

Вовед

Интернет програмирање

Иван Китановски

Бојан Илијоски

Зошто JavaScript?

- JavaScript е еден од **3 технологии** кои сите веб програмери **мора** да ги знаат:
 - **HTML** за дефинирање на содржината
 - **CSS** за специфирање на стилот
 - **JavaScript** за да се програмира однесувањето на страницата

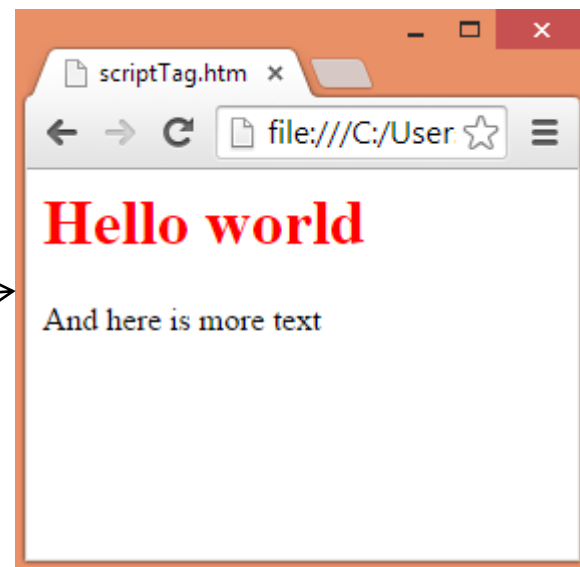
Каде се поставува JavaScript?

- JavaScript кодот се вметнува во рамките на `<script>` тагот:

```
<script type="text/javascript">
//JavaScript kod
</script>
```

- Пример

```
<html>
<head>
  <script type = "text/javascript">
    <!--
      document.write("<h1 style=\"color: red\">");
      document.write("Hello world");
      document.write("</h1>");
    // -->
  </script>
</head>
<body> <p>And here is more text</p> </body>
</html>
```



Во сите понови прелистувачи и HTML5 нема потреба да се наведува бидејќи предефиниран скриптен јазик е JavaScript



Работа со прелистувачи што не поддржуваат скриптирачки јазици

- Некои стари пребарувачи не ги препознаваат script таговите
- Овие пребарувачи ќе ги игнорираат script таговите но, ќе го прикажат кодот на вметнатиот JavaScript
- Со цел да се овозможи старите пребарувачи да го игнорираат целиот код, се користат тагови за коментари од HTML што ја кријат скриптата од пребарувачот и тој ја игнорира
- синтакса

```
<!--  
script here  
// -->
```

- <!-- воведува HTML коментар (почеток)
- за JavaScript да ја игнорира ознаката за крај на HTML коментарот (-->), се користи JavaScript коментар (//), кој важи до крајот на линијата



Пример

```
<html >
<head>
  <title>My first script</title>
</head>
<body bgcolor="#FFFFFF">
<h1>
  <script language="Javascript" type="text/javascript">
    <!-- Hide script from old browsers

        document.write("Hello, world!")

  // End hiding script from old browsers -->
  </script>
</h1>
</body>
</html>
```



Каде се вметнува JavaScript

- Може да се вметне во
 - ☐ заглавјето (<head>)
 - ☐ телото (<body>) на еден HTML документ
 - ☐ и на двете места
- Функциите треба да бидат дефинирани во заглавјето (<head>)
 - ☐ тоа обезбедува функцијата да биде наполнета пред да биде употребена



Надворешен JavaScript

- Може да се смести во одделна .js датотека
`<script src="myJavaScriptFile.js"></script>`
 - надворешната .js датотека овозможува користење на иста скрипта на повеќе HTML страници
 - надворешната .js датотека не може да содржи `<script>` таг
- Пример:
`<script src="myjavascript.js"
 language="JavaScript1.2"
 type="text/javascript">
</script>`

Пишување во конзола

- Ако прелистувачот поддржува дебагирање

- `console.log()`

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<script>
    console.log('Hello');
</script>
</body>
</html>
```

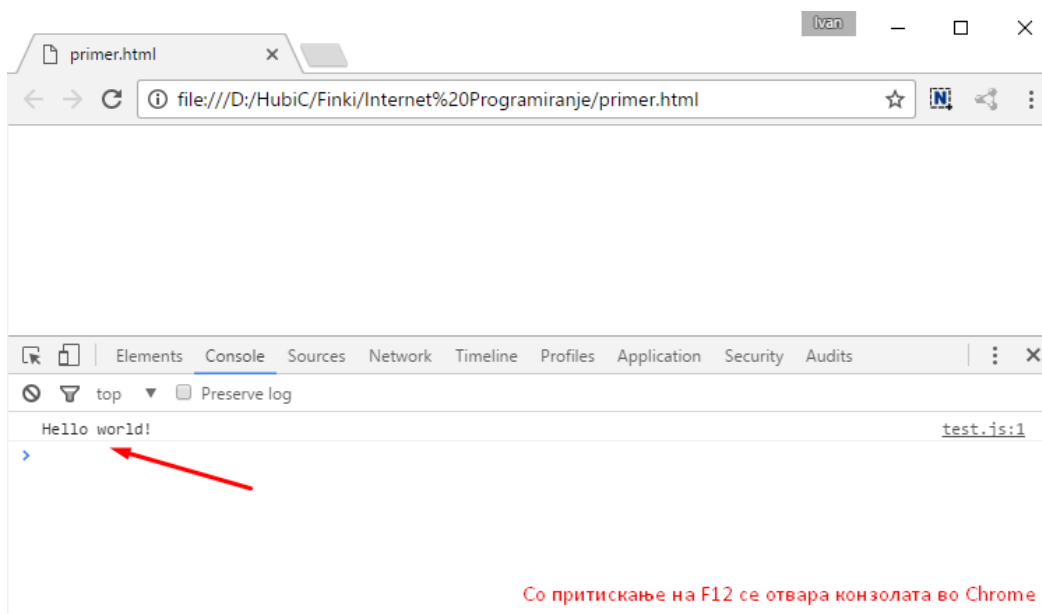

Пример: Надворешен JS

primer.html

```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript" src="test.js"></script>
</head>
<body>
</body>
</html>
```

test.js

```
console.log("Hello world!");
```

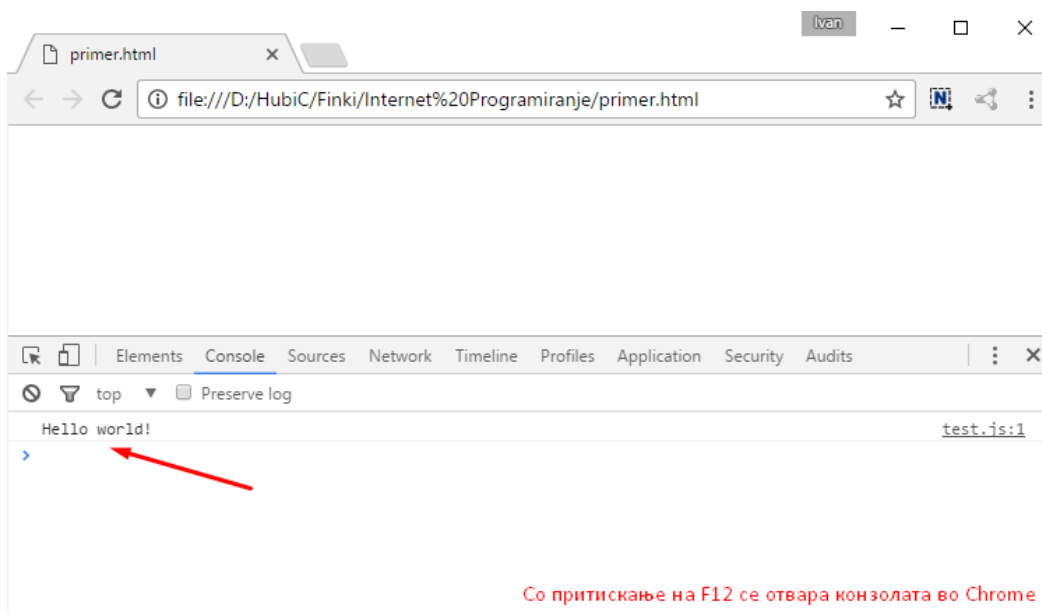


Со притискање на F12 се отвара конзолата во Chrome

Пример: Директно во script тагот

primer.html

```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
        console.log("Hello world!");
    </script>
</head>
<body>
</body>
</html>
```



Со притискање на F12 се отвара конзолата во Chrome

Променливи

- Променливите се декларираат со наредбата `var`
 - зборот `var` е опционален (добар програмерски стил е негова употреба)
 - податочниот вид на променливите не мора да биде деклариран и се определува во време на извршување на скриптата
 - променливите може да содржат вредност од кој и да е вид
 - `Var pi;`
 - `Var ime: Integer;` `//(Javascript 2.0)`
- Променливите може да се иницијализираат со користење на знакот `=`
 - `var pi = 3.1416, x, y, name = "Dr. Dave" ;`
 - имињата на променливите мора да започнуваат со буква, долна цртичка (`_`) или знакот `$`
 - се прави разлика меѓу големи и мали букви во имињата на променливите

Едноставни податочни видови

- JavaScript има три „едноставни“ податочни видови:
 - нумерички вредности (number),
 - текстуални низи (string), и
 - логички вредности (boolean)
 - се останато е објект
- Нумеричките вредности секогаш се сместуваат во формат со подвижна точка
 - хексадецималните броеви започнуваат со 0x
 - некои платформи бројот 0123 го сметаат како октален, другите го сметаат како децимален



Едноставни податочни видови

- Текстуалните низи може да се ограничени во единечни или двојни наводници
 - тие може да содржат и контролни знаци - `\n` (newline), `\"` (double quote), итн.
 - Пример:

```
strFirst = "John";  
strLast = "Kennedy";  
strFull = strFirst + "F." + strLast;
```
- Логичките податоци може да имаат вредност `true` или `false`
 - `0`, `"0"`, празни текстуални низи, `undefined`, `null`, и `NaN` се логички `false`
 - сите останати вредности се логички `true`

Пример

primer.html

```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
        var pi = 3.14;
        var person = "John Doe";
        var answer = 'Yes I am!';
    </script>
</head>
<body>
</body>
</html>
```

Други податочни видови

■ Сложени

- ☐ Object

- ☐ Array

■ Специјални

- ☐ Null

- ☐ Undefined

Пример

primer.html

```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
        var this_is_empty = null;
        var this_is_undefined;
        // ili this_is_undefined = undefined;
    </script>
</head>
<body>
</body>
</html>
```


Оператори

■ операторите се користат за процесирање на вредностите

■ видови оператори

☐ аритметички оператори:

+ - * / % ++ --

☐ релациски оператори:

< <= == != >= >

☐ логички оператори:

&& || !

☐ битови оператори:

& | ^ ~ << >> >>>

☐ оператори за доделување:

+= -= *= /= %= <<= >>=

>>>= &= ^= |=

☐ оператори за текстуални низи:

+

■ условен оператор:

condition ? value_if_true : value_if_false

■ Дополнителни оператори:

new typeof void delete

Оператори

- Специјални релациски операции за тестирање еднаквост:
 - `==` и `!=` се обидуваат да ги конвертираат нивните операнди во ист вид пред да го извршат тестот
 - `===` и `!==` претпоставуваат дека нивните операнди се нееднакви ако истите се од различен вид

Оператори

- Системот ќе се обиде да ги кастира вредностите да може да ја изврши операцијата
 - се може да се претвори во текстуална низа
 - некои текстуални низи може да се претворат во броеви
 - дополнување за логичките вредности
 - во нумерички контекст, **true** се кастира во 1 и **false** се кастира во 0
 - во логички контекст, дефинираните вредности се кастираат во **true**, а недефинираните (**undefined**) се кастираат во **false**
 - во контекст на текстуални низи, **true** се кастира во “true” и **false** се кастира во “false”
 - ништо освен функција не може да се кастира во функција
- Операторите се извршуваат во контекст на операндите
 - $a * b$ \Rightarrow обезбедува нумерички контекст
 - $e(x)$ \Rightarrow контекст на функција (за e)
 - $a + b$ \Rightarrow неопределено (текстуална низа или број)

Пример

primer.html

```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
        var x = 5;           // assign the value 5 to x
        var y = 2;           // assign the value 2 to y
        var z = x + y;       // assign the value 7 to z (x + y)
    </script>
</head>
<body>
</body>
</html>
```

Пример

primer.html

```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
        var x = 10;
        x += 5
    </script>
</head>
<body>
</body>
</html>
```

Пример

primer.html

```
<!DOCTYPE html>
<html>
<head>
  <script type="text/javascript">
    txt1 = "John";
    txt2 = "Doe";
    txt3 = txt1 + " " + txt2;
    console.log(txt3);
  </script>
</head>
<body>
</body>
</html>
```

```
> txt1 = "John";
< "John"
> txt2 = "Doe";
< "Doe"
> txt3 = txt1 + " " + txt2;
< "John Doe"
>
```

Пример

primer.html

```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
        txt1 = "What a very ";
        txt1 += "nice day";
    </script>
</head>
<body>
</body>
</html>
```

```
> txt1 = "What a very ";
< "What a very "


---


> txt1 += "nice day";
< "What a very nice day"


---


> |
```

Пример

primer.html

```
<!DOCTYPE html>
<html>
<head>
  <script type="text/javascript">
    x = 5 + 5;
    console.log(x);
    y = "5" + 5;
    console.log(y);
    z = "Hello" + 5;
    console.log(z);
  </script>
</head>
<body>
</body>
</html>
```

10

55

Hello5

> |

Пример

primer.html

```
<!DOCTYPE html>
<html>
<head>
  <script type="text/javascript">
    a = 5 == 5;
    console.log(a);
    b = "5" == 5;
    console.log(b);
    c = "5" === 5;
    console.log(c);
    d = "5" === "5";
    console.log(d);
  </script>
</head>
<body>
</body>
</html>
```

true

true

false

true

> |

Наредби (1)

- Наредба за доделување:
`greeting = "Hello, " + name;`
`nNum -= 3; nNum = nNum - 3;`
`nNum *= 3; nNum = nNum * 3;`
`nNum /= 3; nNum = nNum / 3;`
`nNum %= 3; nNum = nNum % 3;`
- Блок од наредби:
`{ statement; ...; statement }`
- Празна наредба: `;;` или `{ }`

Наредби (2)

■ Наредби за избор

- If, switch

■ Наредби за повторување

- for, do-while, while. for-in

- Пример:

```
var person = {fname:"John", lname:"Doe", age:25};  
var text = "";  
var x;  
for (x in person) {  
    text += person[x];  
}
```

■ break, continue

Typeof / instanceof

■ Typeof

- Го враќа типот на променливата

```
> temp = 5; // temp is a number  
↵ 5  
-----  
> typeof(temp)  
↵ "number"-----
```

■ Instanceof

- Враќа true само ако променливата е од дадениот тип на објект

Коментари

- Коментарите се идентични како и коментарите во С или во Java:
 - од // до крајот на линијата
 - меѓу /* и */

- Пример

```
<script language="JavaScript">  
<!-- definition of variables  
var num_car= 25;  
var passenger_per_car= 3;  
//calculation of total number of people  
var total_passenger= num_car * passenger_per_car  
Alert(total_passenger);  
// end of script -->  
</script>
```