Telerik Academy

http://academy.telerik.com

Introduction to JavaScript Development

The Magic of Dynamic Web Pages

JavaScript Fundamentals

Telerik Software Academy

http://academy.telerik.com

Telerik Academy

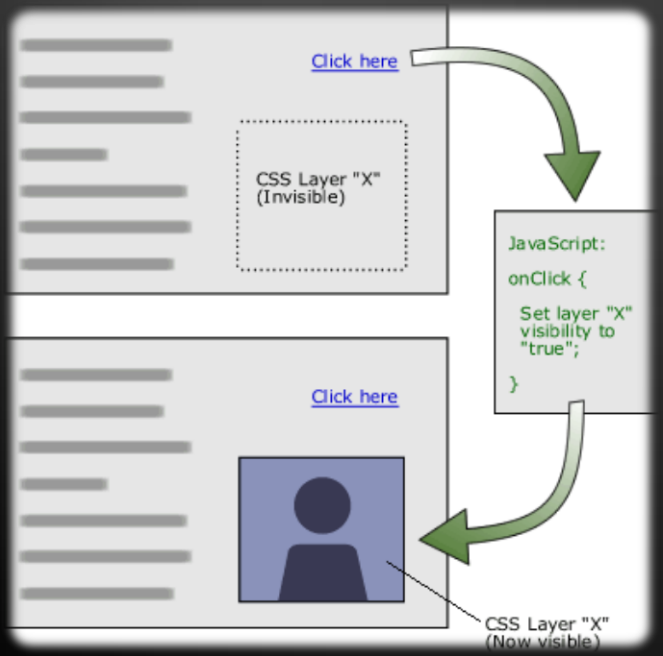
Table of Contents

- Dynamic HTML
- How to Create DHTML?
  - XHTML, CSS, JavaScript, DOM
- Intro to JavaScript
  - JavaScript in Web Pages
- JavaScript Syntax
- Pop-up boxes
- Debugging in JavaScript

Telerik Academy

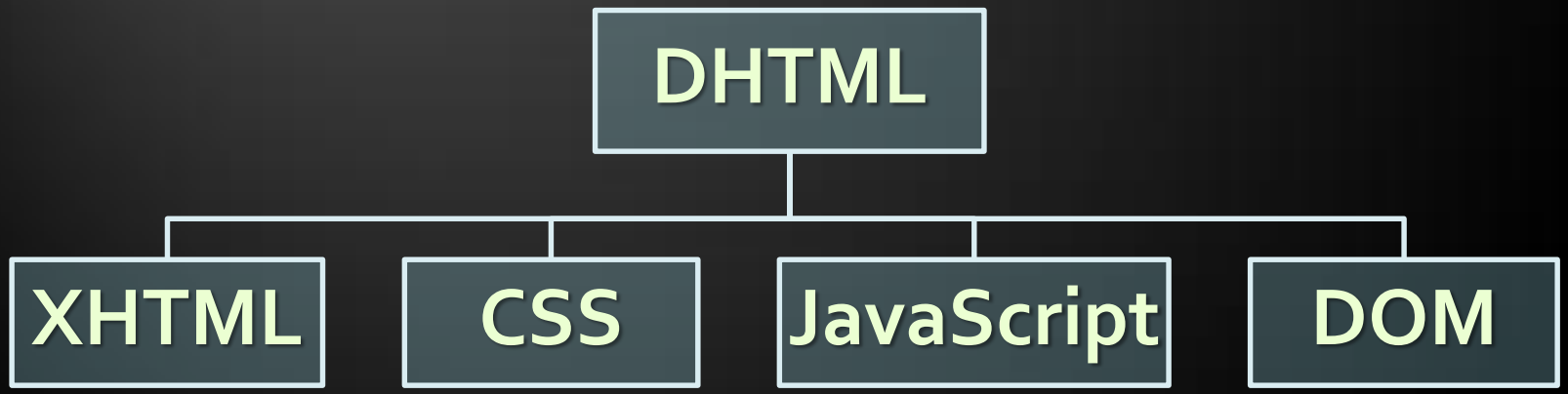
Dynamic HTML

Dynamic Behavior at the Client Side



What is DHTML?

- Dynamic HTML (DHTML)
  - Makes possible a Web page to react and change in response to the user's actions
- DHTML consists of HTML + CSS + JavaScript



Telerik Academy

DTHML = HTML + CSS + JavaScript

- HTML defines Web sites content through semantic tags (headings, paragraphs, lists, …)
- CSS defines 'rules' or 'styles' for presenting every aspect of an HTML document
  - Font (family, size, color, weight, etc.)
  - Background (color, image, position, repeat)
  - Position and layout (of any object on the page)
- JavaScript defines dynamic behavior
  - Programming logic for interaction with the user, to handle events, etc.

Telerik Academy

JavaScript

Dynamic Behavior in a Web Page

JavaScript

- JavaScript is a front-end scripting language developed by Netscape for dynamic content
  - Lightweight, but with limited capabilities
  - Can be used as object-oriented language
  - Embedded in your HTML page
  - Interpreted by the Web browser
- Client-side, mobile and desktop technology
- Simple and flexible
- Powerful to manipulate the DOM

JavaScript Advantages

- JavaScript allows interactivity such as:
  - Implementing form validation
  - React to user actions, e.g. handle keys
  - Changing an image on moving mouse over it
  - Sections of a page appearing and disappearing
  - Content loading and changing dynamically
  - Performing complex calculations
  - Custom HTML controls, e.g. scrollable table
  - Implementing AJAX functionality

What Can JavaScript Do?

- Can handle events
- Can read and write HTML elements and modify the DOM tree
- Can validate form data
- Can access / modify browser cookies
- Can detect the user's browser and OS
- Can be used as object-oriented language
- Can handle exceptions
- Can perform asynchronous server calls (AJAX)

JavaScript Engines

- Depends on Browser
  - V8 in Chrome, Chakra in IE, Spidermonkey in Firefox, JavaScriptCore for Safari, etc.
- Services
  - Memory Management / GC
  - Just-in-Time Compilation
  - Type System
  - etc.

```
1 [['0']] == false; // true
2 [['0']] == true; // false
```

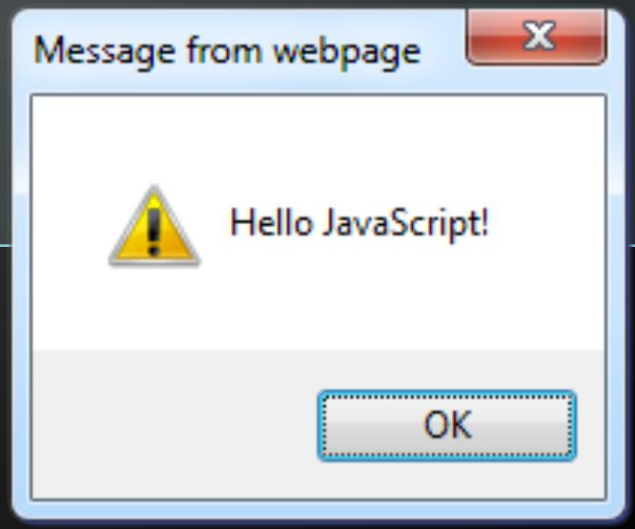
```
1 [] + []; // ""
2 {} + {}; // NaN
3 [] + {}; // "[object Object]"
4 {} + []; // 0
```

The First Script

```
<html>

<body>
  <script type="text/javascript">
     alert('Hello JavaScript!');
  </script>
</body>

</html>
```



Telerik Academy

First JavaScript

Live Demo

Using JavaScript Code

- The JavaScript code can be placed in:
  - `<script>` tag in the head
  - `<script>` tag in the body - not recommended
  - External files, linked via `<script>` tag the head
    - Files usually have `.js` extension

```
<script src="scripts.js" type="text/javscript">
<!- code placed here will not be executed! -->
</script>
```

    - Highly recommended
    - The `.js` files get cached by the browser

Telerik Academy

JavaScript – When is Executed?

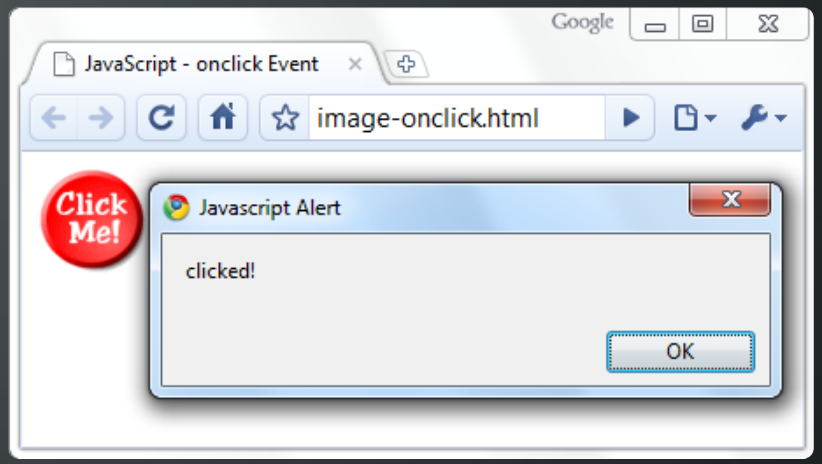
- JavaScript code is executed during the page loading or when the browser fires an event
  - All statements are executed at page loading
  - Some statements just define functions that can be called later
  - No compile time checks
- Function calls or code can be attached as "event handlers" via tag attributes
  - Executed when the event is fired by the browser

```
<img src="logo.gif" onclick="alert('clicked!')" />
```

Calling a JavaScript Function from Event Handler – Example

```
<html>
<head>
<script type="text/javascript">
  function test (message) {
    alert(message);
  }
</script>
</head>

<body>
  <img src="logo.gif"
    onclick="test('clicked!')" />
</body>
</html>
```



Telerik Academy

Event Handlers

Live Demo

Telerik Academy

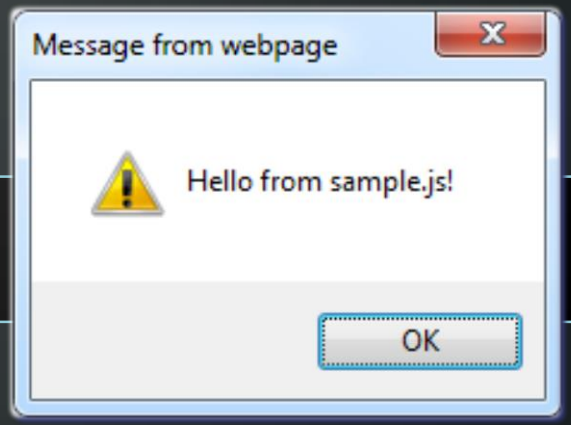
Using External Script Files

- Using external script files:

external-JavaScript.html

```
<html>
<head>
  <script src="sample.js" type="text/javascript">
  </script>
</head>
<body>
  <button onclick="sample()" value="Call JavaScript
    function from sample.js" />
</body>
</html>
```

The <script> tag is always empty.



- External JavaScript file:

```
function sample() {
  alert('Hello from sample.js!')
}
```

sample.js

Telerik Academy

External JavaScript Files

Live Demo

Telerik Academy

Node.js

Running JavaScript in a console

Telerik Academy

Node.js Overview

- Node.js is a server-side platform that uses JavaScript
  - Runs the V8 JS interpretour
  - Allows creating end-to-end apps with JavaScript
  - Usable to test & learn JavaScript Syntax

Telerik Academy

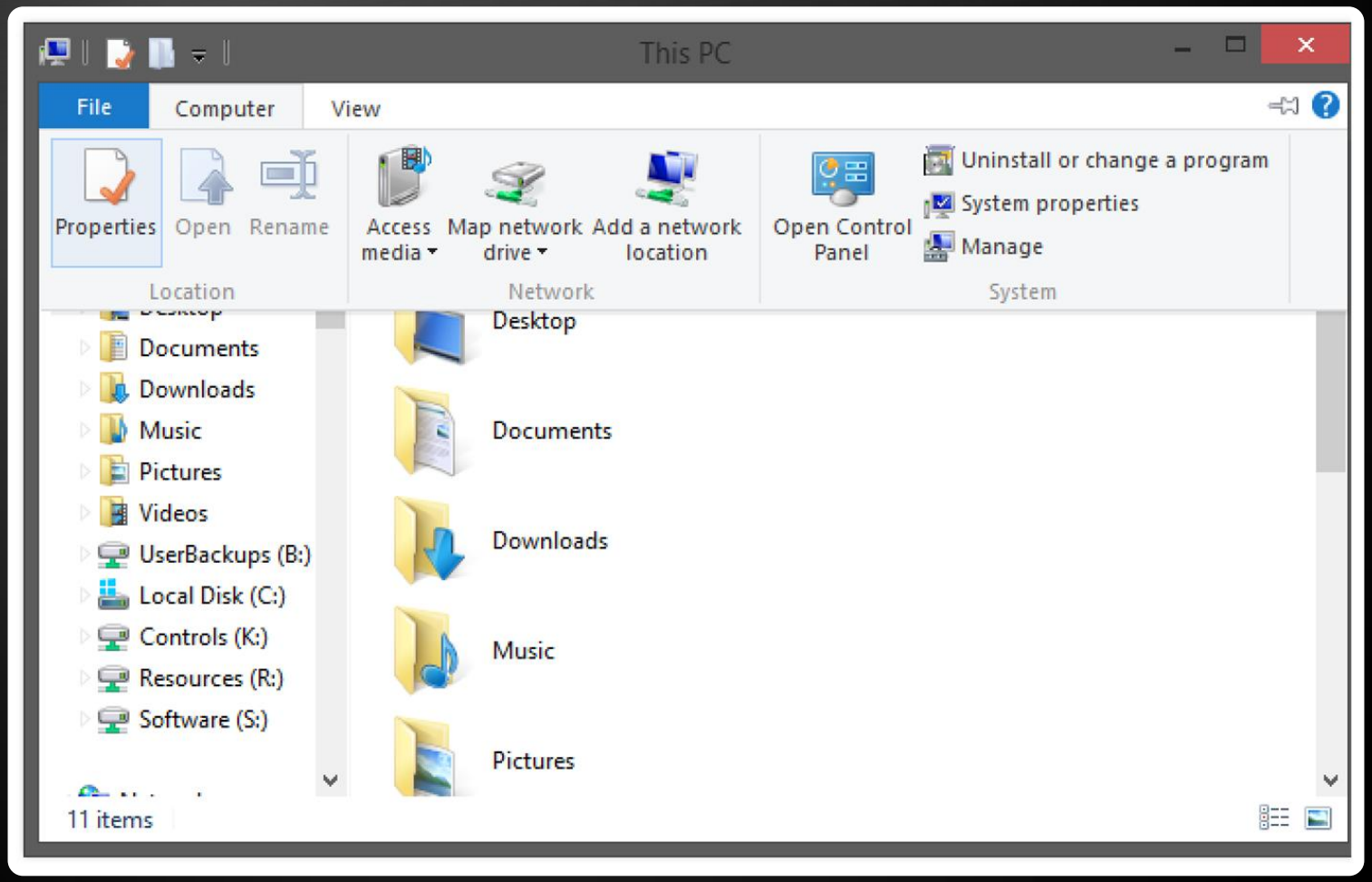
Installing Node.js

1. Visit the Node.js website: https://nodejs.org/
2. Next -> Next -> Next -> …
3. Make sure Node.js is added to PATH
4. Node.js is installed on the machine and can be used through the CMD/Terminal
5. In the CMD/Terminal run "node –v"
   - Should return the version, if node is installed and working

Telerik Academy

Adding Node.js to Path on Windows (1)

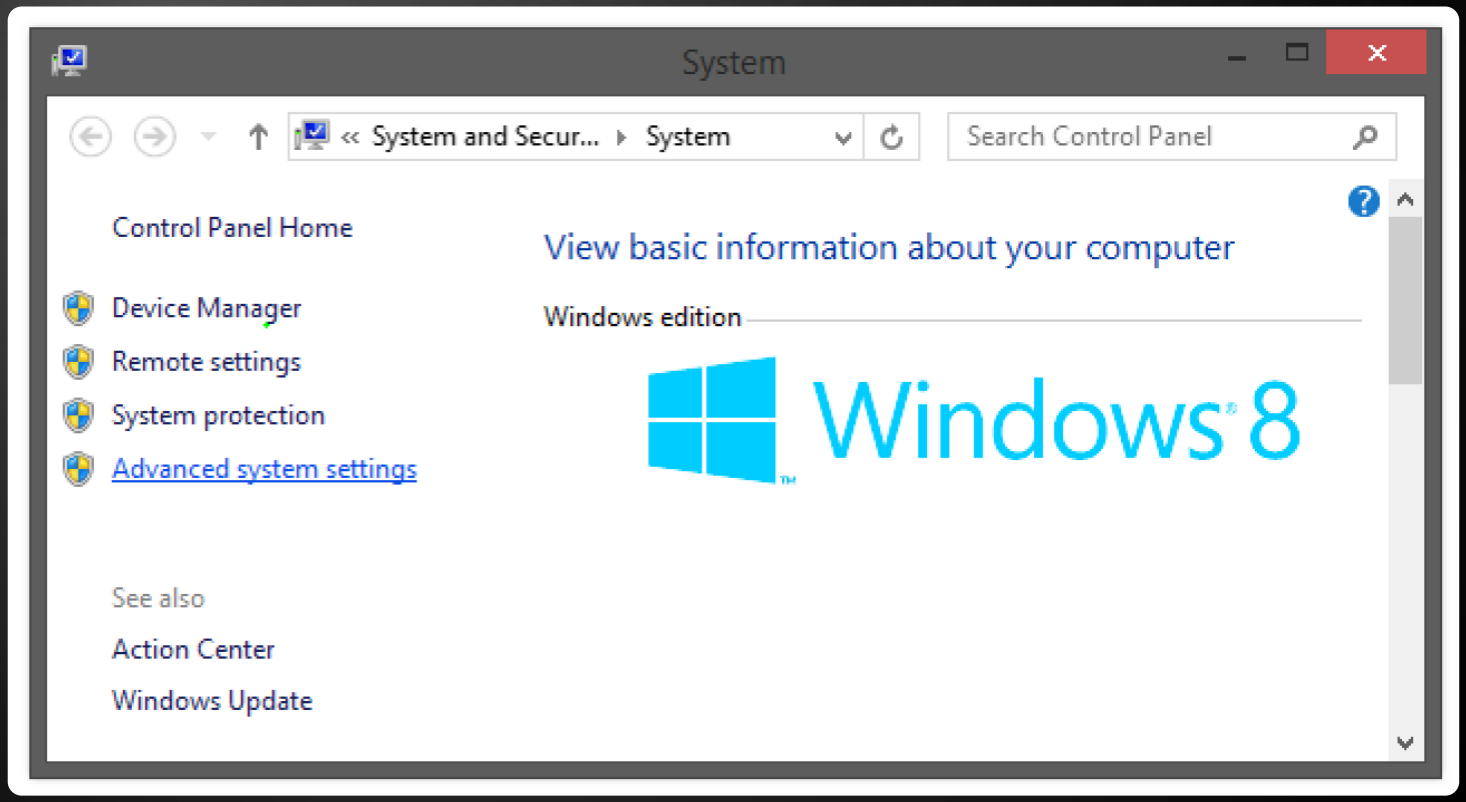
- Go to "Computer" -> "Properties"



Telerik Academy

Adding Node.js to Path on Windows (2)

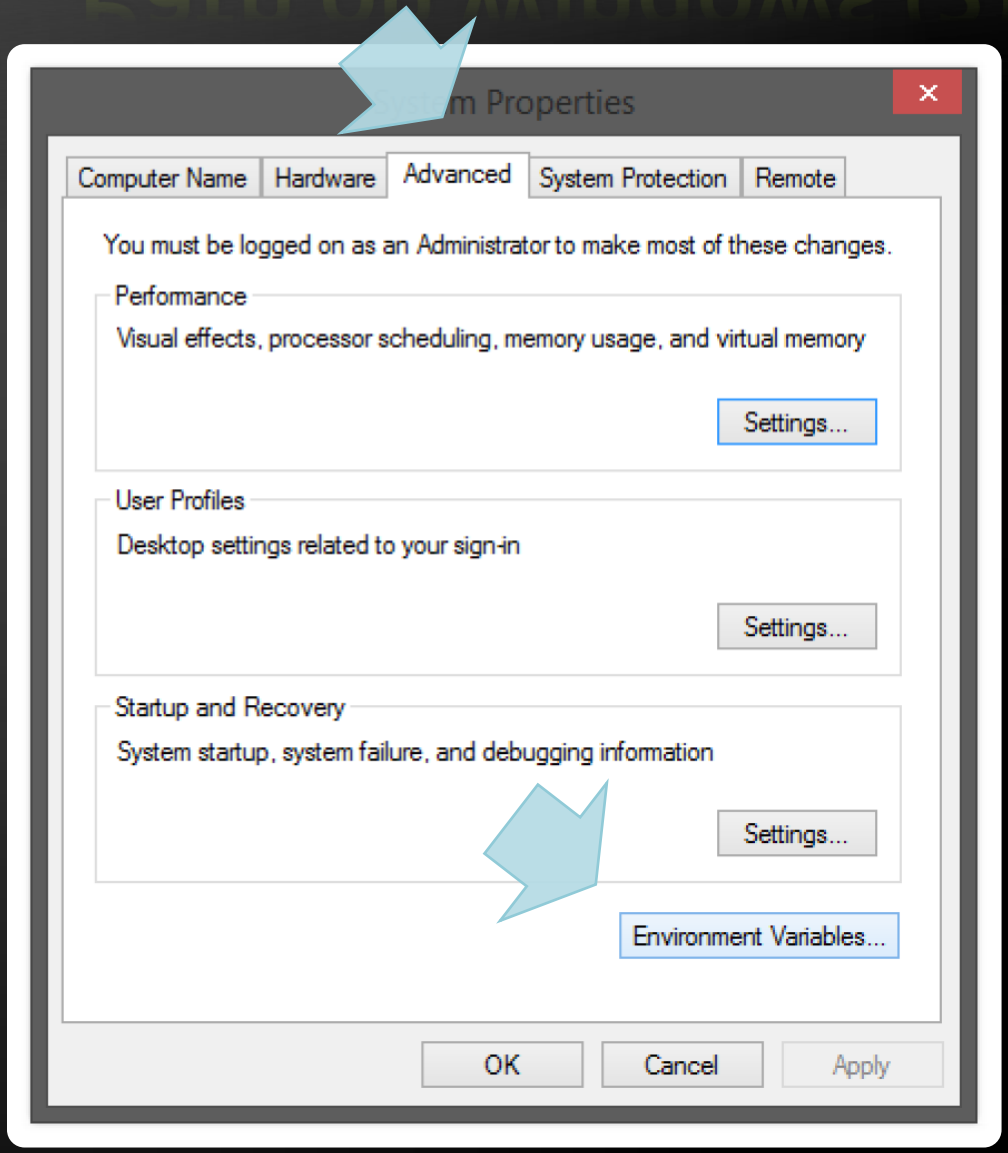
- Go to "Advanced system settings"



Telerik Academy

Adding Node.js to Path on Windows (3)

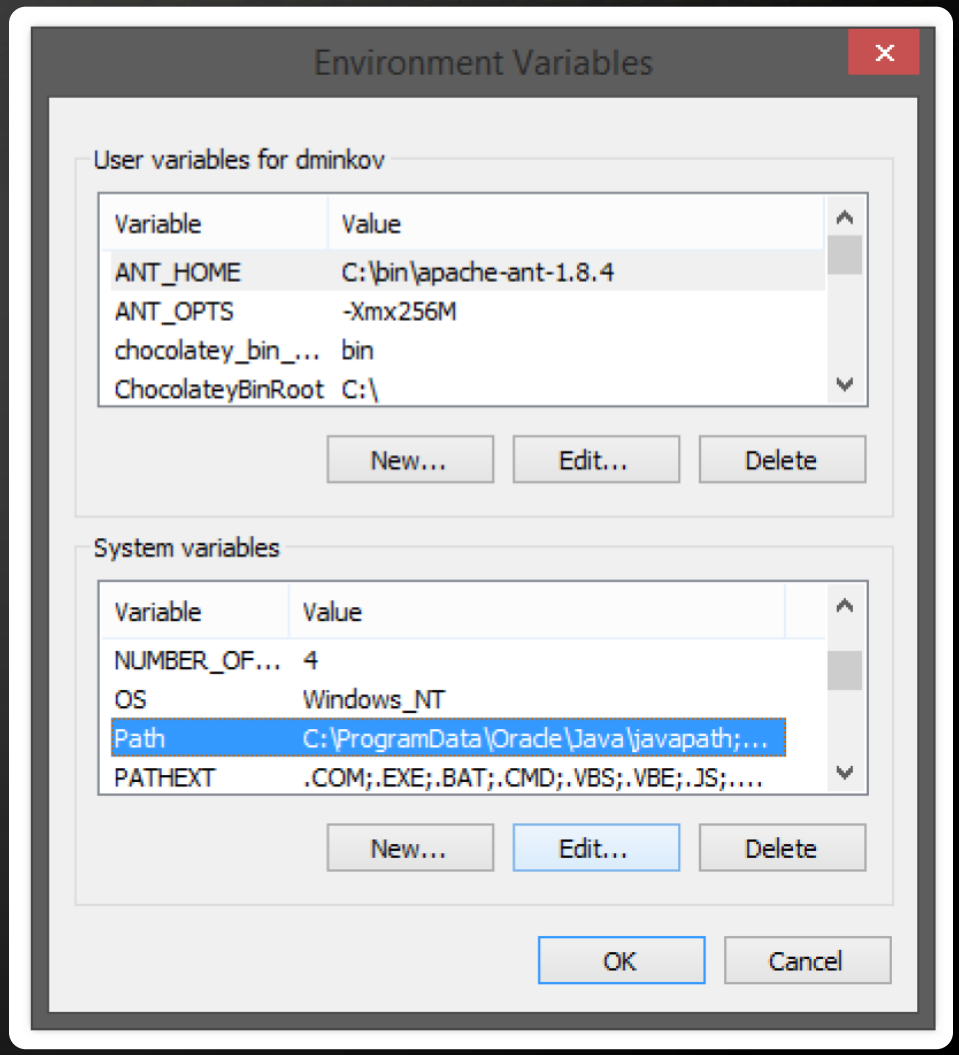
- Go to "Advanced" -> "Environment Variables"



Telerik Academy

Adding Node.js to Path on Windows (4)

- Go to "System Variables"
  - -> "Path"
  - -> "Edit"

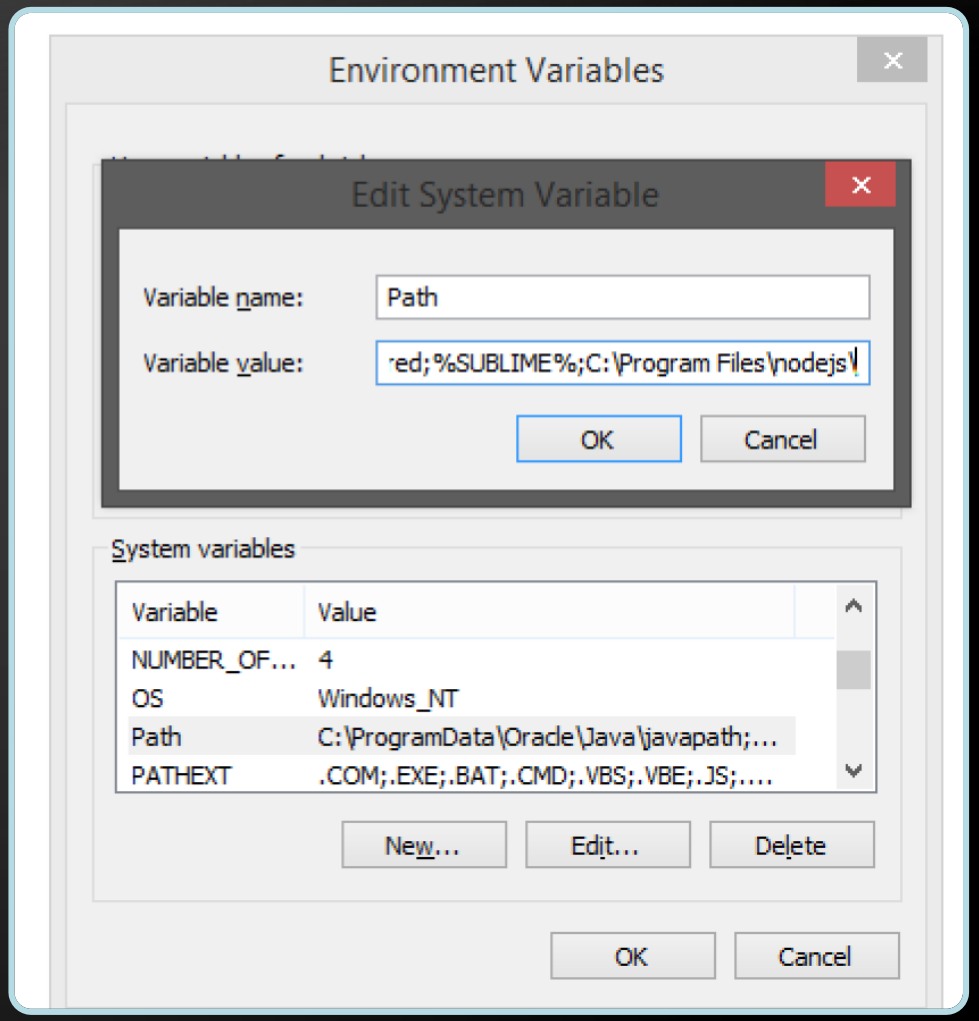


Telerik Academy

Adding Node.js to Path on Windows (5)

- Add to the end:

```
;C:\Program Files\nodejs\
```



Telerik Academy

Adding Node.js To Path on Windows

Live Demo

Telerik Academy

Adding Node.js to Path on Linux and OS X

- Open ~/.bashrc with admin sudo
- Find the Node.js path
  - Usually it is "/usr/bin/node"
- Add the Node.js path to ~/.bashrc

```
PATH=/usr/bin/node:$PATH
```

- You have Node.js in the PATH

Telerik Academy

Adding Node.js to Path on OS X & Linux

Live Demo

Telerik Academy

The JavaScript Syntax

```
if (pop < 10)
{
    map.graphics.add(features[i].setSymbol(onePopSymbol));
}
else if (pop >= 10 && pop < 95)
{
    map.graphics.add(features[i].setSymbol(twoPopSymbol));
}
else if (pop >= 95 && pop < 365)
{
    map.graphics.add(features[i].setSymbol(threePopSymbol));
}
else if (pop >= 365 && pop < 1100)
{
    map.graphics.add(features[i].setSymbol(fourPopSymbol));
}
else
{
    map.graphics.add(features[i].setSymbol(fivePopSymbol));
}
```

JavaScript Syntax

- The JavaScript syntax is similar to C#
  - Operators (+, *, =, !=, &&, ++, …)
  - Variables (typeless)
  - Conditional statements (`if`, `else`)
  - Loops (`for`, `while`)
  - Arrays (`my_array[]`) and associative arrays (`my_array['abc']`)
  - Functions (can return value)

Standard Popup Boxes

- Alert box with text and [OK] button
  - Just a message shown in a dialog box:

```
alert("Some text here");
```

- Confirmation box
  - Contains text, [OK] button and [Cancel] button:

```
confirm("Are you sure?");
```

- Prompt box
  - Contains text, input field with default value:

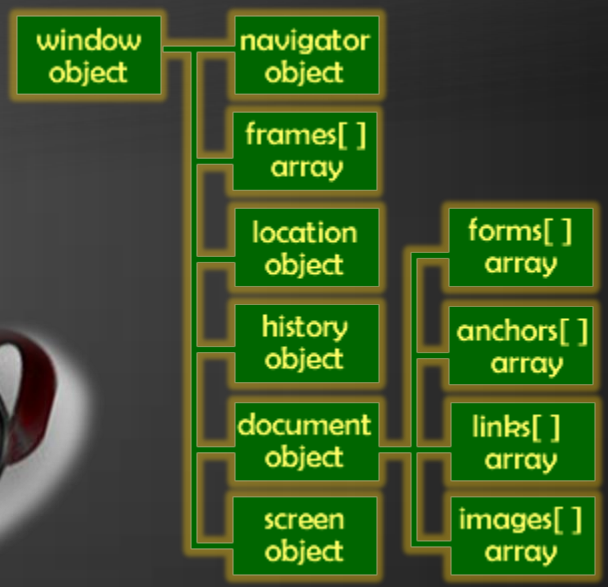
```
prompt ("enter amount", 10);
```

Telerik Academy

Popup Boxes

Live Demo

Telerik Academy

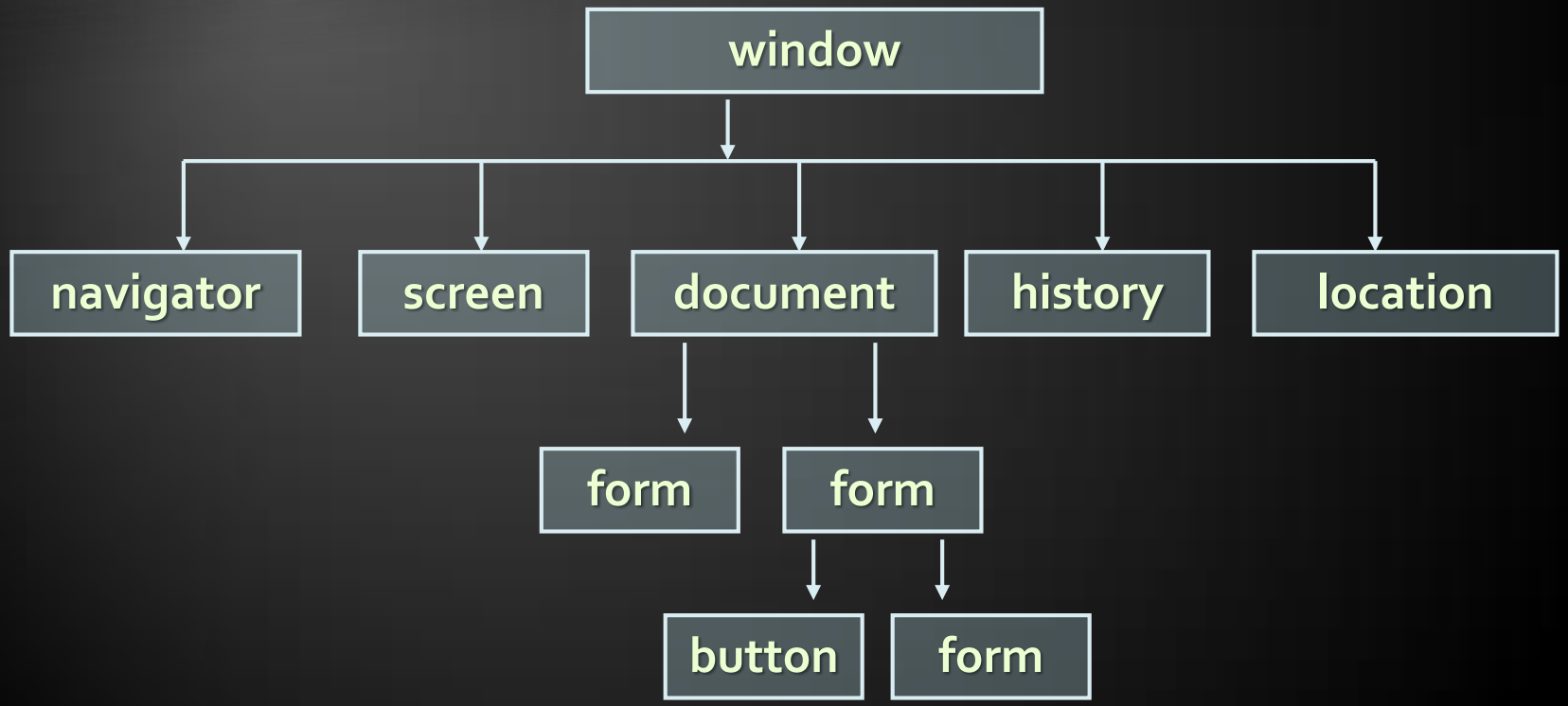


The Built-In Browser Objects

Built-in Browser Objects

- The browser provides some read-only data via:
  - `window`
    - The top node of the DOM tree
    - Represents the browser's window
  - `document`
    - holds information the current loaded document
  - `screen`
    - Holds the user's display properties
  - `browser`
    - Holds information about the browser

DOM Hierarchy – Example



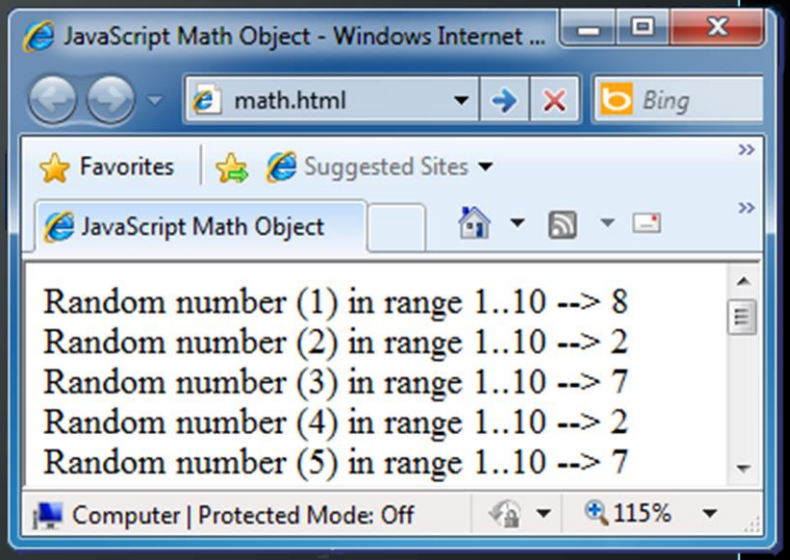
Telerik Academy

Other JavaScript Objects

The Math Object

- The Math object provides some mathematical functions

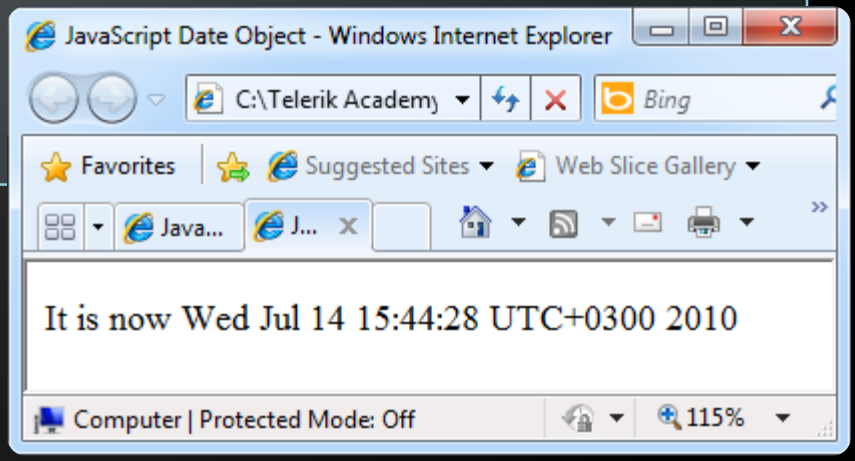
```
for (i=1; i<=20; i++) {
  var x = Math.random();
  x = 10*x + 1;
  x = Math.floor(x);
  document.write(
    "Random number (" +
    i + ") in range " +
    "1..10 --> " + x +
    "<br/>");
}
```



The Date Object

- The Date object provides date / calendar functions

```
var now = new Date();
var result = "It is now " + now;
document.getElementById("timeField")
   .innerText = result;
...
<p id="timeField"></p>
```



Timers: setTimeout()

- Make something happen (once) after a fixed delay

```
var timer = setTimeout(bang, 5000);
```

5 seconds after this statement executes, this function is called

```
clearTimeout(timer);
```

Cancels the timer

Timers: setInterval()

- Make something happen repeatedly at fixed intervals

```
var timer = setInterval(clock, 1000);
```

This function is called continuously per 1 second.

```
clearInterval(timer);
```

Stop the timer.

Timer – Example

```
<script type="text/javascript">
  function timerFunc() {
    var now = new Date();
    var hour = now.getHours();
    var min = now.getMinutes();
    var sec = now.getSeconds();
    document.getElementById("clock").value =
      "" + hour + ":" + min + ":" + sec;
  }

  setInterval(timerFunc, 1000);
</script>

<input type="text" id="clock" />
```

Telerik Academy

Other JavaScript Objects

Live Demo

Telerik Academy

Debugging JavaScript

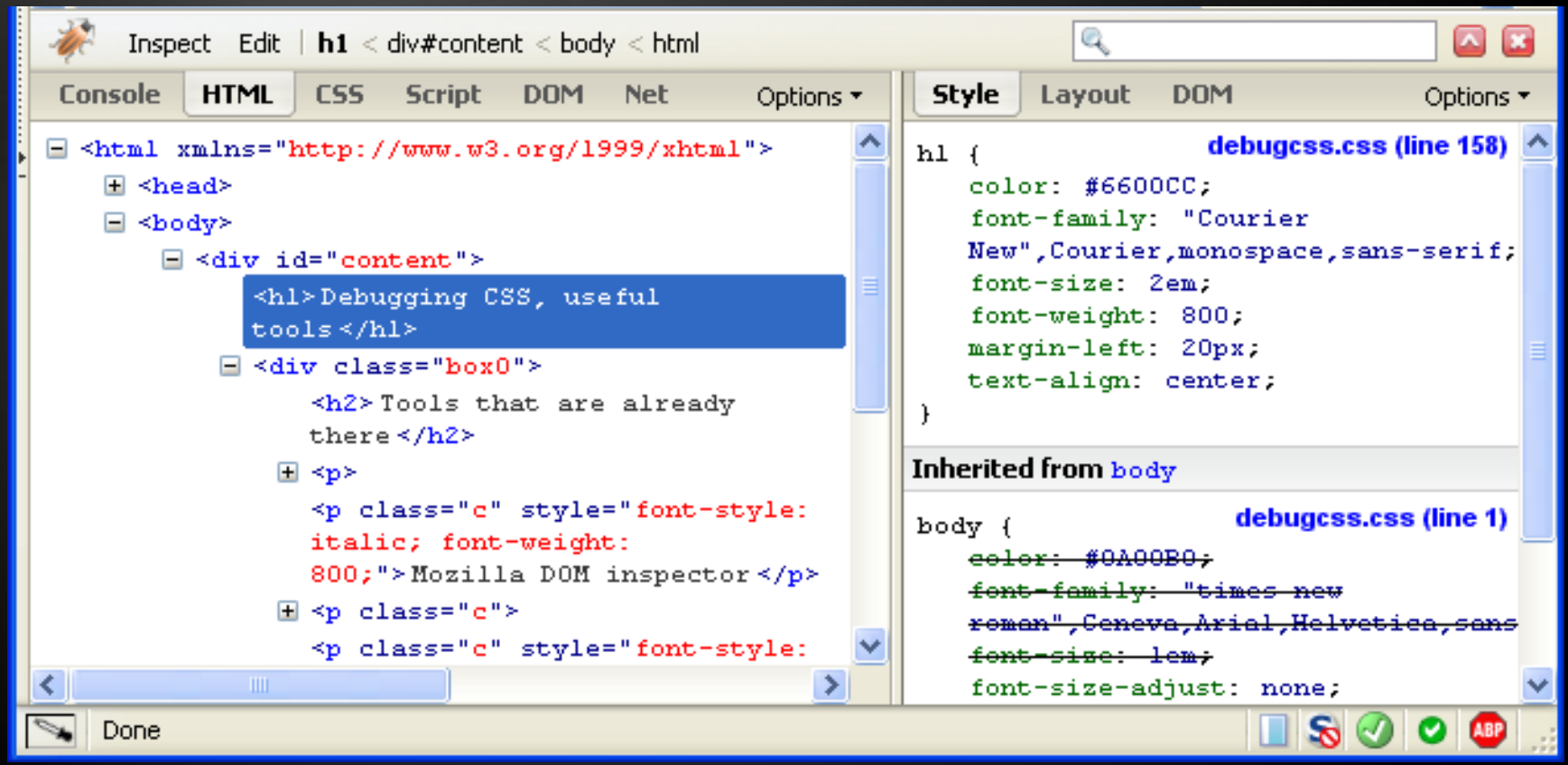
Debugging JavaScript

- Modern browsers have JavaScript console where errors in scripts are reported
  - Errors may differ across browsers
- Several tools to debug JavaScript
  - Microsoft Script Editor
    - Add-on for Internet Explorer
    - Supports breakpoints, watches
    - JavaScript statement `debugger;` opens the script editor

Firebug

- Firebug – Firefox add-on for debugging JavaScript, CSS, HTML
  - Supports breakpoints, watches, JavaScript console editor
  - Very useful for CSS and HTML too
    - You can edit all the document real-time: CSS, HTML, etc
    - Shows how CSS rules apply to element
  - Shows Ajax requests and responses
  - Firebug is written mostly in JavaScript

Firebug (2)


Inspect Edit | h1 < div#content < body < html
Console HTML CSS Script DOM Net Options
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<body>

<h1>Debugging CSS, useful tools</h1>

<h2>Tools that are already there</h2>

Mozilla DOM inspector

<p class="c" style="font-style:
Style Layout DOM Options
h1 { debugcss.css (line 158)
color: #6600CC;
font-family: "Courier New",Courier,monospace,sans-serif;
font-size: 2em;
font-weight: 800;
margin-left: 20px;
text-align: center;
}
Inherited from body
body { debugcss.css (line 1)
color: #0A00B0;
font-family: "times new roman",Geneva,Arial,Helvetica,sans
font-size: 1em;
font-size-adjust: none;
Done

Debugging JS in the Browser

Live Demo

Telerik Academy

Debugging Node.js

- JavaScript can also be debugged with Node.js:
  - Open Terminal/CMD
  - Run `$ npm install -g node-inspector`
  - In CMD/Terminal, navigate to the folder of the file to debug
  - Run in `$ node-debug FILE_TO_DEBUG`
  - A browser opens with the code, and it can be debugged

Debugging Node.js

Live Demo

Telerik Academy

JavaScript Console Object

- The `console` object exists only if there is a debugging tool that supports it
  - Used to write log messages at runtime
- Methods of the `console` object:
  - `debug(message)`
  - `info(message)`
  - `log(message)`
  - `warn(message)`
  - `error(message)`

Telerik Academy

Introduction JavaScript Development

Free Trainings @ Telerik Academy

- "Web Design with HTML 5, CSS 3 and JavaScript" course @ Telerik Academy
  - html5course.telerik.com

- Telerik Software Academy
  - academy.telerik.com

- Telerik Academy @ Facebook
  - facebook.com/TelerikAcademy

- Telerik Software Academy Forums
  - forums.academy.telerik.com