getElementById()` method. See a program in Fig. 2.1.

```
<html>
<body>
<h2 style = "color: DarkSlateBlue";>Use JavaScript to change text</h2>
<p>In this example, we write the following bold text in &lt;p&gt; element.</p>
<p id="p1"></p>
<script>
document.getElementById("p1").innerHTML = "<b>JavaScript is interesting.</b>";
</script>
</body>
</html>
```

Figure 2.1: HTML program to write text in an element using JavaScript

The output (screenshot) of the program is given in Fig. 2.2.

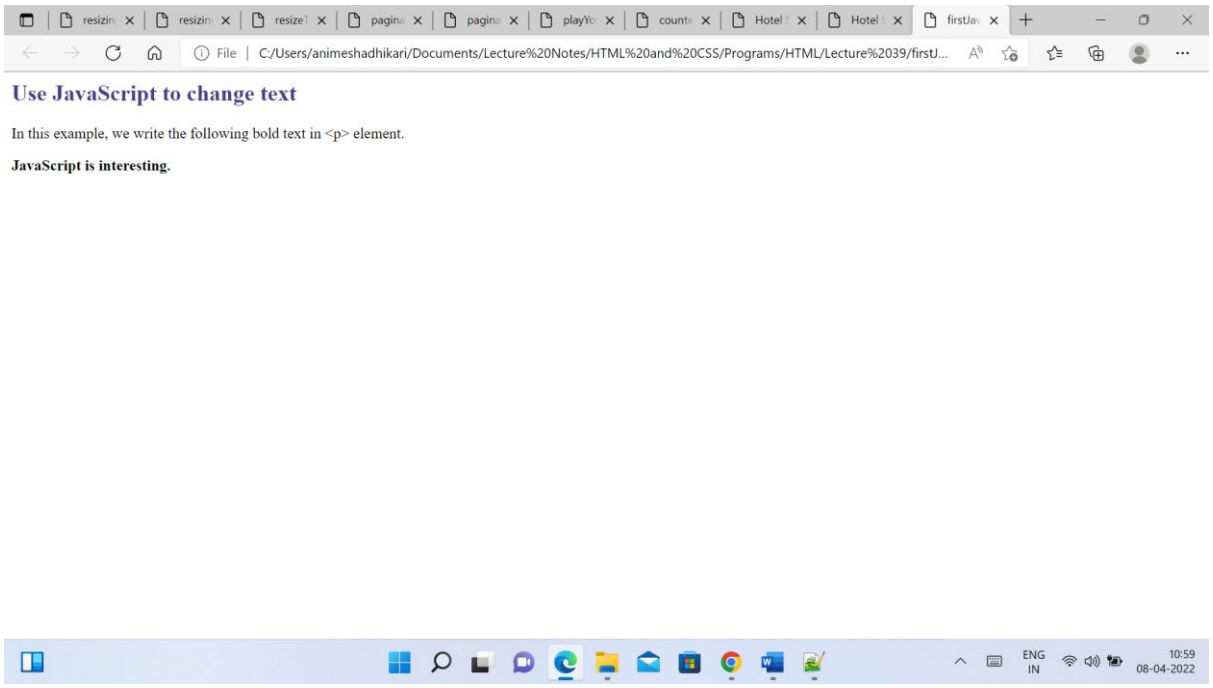

Figure 2.2: The output of the program given in Fig. 2.1

2.2 Change of text

Using JavaScript, one can change the text in an element. See Fig. 2.3.

```
<html>
<body>
<h2 style = "color: DarkSalmon";>Change text by JavaScript</h2>
<p>JavaScript can change the content of an HTML element:</p>
<button type="button" onclick="display()">Change text</button>
<p id="p1">This is a line of text.</p>
<script>
function display () {
  document.getElementById("p1").innerHTML = "<b>JavaScript is interesting.</b>";
}
</script>
</body>
</html>
```

Figure 2.3: HTML program to change text in a paragraph using JavaScript

The output (screenshot) of the program is given in Fig. 2.4.

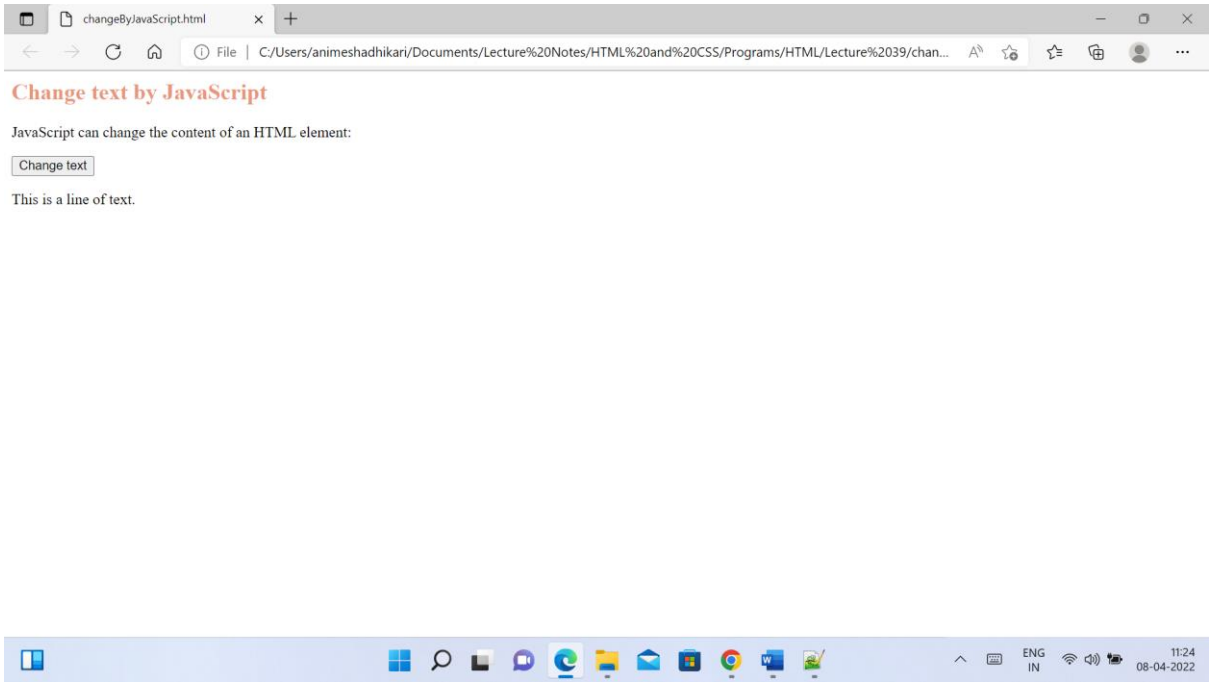

Figure 2.4: The output of the program given in Fig. 2.3

2.3 Change of style

One can change style of an element by JavaScript. See a program in Fig. 2.5. The output (screenshot) of the program is given in Fig. 2.5.

```
<html>
<body>
<h2 style = "color:DarkSeaGreen";>Change style by JavaScript</h2>
<p id="p1">JavaScript can change the style of an HTML element.</p>
<script>
function changeStyle() {
  document.getElementById("p1").style.fontSize = "20px";
  document.getElementById("p1").style.color = "DarkSalmon";
  document.getElementById("p1").style.backgroundColor = "Gainsboro";
}
</script>
<button type="button" onclick="changeStyle()">Change style</button>
</body>
</html>
```

Figure 2.5: Program to style HTML element by JavaScript

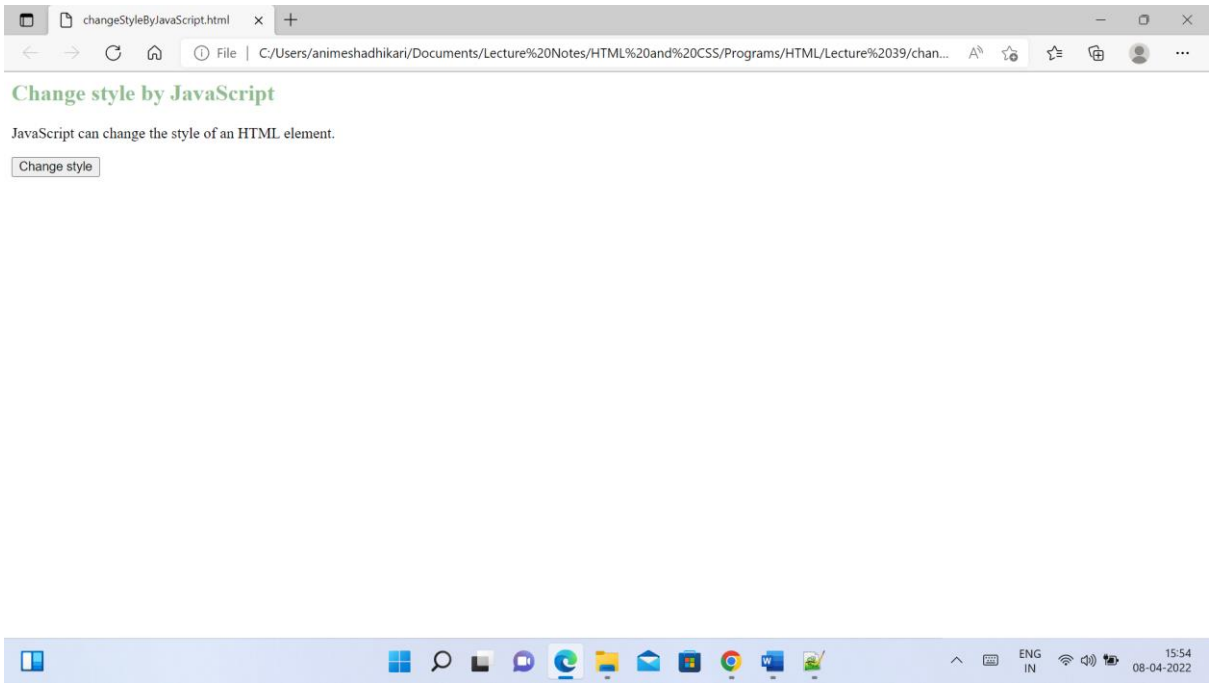

Figure 2.6: The output of program given in Fig. 2.5 (before click)

2.4 Change of attribute

Using JavaScript, one can change the attribute of an element. See the program in Fig. 2.7. The output (screenshot) of the program is given in Fig. 2.8.

Lecture 3

Contents: JavaScript in <head> element; JavaScript in <body> element; JavaScript in external file; Output statements; Exercises

3.1 JavaScript in <head> element

A JavaScript function is a block of JavaScript code, that can be placed in the <head> element, and can be called for execution. A program is given in Fig. 3.1.

```
<html>
<head>
<script>
function f123() {
  document.getElementById("p3").innerHTML = "Good evening";
}
</script>
</head>
<body>
<h2>JavaScript in Head</h2>
<p id="p3">Good morning</p>
<button type="button" onclick="f123()">Execute it</button>
</body>
</html>
```

Figure 3.1: HTML program having JavaScript in <head> element

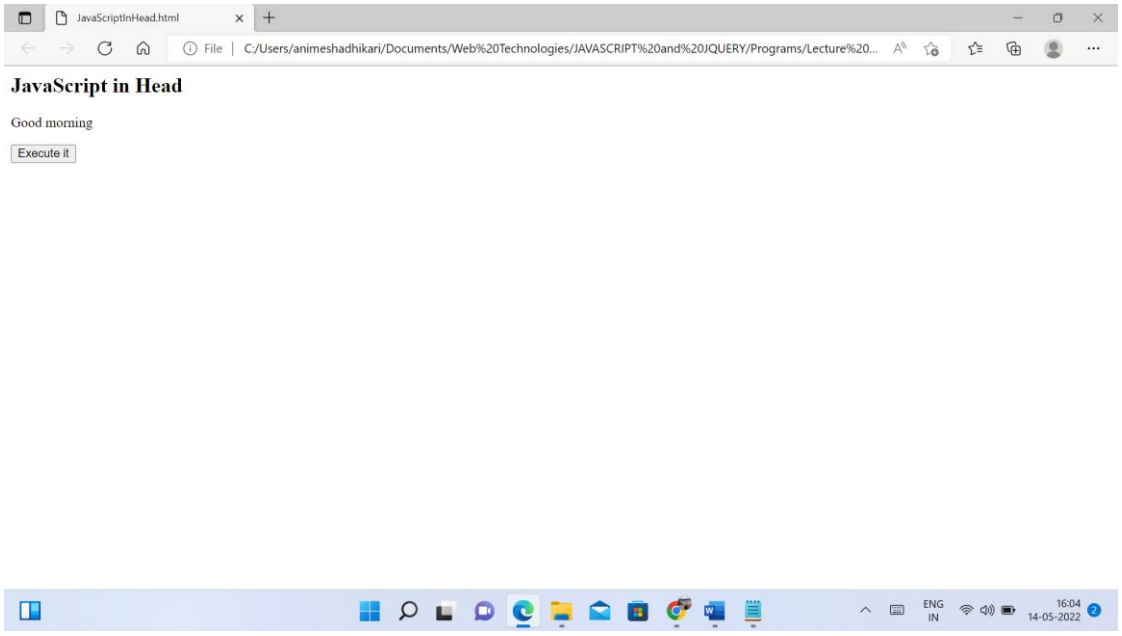

Figure 3.2: The output of program given in Fig. 3.1

An example of event is a mouse click. Function `f123()` is called, when the mouse is clicked on the button. The output (screenshot) of the program is shown in Fig. 3.2.

3.2 JavaScript in <body> element

JavaScript can be placed in the `<body>` element. See the program in Fig. 3.3.

```
<html>
<body>
<h2>JavaScript in Body</h2>
<p id="p3">Good morning</p>
<button type="button" onclick="f123()">Execute it</button>
<script>
function f123() {
  document.getElementById("p3").innerHTML = "Good evening";
}
</script>
</body>
</html>
```

Figure 3.3: HTML program containing JavaScript in `<body>` element

The output (screenshot) of the program is shown in Fig. 3.4.

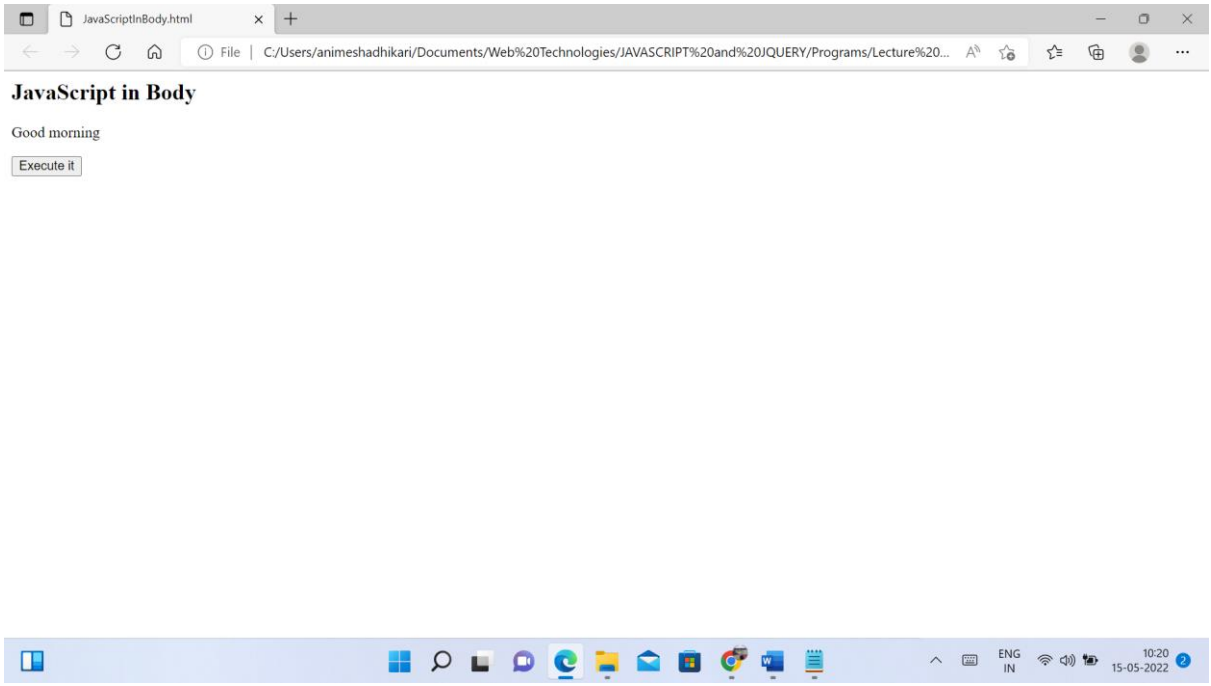

Figure 3.4: The output of program given in Fig. 3.3

3.3 JavaScript in external file

JavaScript can also be placed in external files. JavaScript files have the file extension `.js`. In this example, we place function `f123()` in the external file, named as `ExternScript.js`.

```
<html>
<body>
<h2>JavaScript in External File</h2>
<p id="p3">Good morning</p>
<button type="button" onclick="f123()">Execute it</button>
<p>This example links to "ExternScript.js"</p>
<script src="ExternScript.js"></script>
</body>
</html>
```

Figure 3.5: HTML program containing JavaScript in external file

The content of external file `ExternScript.js` is given in Fig. 3.6.

```
function f123() {
  document.getElementById("p3").innerHTML = "Good evening";
}
```

Figure 3.6: Program `ExternScript.js`

The output of program is shown in Fig. 3.7.

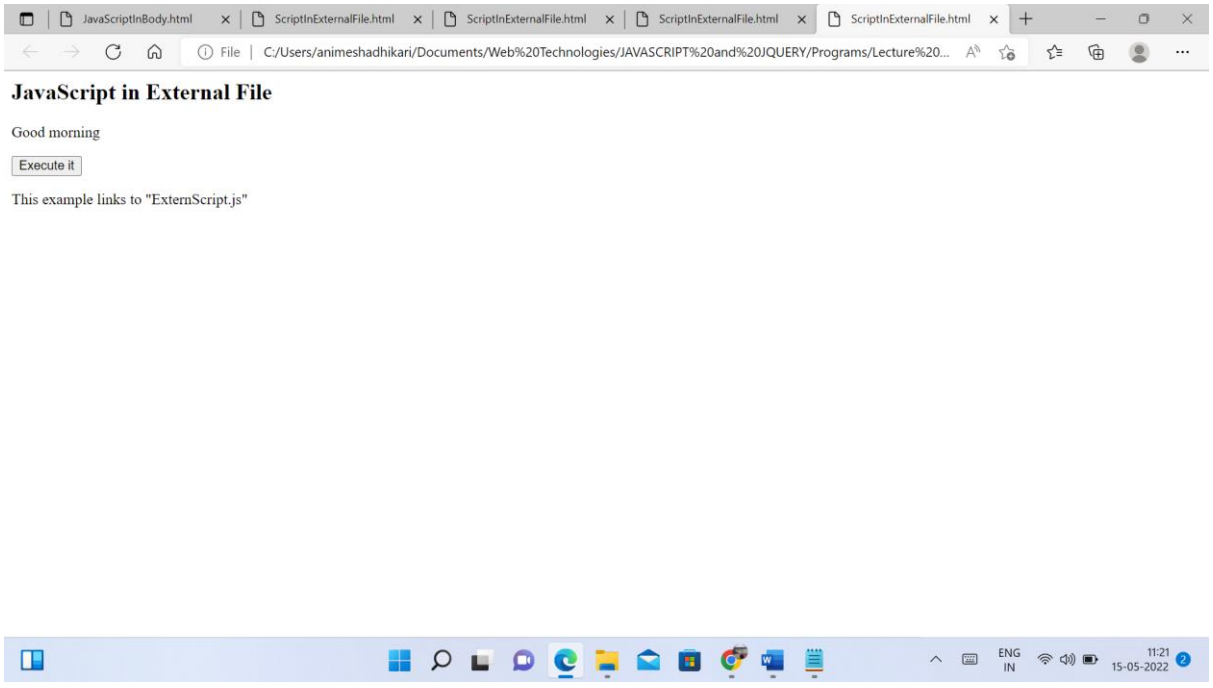

Figure 3.7: The output of program given in Fig. 3.5

We call function `f123()` is placed in file `ExternScript.js`. This external file is attached using `src` attribute. See Fig. 3.5. It is possible to call multiple external files. In otherwords, an HTML file can call many functions from multiple `.js` files. Suppose that an HTML file calls functions from three `.js` files. Then three `.js` files are to be hooked with the main `.html` file.