

HTML

04.06.2023.

(Hipertensta iestīmēšanas valoda)
(HyperText Markup Language)

- Tā ir iestīmēšanas valoda, kas ir iestrādāta tīmekļa lapušu un citas pārlūnprogrammas attēlojāmās informācijas glabāšanai.
- Pēdējā HTML versija ir **HTML 5**.
- HTML pirmo reizi iaveidoja **Jims Berners-Li, Roberts Kailijs**, u.c. sānot no 1989.g.
- HTML iamanta:
 - tagus;
 - atribūtus.

HTML tagi

• Tagus iamanta, lai atāmētu HTML elementa sākumu, un tie parasti tiek ievietoti lenķveida ieuavās (< ; >)

Kas

• Piemēram: `<h1>`

HTML struktūri

• Lielākā daļa tagu ir jāatver `<h1>` un jāaizver `</h1>`, lai tie varētu darboties.

• Atribūtos ir ietverta papildu informācija.

• Piemēram: ``

Šajā gadījumā attēla avots (`src`) un alternatīvais teiksts (`alt`) ir taga `` atribūti.

<!DOCTYPE html> - norāda, kurā valodā
tiks raustīts šajā lapā. Šajā gadījumā valoda ir HTML5.

<html> - norāda, ka turpmāk mēs raustīsim
HTML vodā.

<head> - šeit tiek ievietoti visi lapas metadati -
satura, kas galvenokārt paredzēts menlētājprogrammām
u.c. datorprog.

<body> - lapas saturs, no kura redāt cilvēks.

<title> - lapas nosaukums, kas būs redzams
pārlūpprogrammas loga vai cilnes augšdaļā.

Nosaukums X

<meta> - šeit tiek glabāta informācija par
dokumentu: raustaīju vedejums, nosaukums,
aprausts.

Piemērs: **<head>**

```
<title>My first Webpage</title>
<meta charset="UTF-8">
<meta name="description" content="This
field contains info about your page. It is
usually around 2 sentences long.">
<meta name="author" content="Dogs">
</head>
```

meta

<body> - ietver testu, attēlus, tabulas,
veidlapas un visu pārejo, no mēs varam iegūt
redzam interneta.

<h1>Text</h1>

Virsrauti: **<h1>** } galvenie virsrauti.

<h2>

<h3>

<h4>

<h5>

<h6>

<u> - ianeido jaunu zinduopu, kuri var pievienot teistu.

**** - Bold (treinraists). Iesūmē svāīgāko teista daļu. Jācēl teistu.

**** - Strong (treinraists). Jācēl galveno teistu, tāpat uā Bold funkcija.

<i> Italic (slīpraists).

**** Emphasised Text (stipraists). Parasti izmanto uā attēlu apraistu. Iesūtās tāpat uā **<i>** kurstus

<mark> Marked Text (iemāsots teists) (daeltenā veāsā).

<small> Small Text (mazs teists). Samazina ~~teists~~ fontai iamēru par 1.

<strike> Striked Out Text (nosvītrots teists) Tiek iannantots ar **<s>**.

<u> Underlined Text (pasvītro teistu).

<ins> Inserted Text (ievietots teists) Tiek attēlots ar pasvītrojumu, lai parādītu ievietoto teistu. Iesūtās tāpat uā **<u>**.

<sub> Subscript Text (apaušrausta teists) maxi burti. Iesūtās tāpat uā **<small>**.

<sup> Superscript Text (augšrausta teists) ar maxi burti (**<small>**).

<big> Palielina fontai iamēru par 1.

**
** Jauna zinda

<p></p> Jauna zinduopa

<a> (anchor) (enums)

Teists

- Atribūta pirmā dala norāda uz lapu, kas tiks atvērta pēc nolīmēšanas uz saites.

- Tumēz atrībūta otrajā dala ir teists, kas tiks parādīts apmeklētājam, lai mudinātu viņu nolīmēt uz šīs saites.

linus

Teists

 - satur atrībūtu. Ietver info par attēla avotu, augstumu, platumu un alternatīvo teistu.

- Tiešsaistē attēlu failiem parasti tiek izmantoti šādi failu veidi: •jpg, •png un •gif.

↑
42

↑
42

- Tēma diadinās 3 dažādi sarakstu veidi.

Sākārtotais saraksts (Numurēts saraksts)

 - sākārtots satura saraksts

- Tagā mēs ievietām naturi saraksta vienumu tagā,

Piemēram:

Kods:

```
<ol>
<li> Objets </li>
<li> Objets </li>
<li> Objets </li>
</ol>
```

Output:

1. Objets
2. Objets
3. Objets

- Tien pielinti vārti numuri jeb teustoši tieši numurēts.

Nesakārtots saraksts (nenumurēts/aizāmēti)

- Tas ir labāk parādītams kā aizāmju saraksts, tajā nav snaitlīnijas.

Kods:

```
<ul>
<li> Objets </li>
<li> Objets </li>
<li> Objets </li>
</ul>
```

Output:

- Objets
- Objets
- Objets

Definīciju saraksts. <dl>

```
<dl>
```

```
<dt> Objets </dt>
```

```
<dd> Definīcija </dd>
```

```
</dl>
```

Output:

Objets

Definīcija

Tabulas HTML.

- Lai ēdot tabulu, mums ir jāizmanto elements **<table>**.

- Šajā tagā mēs strukturējam tabulu, izmantojot tabulas rindas **<tr>** un šūnas **<td>**.

```
<table>
  <tr>
    <td> 1. rinda - Kolonna 1 </td>
    <td> 1. rinda - Kolonna 2 </td>
    <td> 1. rinda - Kolonna 3 </td>
  </tr>
  <tr>
    <td> 2. rinda - Kolonna 1 </td>
    <td> 2. rinda - Kolonna 2 </td>
    <td> 2. rinda - Kolonna 3 </td>
  </tr>
</table>
```

<thead> - Table Head (tabulas virzants)

Virzants arzants ar **<th>** palīdzību.

<tbody> - Table Body (Tabulas saturs)

<tfoot> - Table foot (Tabulas apņemtāja)

<colgroup> - Column Group (Sleju grupa)

<th> - Table Header (Tabulas galvene).

Datu šūna tabulas galvenei. (Izceļti burti - treimantā)

<caption> - var pievienot tabulai paraustu.
(virzants)

- Tabulas šūnas var sapludināt izmantojat **colspan** un **rowspan** atribūtus.

<col> ierāso kolonnas.

Fonta parametru iestatīšana.

- **color** - krāsa
- **face** - fonta šūts
- **size** - fonta izmērs (norādīts ar sākotli)

color: krāsas nods, kas norādīts ar RGB krāsu
modeli vai arī ar sāusinātu nosaukumu
angļu valodā (blue).

<pre> **font color="#0000FF" size="5"**
face="Arial"> Teists **</pre>**

Fonta lielums absolūtās vienībās	Fonta lielums relatīvās vienībās
size="1"	size="-2"
size="2"	size="-1"
size="3"	size="0"
size="4"	size="+1"
size="5"	size="+2"
size="6"	size="+3"
size="7"	size="+4"

<pre> Saglabā teista formātējumu, savādāns
font.

<tt> Vienāda platumā fonti, tāpat causta
nā ieprievējo.

<cite> Teista elements tiek uzsudīts par
citātu. (Kā slīpīgums).

<kbd> Imitē raustāmmašinas stilu. Raustās
tāpat nā **<tt>** vai **<pre>**.

<address> Definē teista elementu nā adresi,
kas tiek parādīta mazīvā. Raksta tāpat nā **<cite>**.

CSS (1. daļa)

("Cascading Style Sheet")
stila lapas rāsu vadēšana.

CSS tiek izmantots, lai vienuāršā un ētā veidā kontrolierētu tīmekļa dokumenta stilu.

- Tieka veidota un uzstādīta, izmantojot cīņēju grupu **W3C**. (World Wide Web Consortium).

- ↗ • Tā ugunsgrīda, kas sniedz ieteikumus par to, kā internets darbojas un kā tam vajadsētu attīstīties.

- W3C nevar kontroliēt valodas lantiso ieviešanu. Šo programmatūru veido neatkarīgi uņēmumi un organizācijas.

- **Cascading Style Sheets level 1 (CSS1)** nāca uzlādē no W3C kā ieteikuma 1996. g. decembrī.

- ↘ • Šajā versijā ir apraustīta CSS valoda, kā arī vienuāršs vizuālā formatejuma modelis visiem HTML tagiem.

- **CSS2** kļuva par W3C ieteikumu 1998. g. maijā un balstās uz **CSS1**.

- ↙ • Šī versija papildina atbalstu multivides stila lapām, piem., printeri, suņas ierīces, lejupielādētās fonti, elementu pozicionēšana un tabulas.

- CSS sastāv no stila uārtulām, kuras interpretē pārlūpprogrammā un pēc tam piemēro attiecīgajiem dokumenta elementiem.

- Stila ~~atlasī~~ noteikums sastāv no 3 daļām:

- **Selector (atlasītājs)** - HTML tags, kuras tiks piemērots stils. Tas var būt jebkurs tag, piem., `<h1>` vai `<table>`, utt.

- **Property (rewižts)** - HTML taga atribūta veids. Visi HTML atribūti tiek pārversti CSS rewižtos. Tas var būt urāsainas apmales, utt.

- **Value (vērtība)** - Revižtiem tiek piešķirtas vērtības. Piem., māsu rewižta vērtība var būt sānana vai `#F1F1F1` utt.

Piemērs:

```
selector { property: value }
```

```
table { border: 1px solid #C00; }
```

atlasītājs (tabula) rewižts (apmala) vērtības

1) Piešķir māsu visiem 1. lēmena vieraustiem.

```
h1 {
```

```
color: #36CFFF;
```

```
}
```



2. Universālie atlasītāji (atbilst jebkura elementa tipa nosaukumam).

* {

color: #000000;

}

- c) Katru dokumenta elementa saturu atveido melnā urāsā.

3. Pēcnācēju atlasītāji (Ja vēlas piemērot stila vārdīlu konkrētam elementam tātai tad, ja tas atrodas noteiktā elementā).

- Stila noteikums tiks piemērots `` elem. tātai tad, ja tas atrodas tagā ``.

ul em {

color: #000000;

}

4. Klašu atlasītāji.

1) Pamatojoties uz klases elementu atribūtu. Visi elem., kuriem ir šī klase, tiks formātēti saskaņā ar definēto noteikumu.

.blacu {

color: #000000;

}

2) Šis noteikums satur melnu krāsu uz zām elementam, kura klases atribūts mūsu dokumentā ir iestatīts uz melnu. Varat to padarīt nedaudz konkrētāku.

Black {

color: #000000;

}

3) Šis noteikums atveido saturu melnā krāsā tīkai **<h1>** elementiem, kuru klases atribūts ir iestatīts uz melnu.

<p class="center Bold">

Tekssts

</p>

5. ID atlasiitāji. Vai definēt stila noteikumus, pamatojoties uz elementa id atribūtu.

• Visi elem. tiks formātēti ar šo ID sasaukanā ar definēto vārtulu.

Black {

color: #000000;

}

2) Šis vārtulas satur tien atveidots melnā krāsā uz zām elementam, kura ID atribūts mūsu dokumentā ir iestatīts nā melns.

• Vai to padarīt nedaudz konkrētāku.

h1 # Black {

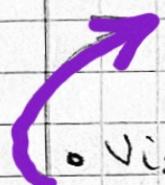
color: #000000;

}

• Šī uārtula saturs tiek atveidots melnā krāsā tūai elem. `<h1>`, kuru id atribūts ir iestatīts nā melns.

3) ID atlasiņāju patiesais spēns ir tad, ja tas pamato nā pamatu pēchnācēju atlasiņājiem.

Black h2 {
color: #000000;



• Visi 2. līmena viersnīci tiek parādīti melnā krāsā, ja šie viersnīci atradēsies tagos, kuru id atribūts ir iestatīts nā melns.

6. Child Selectors. Bērnu atlasiņāji.

Vēl viens sentora veids, kas ir loti līdzīgs pēchnācēju atlasiņājiem, taču tam ir atšķirīga funkcionalitāte.

Body > p {
color: #000000;
}

• Šis noteikums visas iedujas atvēdos melnā, ja tās ir elementa `<body>` tiesīe atkarīnājumi. Citas iedujas, kas ir ievietotas citos elem., piem., `<div>` vai `<td>`, neietiek sīs noteikums.

7. Atribūtu atlasītāji. Norādīta stila
nākuma atbildē visiem ievades elementiem,
neviem ir tipa atrībūts ar vērtību "text".

input [type="text"] {

color: #000000;

}

- Netiek ieteikmēts un vāsa tiek piemērota
tīnai vēlamajiem teusto laumiem.

Atribūtu atlasītājam tiek piemēroti šādi
noteikumi:

• p [lang] - atlasa visus iedruopas elementus
ar lang atrībūtu.

• p [lang="fr"] atlasa visus iedruopas
elementus, naru lang atrībūta vērtība ir
tiesī "fr".

• p [lang ~="fr"] atlasa visus iedruopas
elementus, naru lang atrībūta ir vārds "fr".

• p [lang |= "en"] atlasa visus iedruopas
elementus, naru lang atrībūts satur vērtības
nas ir tiesī "en" vai sānas ar "en-".

Vairāki stila noteikumi:

h1 {

color: #36C;

font-weight: normal;

letter-spacing: .4em;

margin-bottom: 1em;

text-transform: lowercase;

}

```
h1, h2, h3 {  
    color: #36C;  
    font-weight: normal;  
    letter-spacing: .4em;  
    margin-bottom: 1em;  
    text-transform: lowercase;  
}
```

```
#content, #footer, #supplement {  
    position: absolute;  
    left: 510px;  
    width: 280px;  
}
```

Jūs veidi nā ievietot CSS:

- External; (Ārējais) CSS
 - Internal; CSS (Ciešējais)
 - Inline CSS; (zindā)
- Izmantojot ārējo stila lapu, var mainīt visas vietas izmaksas, mainot tīcīe 1 failu.
- Katrā HTML lapā už jāizvaij atsauce uz ārējo stilu lapas faila elementā `<link>` galvenes sadalā.
`<head>`
`<link rel="stylesheet" href="mystyle.css">`
`</head>`
- Ciešējo stila lapu var izmantot, ja vienai HTML lapai už unikāls stils.
- Ciešējais stils ir definēts elem. `<style>`, galvenes sadalā.

```
<style>
body {
    background-color: linen;
}
h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
```

- **Inline style** (iešlautais stils) var izmantot, lai vienam elementam piemērotu ieviņātu stilu.

- Lai izmantotu iešlautos stilus, pievienojiet stila atribūtu attiecīgajam elementam.
- Stila atribūts var satvērt jebnuru CSS rewižētu.

```
<h1 style="color: blue; text-align: center;"> Viens rads </h1>
<p style="color: red;"> Tevsts </p>
```

- Ja daži ceļviāti ir definēti vienam un tam pašam atlasiņājam (elem.) dažādās stilek lapās, tins izmantojota vērtība no pēdējās lasītās stila lapas.

Ārējā stila lapa (External):

```
h1 {
    color: navy;
}
```

Iesējējā stila lapa (Internal):

```
h1 {
    color: orange;
}
```

JavaScript

- JavaScript, bieži sašināts kā JS, ir programmēšanas valoda, kas ir viena no globālā tīmekļa galvenajām tehnoloģijām, kā arī daļa no HTML un CSS.

- **JavaScript** ir augsta līmena, bieži vien tiesi laiņa kompileta valoda, kas atbilst **ECMAScript** standartam.

- **JavaScript** daudzās sānotnējā tīmekļa iemantotā tīmekļa pārlūpprogrammās, tāču tagad tie ir daļu serveru un daudzu lietojumprogrammu galvenie komponenti. Populārāna iepildlaiņa sistēma šim lietojumam ir **Node.js**.

- Lai gan Java un JavaScript ir līdzīgas pēc nosaukuma, sintaxe un attiecīgajām standarta bibliotēkām, abas valodas ir atšķirīgas un ievērojami atšķiras pēc diadina.

- JavaScript ir pasaulei populārāna programmēšanas valoda.

- JavaScript izveidoja **Brendans Eihs** 1995.gadā, un tas nolika par **ECMA** standartu 1997.g.

- **ECMA-262** ir standarta oficiālais nosaukums.
ECMAScript ir valodas oficiālais nosaukums.

<script> tags.

- HTML valodā JavaScript rads tieši ievietots stāv tagiem **<script>** un **</script>**.

```
<body>
  <p id="demo"></p>
  <script>
    document.getElementById("demo").innerHTML = "My first JS";
  </script>
</body>
```

Output:

My first JavaScript

JavaScript funkcijas un notieņumi.

- JS function ir JS iedaļš bloks, ko var iapildēt, kad tas tiek izsaņuts.
 - Piem., funkcija var izsaņut, kad notiek notieņums, piemēram, lietotājs nomainīja vienu pogas.

JavaScript sadaļā <head> vai <body>

- HTML dokumentā varat ievietot neierobežotu skaitu scriptu.
- Scriptus var ievietot HTML lapas sadaļā <body> vai <head>, vai abos.

JavaScript in <head>

- Šajā piemērā JS function ir ievietota HTML lapas sadaļā <head>.
 - Funkcija tiek izsaņuta, kad tiek nomainīts viens pogas:

```
<head>
<script>
function myFunction() {
    document.getElementById("demo").innerHTML = "Paragraph changed."
}
</script>
</head>
<body>


Texts.


<button type="button" onclick="myFunction()"> Poga </button>
</body>
```

Output:

Pirms:

Texts.

Poga

Pēc:

Paragraph changed.

Poga

(Kad uzsprīdīja pogas, tekssts mainījās.)

JS <body>

- Šajā piemērā JS function ievietota HTML lapas sadaļā <body>.
- Funkcija tiek ievadīta, kad tiek no nomainījīgās. mē pagās:

```
<body>
<p id="demo"> Teists, </p>
<button type="button" onclick="myFunction()"> Poga </button>
<script>
function myFunction() {
    document.getElementById("demo").innerHTML = "Teists iamaicināts." ;
}
</script>
</body>
```

Output - tāds pats kā iepriekš!

- Scriptu ievietošana elementā <body> apņemtā ualabo attēlošanas ātrumu, jo skripta interpretācija paleinās displejū.

External (Ārējais) JS.

- Scriptus var ievietot arī ārējos failos:

External file: myScript.js

```
function myFunction () {
    document.getElementById ("demo").innerHTML = "Teists iamaicinās";
}
```

- Ārējie skripti ievietoti ārējā failā, ja viens un tas pats mēs tiek izmantots daudzās dažādās tīmekļa lapās.

- JS failiem ir faila paplašinājums .js.

- Lai izmantotu ārējū skripty, ievietojiet skripta faila nosaukumu <script> taga atributā src (source):

```
<body>
<p id="demo"> Teists </p>
<button type="button" onclick="myFunction ()"> Poga </button>
<script src="myScript.js"> </script>
</body>
```

- Šis piemērs atrodas vienā saistīts ar "myScript.js".
(myFunction tiek glabāta "myScript.js")

Outputs ā fāds pat nā iepriekš!

- Vaiat ievietot ārejā skripta atsauci sadaļā `<head>` vai `<body>`.
- Skripts daubosies tā, it nā tas atrastos tieši tur, nee atrodas tags `<script>`.
- Ārējie skripti nedrīkst saturēt tagus `<script>`.

Ārejās JS priešrocības.

Skriptu ievietošanai ārejās failos ir dažas priešrocības:

- Tas atdala HTML un nodu;
- Tas atvieglo HTML un JS lasīšanu un izmaksu;
- Kšķatniņā saglabātie JS faili var pātrināt lapu ielādei.

Lai vienai lapai pievienotu vairākus skripta failus, izmantojiet vairākus skripta tagus:

```
<script src="myScript1.js"></script>
<script src="myScript2.js"></script>
```

Ārejās atsauces:

Ils ārejo skriptu var atsaunties 3 dažādos veidos:

- 1) Ar pilnu URL (pilnu tīmekļa adresi);
- 2) Ar faila ceļu (piem., /js/);
- 3) Bez jebnāda ceļa.

1) Šajā piemērā tiek izmantots pilns URL, lai iaveidotu saitīcu myScript.js:

```
<body>
<p id="demo"> Testi </p>
<button type="button" onclick="myFunction ()"> Poga </button>
<script src="https://www.w3schools.com/js/myScript.js">
</script>
</body>
```

2) file path:

```
<body>
<p id="demo"> Testis </p>
<button type="button" onclick="myFunction()"> Poga </button>
<script src="/js/myScript.js"></script>
</body>
```

3) no path:

```
<body>
<p id="demo"> Testis </p>
<button type="button" onclick="myFunction()"> Poga </button>
<script src="myScript.js"></script>
</body>
```

JavaScript Output.

JavaScript displeja iespējas.

JS var "parādīt" datus dažādos veidos:

- Raustīšana HTML elementā, izmantojot innerHTML;
- Ieraustīšana HTML ieradē, izmantojot document.write();
- Raustīšana brūdingājuma lodaīnā, izmantojot window.alert();
- Raustīšana pārlūpprogrammas konsolē, izmantojot console.log();

InnerHTML izmantašana.

- Lai pievērtu HTML elementam, JavaScript var izmantot metodi document.getElementById(id).
- Attributs id nosaka HTML elementu. InnerHTML renvisās nosaka HTML saturu;

```
<body>
<p id="demo"></p>
<script>
  document.getElementById("demo").innerHTML = 5+6;
</script>
```

Output:

- HTML elementa ievērējā HTML renvieta maina ī ieraksts veids, kā parādīt datus HTML.

document.write() iemantāšana.

- Testējotas redzētu to ir ēri iemantot:

```
<body>  
<script>  
    document.write(5 + 6);  
</script>  
</body>
```

Output:

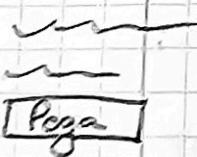
11

- Nevis neissiet document.write() pēc dokumenta izlades parādījumos. Tas pārautis viesu dokumentu.

- Iemantojot document.write() pēc HTML dokumenta izlades, tiks izdzēsts viss esotais HTML:

```
<body>  
<button type="button" onclick="document.write(5 + 6)">Poga</button>  
</body>
```

Output piems:



pēc

11

window.alert() iemantāšana.

- Varat iemantot brūdinājuma lodiņu, lai parādītu datus:

```
<body>  
<script>  
    window.alert(5 + 6);  
</script>  
</body>
```

- Parādīsies parādījuma lauciņš ar varaustu it.

"saiti.com norāda:
11"

(labi)

- Jās varat izlaist window atlēguārdu.

- Programmā JS window objekts ir globālā tērējuma objekts. Tas nozīmē, ka mainīgiz, renviāti un metodēs pēc norādījuma nerādītāna nav obligāta:

```
<Body>  
<script>  
alert(5 + 6);  
</script>  
</Body>
```

- Outputs tāds pārds.

console.log() izmantotāšana.

• Atulūdošanas nolūkos var izsaucīt metodi console.log()
pārlūpēprogrammā, lai parādītu datus.

```
<Body>  
<script>  
console.log(5 + 6);  
</script>  
</Body>
```

- Tastatūras taustiņš F12 + Fn
atvīvās atulūdatānu.
- Pēc tam atulūdotāja ierēķīne
atlasiet "Console".
- Pēc tam vēlreis nos piediet
palaist ("Run").

JavaScript Print

- JS nav neviens drūmas objekta vai drūmāšanas metodes.
- Šīs nevarēat pievērt ievadītācīem, izmantojot JS.
- Vienīgais ianēmums ir tas, ka pārlūpēprogrammā varat izsaucīt
metodi window.print(), lai iadrūmātu pārējā loga saturu.

```
<Body>  
<button onclick="window.print()">Printēt šo lapu. </button>  
</Body>
```

Output:

Printēt šo lapu.

← Parādīsies logs, kurš piedāvās iprintēt
lapu ar dato tehniku.

JavaScript Statements (pāriņojumi)

- JS programma ir to pāriņojumu saraksts, kas jāapilda datorā.

```
<body>
<p id="demo"></p>
<script>
let x, y, z;           //Statement 1
x=5;                  // Statement 2
y=6;                  // Statement 3
z=x+y;                //Statement 4
```

f
—
t
document.getElementById("demo").innerHTML =
"The value of z is " + z + ". "
</script>
</body>

Output:

The value of z is 11.

JavaScript programmas.

- Datozprogramma ir "instruciju" saraksts, kas datoram "jāapilda".
 - Programmēšanas valodā šīs programmēšanas instrucijas sauc par pāriņojumiem.
 - JavaScript programma ir programmēšanas pūrenšrastu saraksts.
 - HTML valodā JS programmas iepilda tīmekļa pārlūpprogrammas.

JavaScript pāriņojumi.

JavaScript pāriņojumi sastāv no:

Values, Operators, Expressions, Keywords and Comments.

- Šis pāriņojums liec pārlūpprogrammai ieraustīt "Hello Dolly".
HTML elementā ar id="demo":