Troubleshooting, Debugging, and Cross-Browser Issues

Bok, Jong Soon
jongsoon.bok@gmail.com
https://github.com/swacademy

Simple Ways to Debug

- Is to use some form of output functionality to check the values of variables after processing.

  i.e. `alert(someVariable);`

  `document.write(someVariable);`

- If interested in a quick check, don't want to open a debugger, add the `alert` box, check the values you need to check, and then immediately remove the alert box.

Simple Ways to Debug (Cont.)

```
<script type="text/javascript">
  function sumNumbers(numArray) {
     var result = 0;
     if (numArray.length > 0) {
        for (var i = 0; i < numArray.length; i++) {
           if (typeof numArray[i] == "number") {
              result += numArray[i];
           } else {
              result = NaN;
              break;
           }
        }
     } else {
        result = NaN;
     }
     return result;
  }
  var ary = new Array(1,15,"three",5,5);
  var res = sumNumbers(ary); // res is NaN
  if (isNaN(res)) alert("Encountered a bad array or array element");
</script>
```

Code from : Shelley Powers, *JavaScript Cookbook*, O'Reilly, July 2010., pp.180-181

Development and Debugging Tools by Browser

- Google Chrome Developer Tools
  - https://developers.google.com/chrome-developer-tools/
- Firefox Firebug
  - http://getfirebug.com/faq/
- IE Built-in Debugger
  - http://msdn.microsoft.com/en-us/library/ie/gg699336(v=vs.85).aspx
- Opera Dragonfly
  - http://www.opera.com/dragonfly/
- Safari Web Development Tools
  - https://developer.apple.com/technologies/safari/developer-tools.html

Sample Code to Debugging

```
<html>
 <head>
  <title> JavaScript Debugging Example </title>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
  <script type="text/javascript">
      var display;
      function init() {
         display = document.getElementById("results");
      }
      function firstParam() {
         //set breakpoint here
         var a = 5;
         secondParam(a);
      }
      function secondParam(a) {
         var b = 10;
         thirdParam(a, b);
      }
      function thirdParam(a, b) {
         var c = 15;
         var d = a + b + c;
         if (window.console && window.console.log) {
            window.console.log(a + " + " + b + " + " + c + " = " + d);
         } else {
            display.innerHTML = a + " + " + b + " + " + c + " = " + d;
         }
      }
  </script>
 </head>
 <body onload="init()">
   <p><button onclick="firstParam();">Run</button></p>
   <div id="results"></div>
 </body>
</html>
```

Sample Code to Debugging (Cont.)

```
2  <html>
3   <head>
4    <title> New Document </title>
5    <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
6    <script type="text/javascript">
7     function hello(){
8        var msg = "Hello, World!";
9        console.log("%s", msg);
10    }
11   </script>
12  </head>
13  <body onload="hello()">
14    <p>Hi</p>
15  </body>
16 </html>
```

Chrome Developer Tools

- Select the **Wrench menu**  or  at the top-right of your browser window, then select **Tools** -> **Developer tools**.
- **Right-click** on any page element and select **Inspect element**.

Chrome Developer Tools (Cont.)

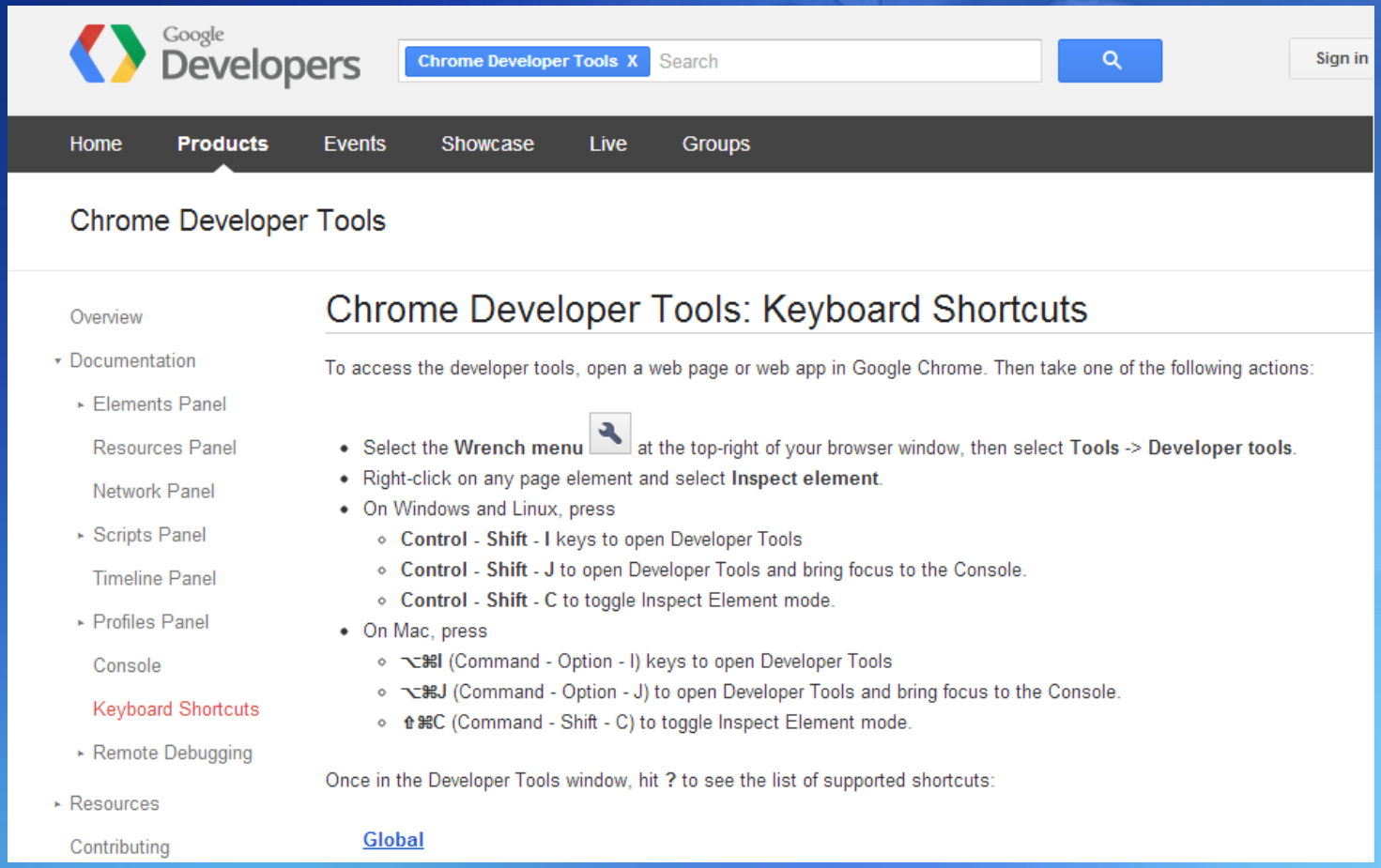
- Resources
  - http://www.youtube.com/watch?v=nOEw9iiopwI
  - http://www.youtube.com/watch?v=htZAU7FM7GI

Chrome Developer Tools (Cont.)

- On Windows and Linux, press
  - `Control - Shift - I` keys to open Developer Tools
  - `Control - Shift - J` to open Developer Tools and bring focus to the Console.
  - `Control - Shift - C` to toggle Inspect Element mode.
- On Mac, press
  - `⌥⌘I` (`Command - Option - I`) keys to open Developer Tools
  - `⌥⌘J` (`Command - Option - J`) to open Developer Tools and bring focus to the Console.
  - `^⌘C` (`Control - Option - C`) to toggle Inspect Element mode.

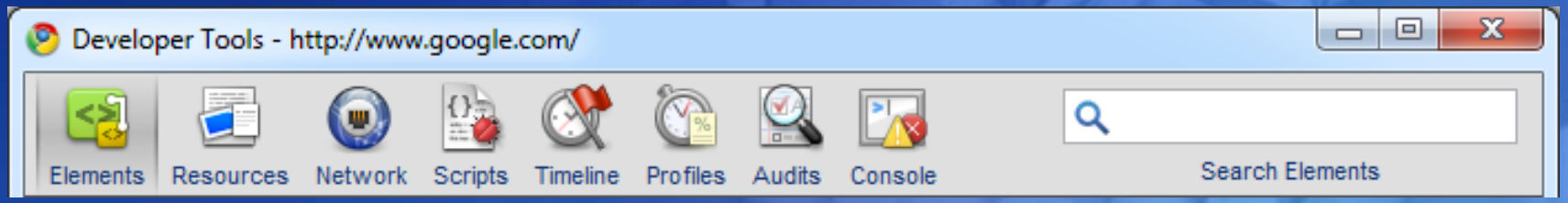
Chrome Developer Tools: Keyboard Shortcuts

- https://developers.google.com/chrome-developer-tools/docs/shortcuts



Chrome Developer Tools (Cont.)

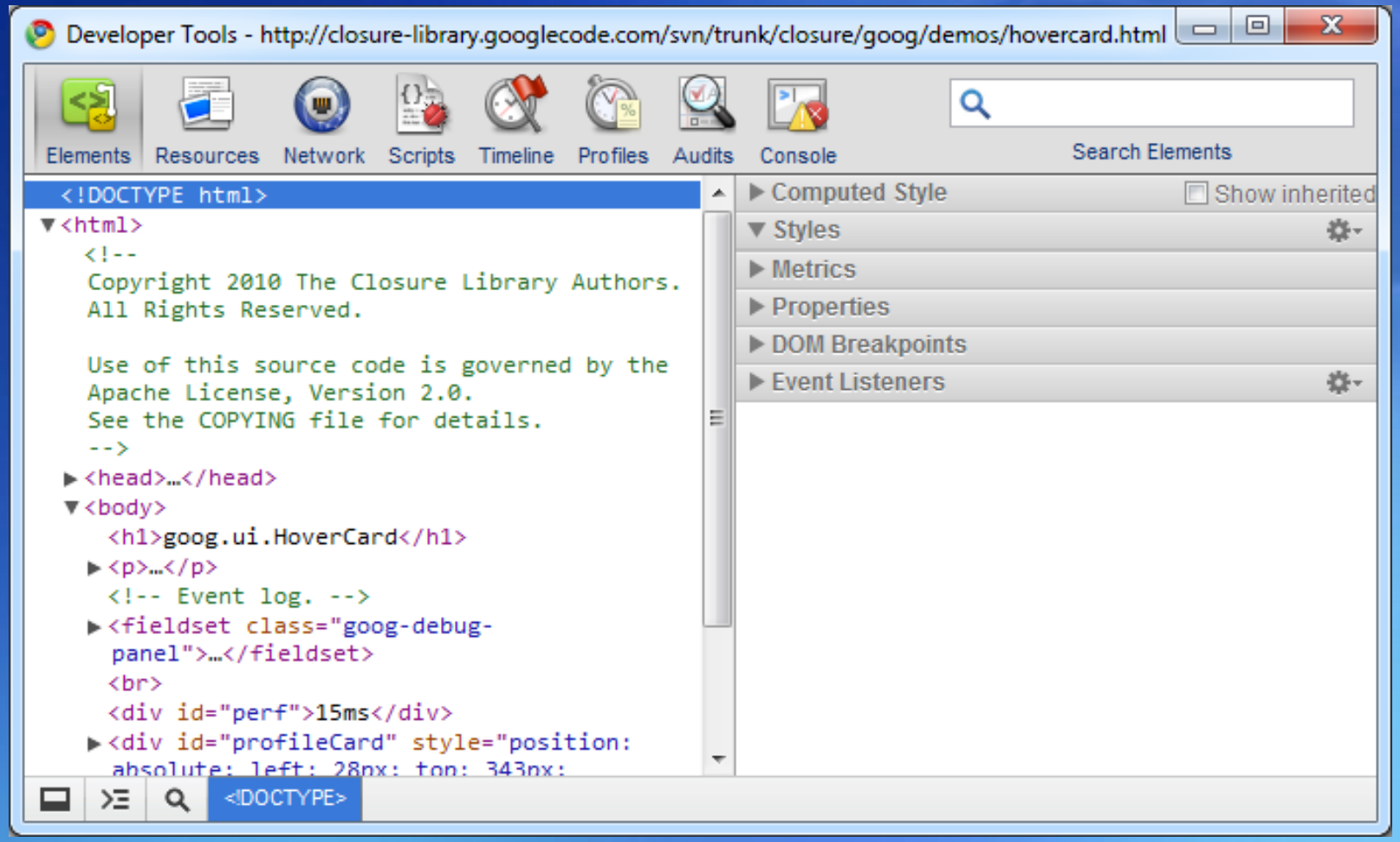
- Are organized into task-oriented groups.
- Are represented by icons in the toolbar at the top of the window.



- Lets you work with a specific type of page or app information, including DOM elements, resources, and scripts.
- Provides a search field that enables you to search the current panel.

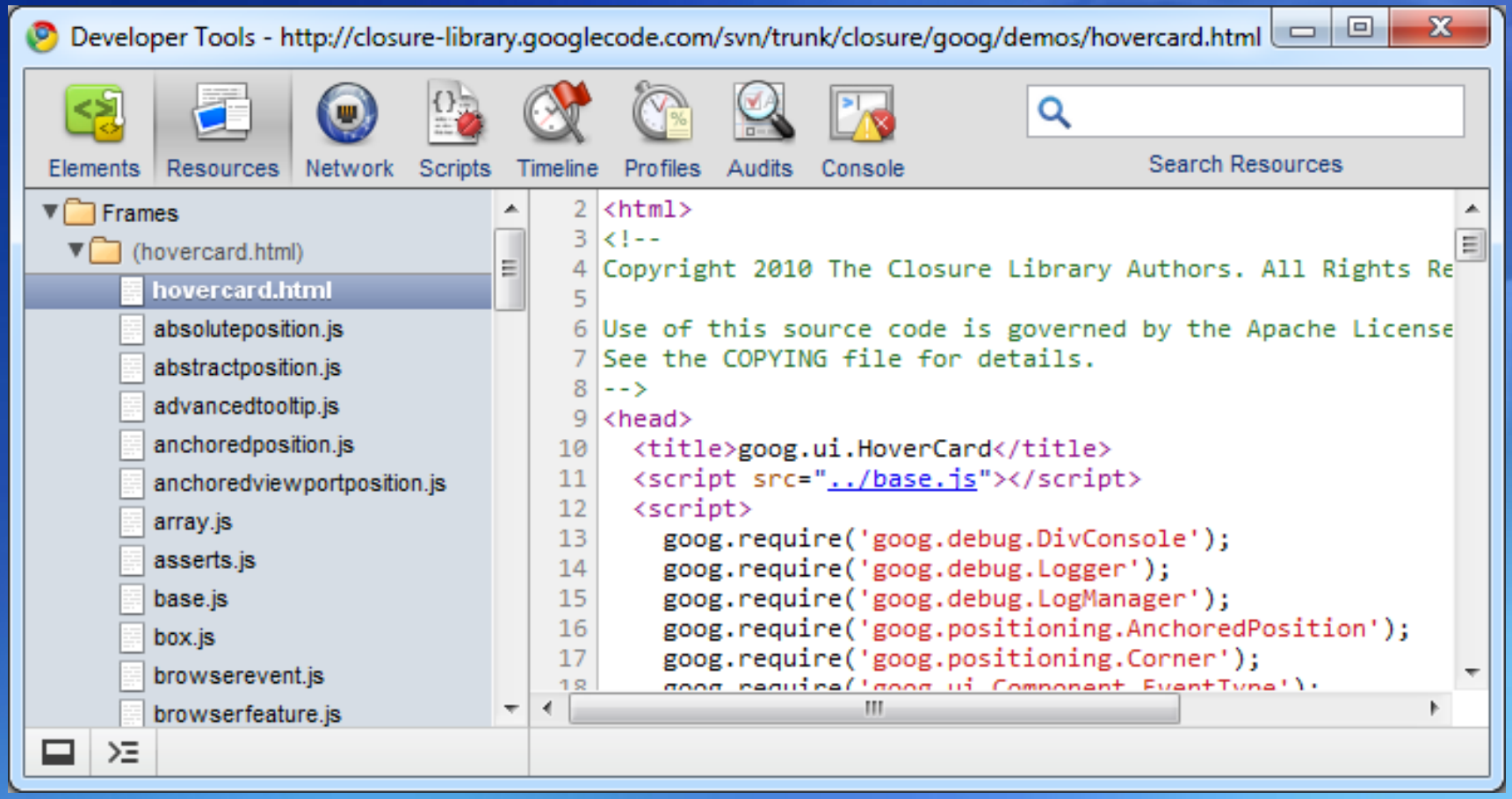
Chrome Developer Tools (Cont.)

- Elements Panel



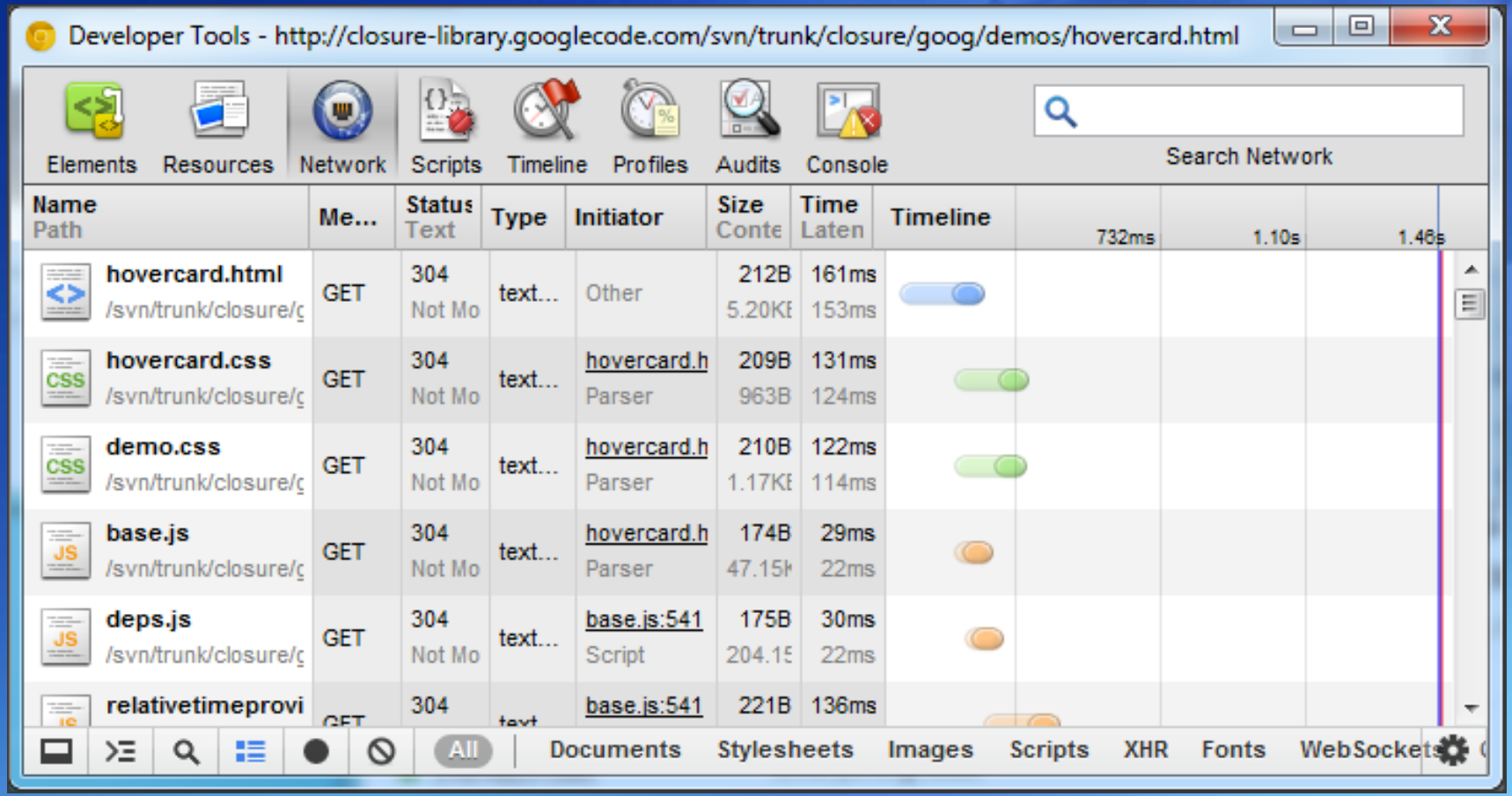
Chrome Developer Tools (Cont.)

- Resources Panel



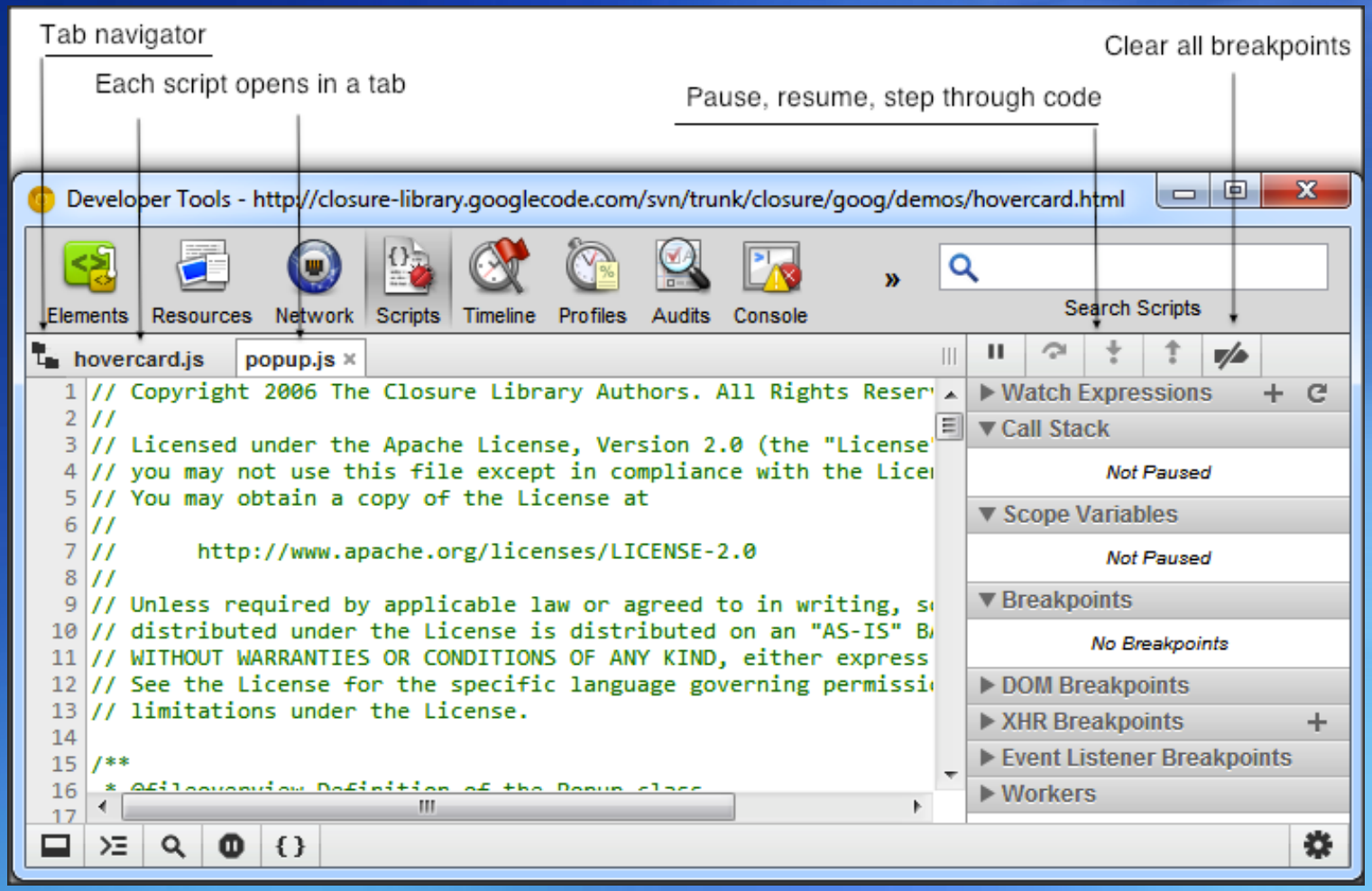
Chrome Developer Tools (Cont.)

- Network Panel



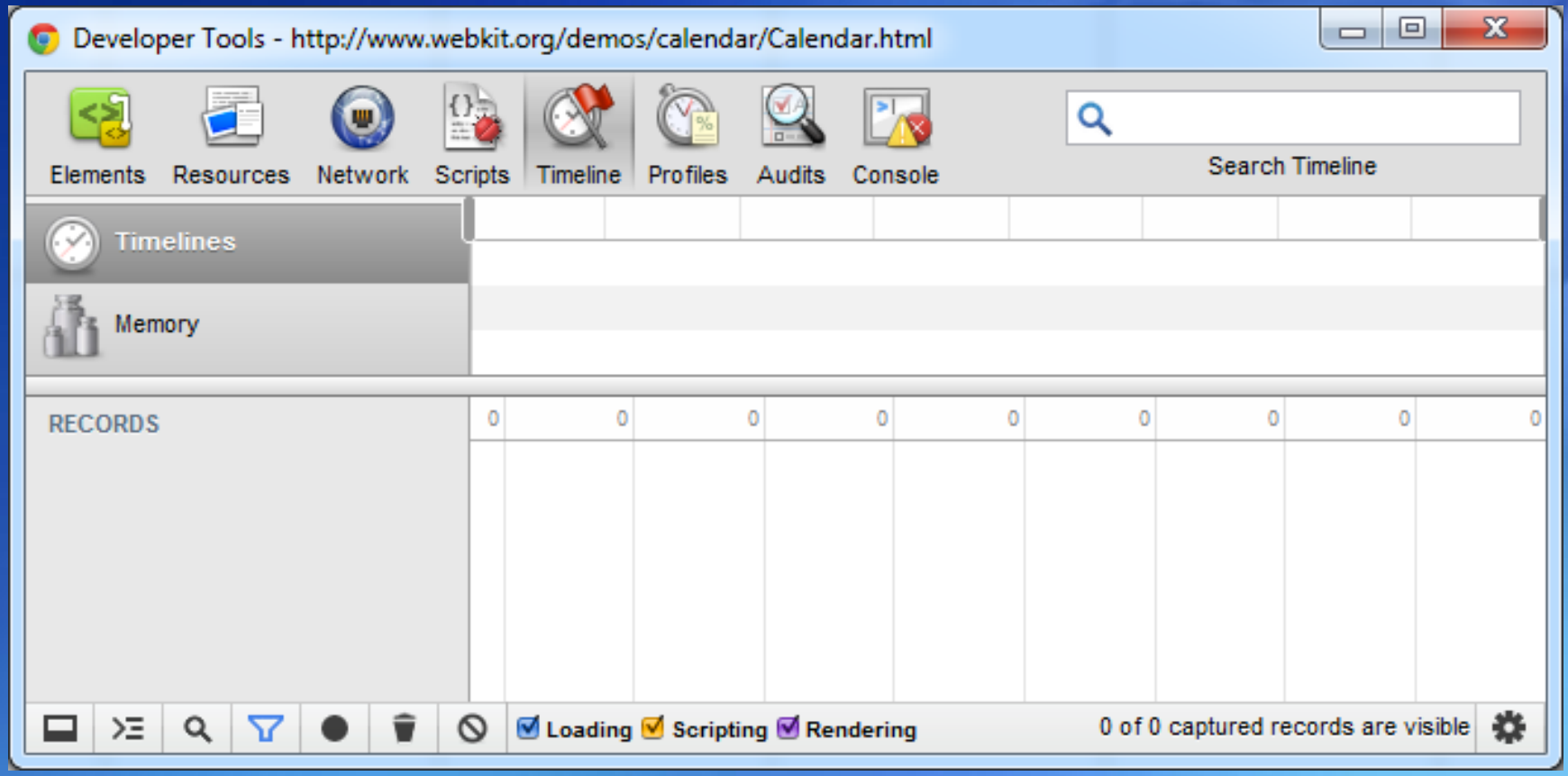
Chrome Developer Tools (Cont.)

- Scripts Panel



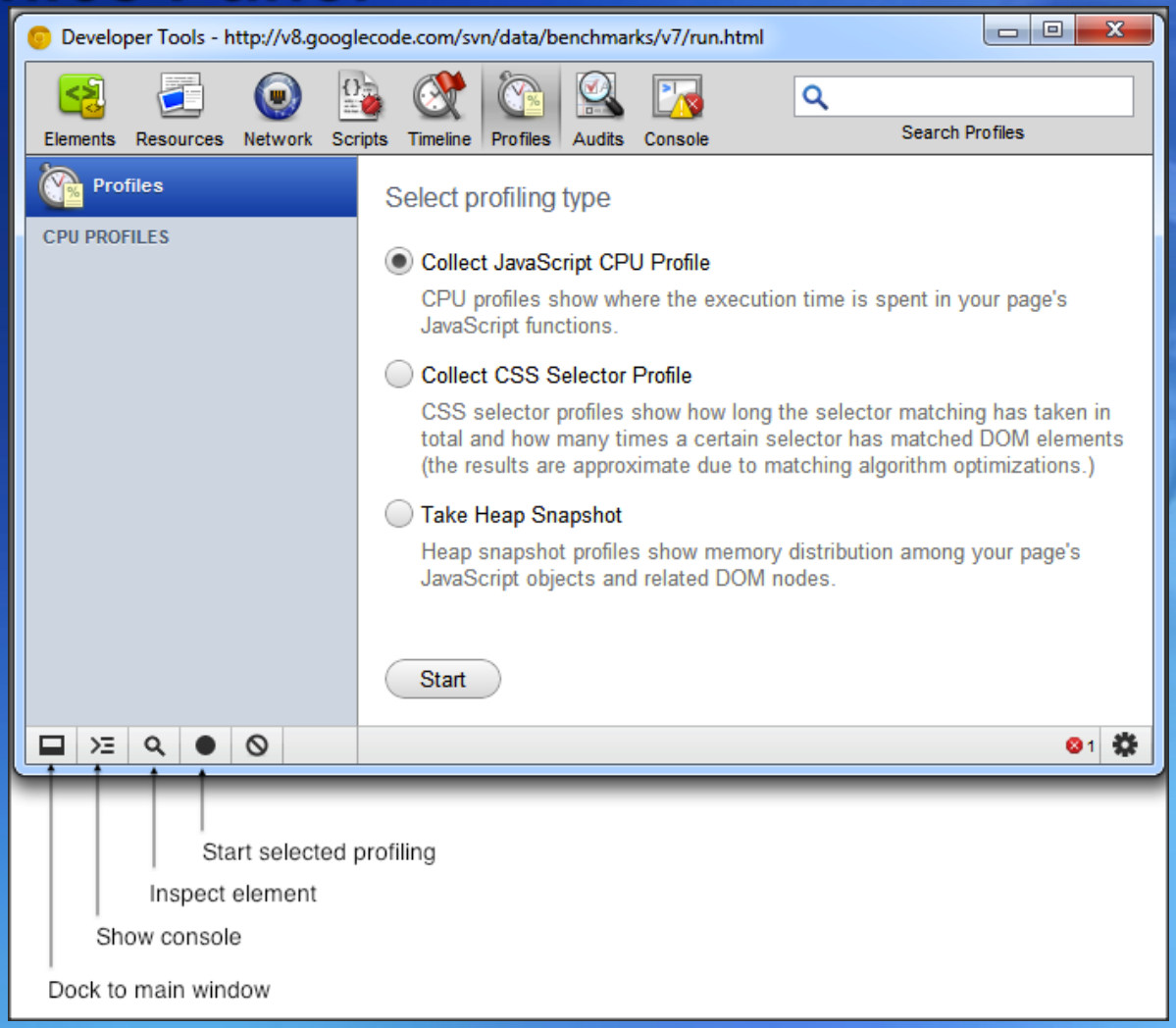
Chrome Developer Tools (Cont.)

- Timeline Panel



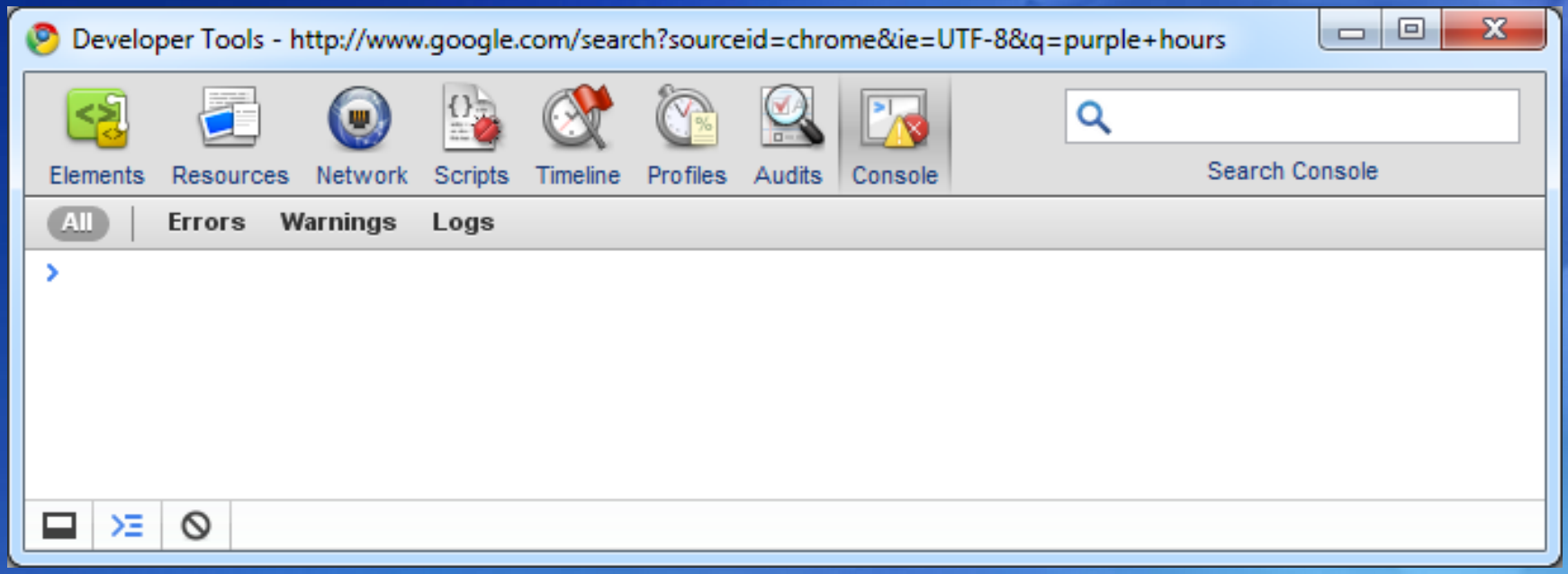
Chrome Developer Tools (Cont.)

- Profiles Panel

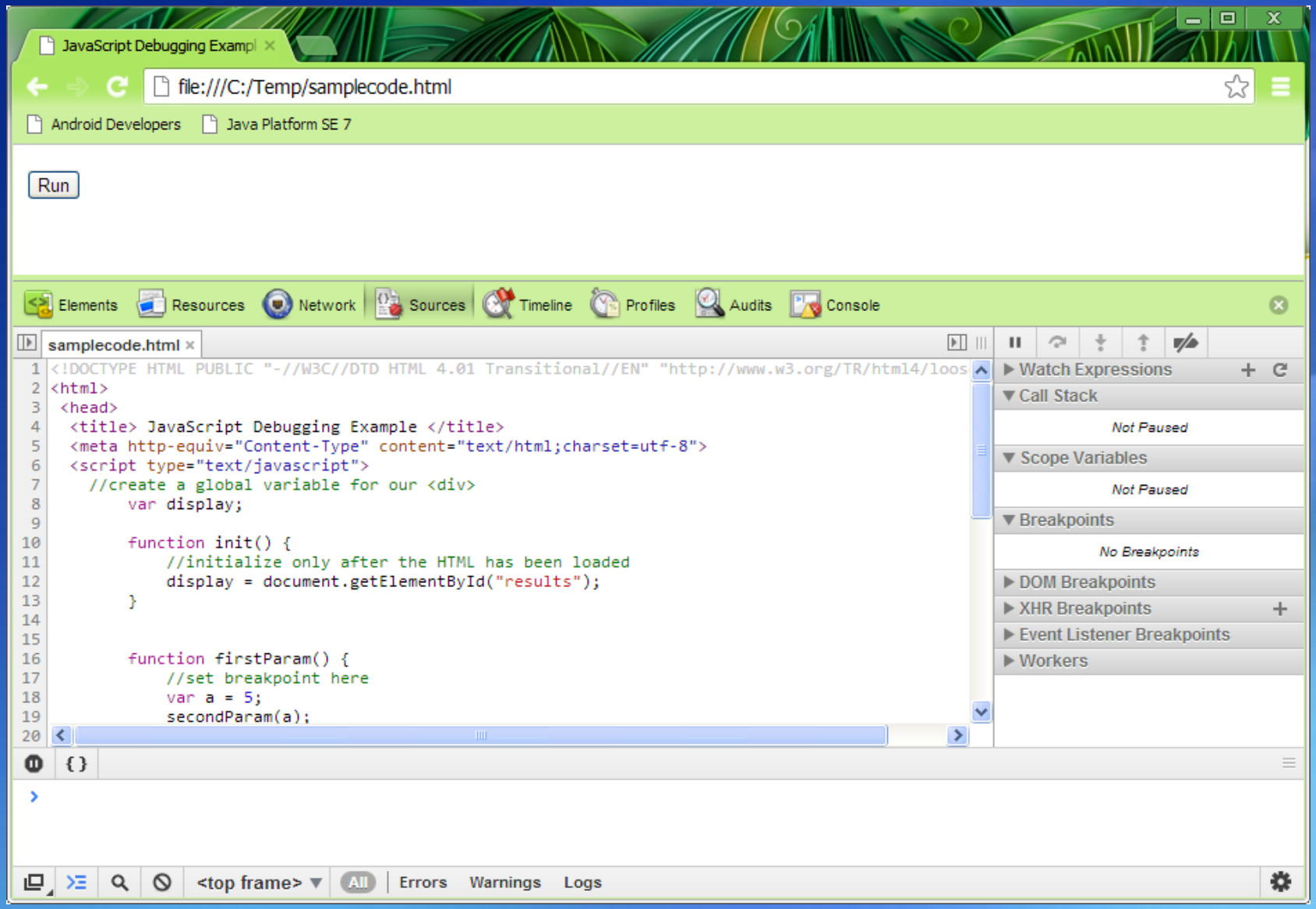


Chrome Developer Tools (Cont.)

- Console Panel



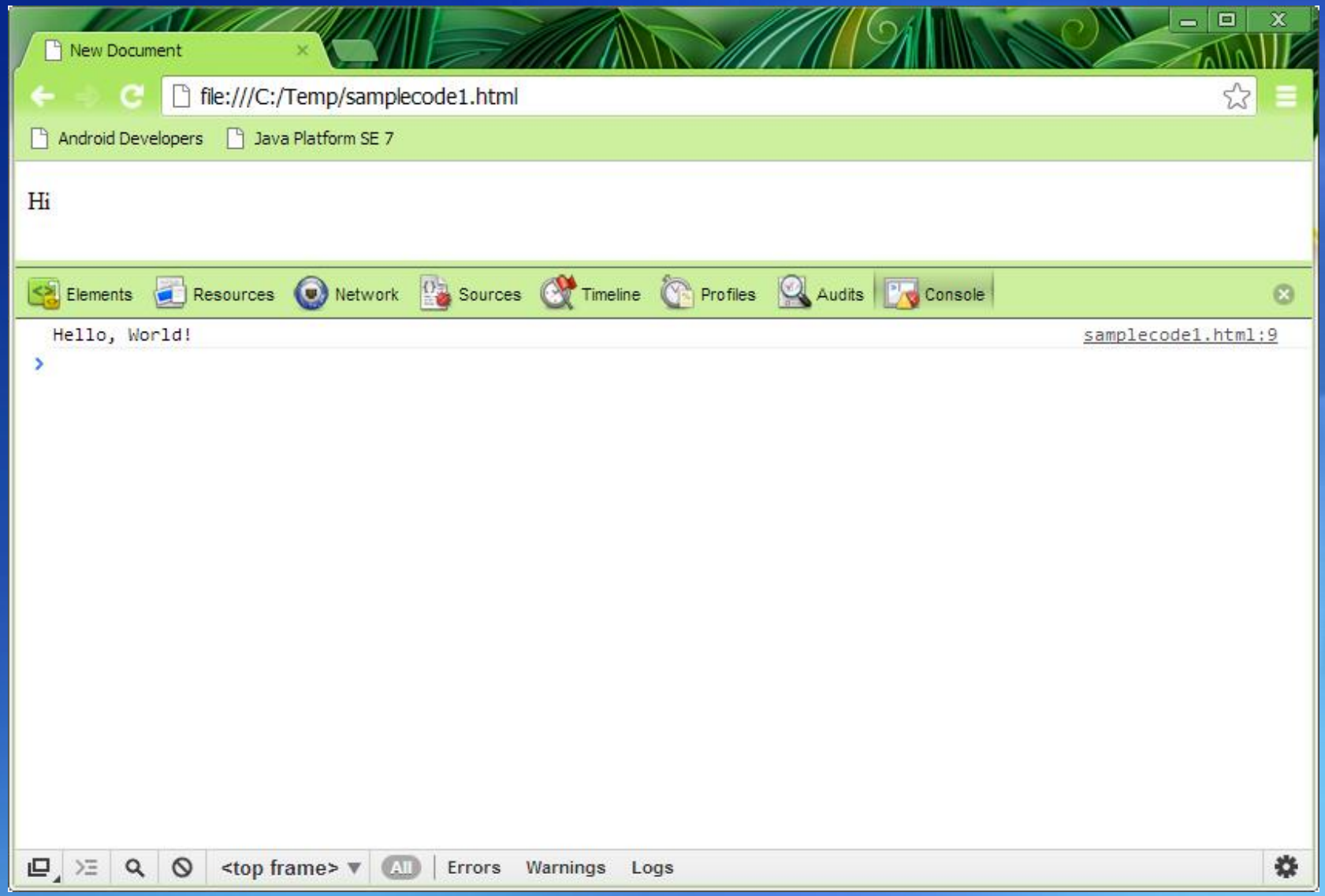
Debugging Test for Chrome Developer Tools



Debugging Test for Chrome Developer Tools (Cont.)

1. In Chrome Browser, load the example.
2. Press `Ctrl + Shift + I` to open the `Chrome Developer` tools, and click the `Sources` tab.
3. In the bottom pane, scroll to the first function, click the line that says "`var a = 5;`", and click `Breakpoints` tab in the right pane.
4. Click `F8` to process `Pause script execution`, and then click `Refresh` in the browser toolbar, and then click `F11` to process `Step into next function call`.
5. Click `Run` button in the web page, click the `Scope Variables` tab on the right panel.

Using console.log in Chrome

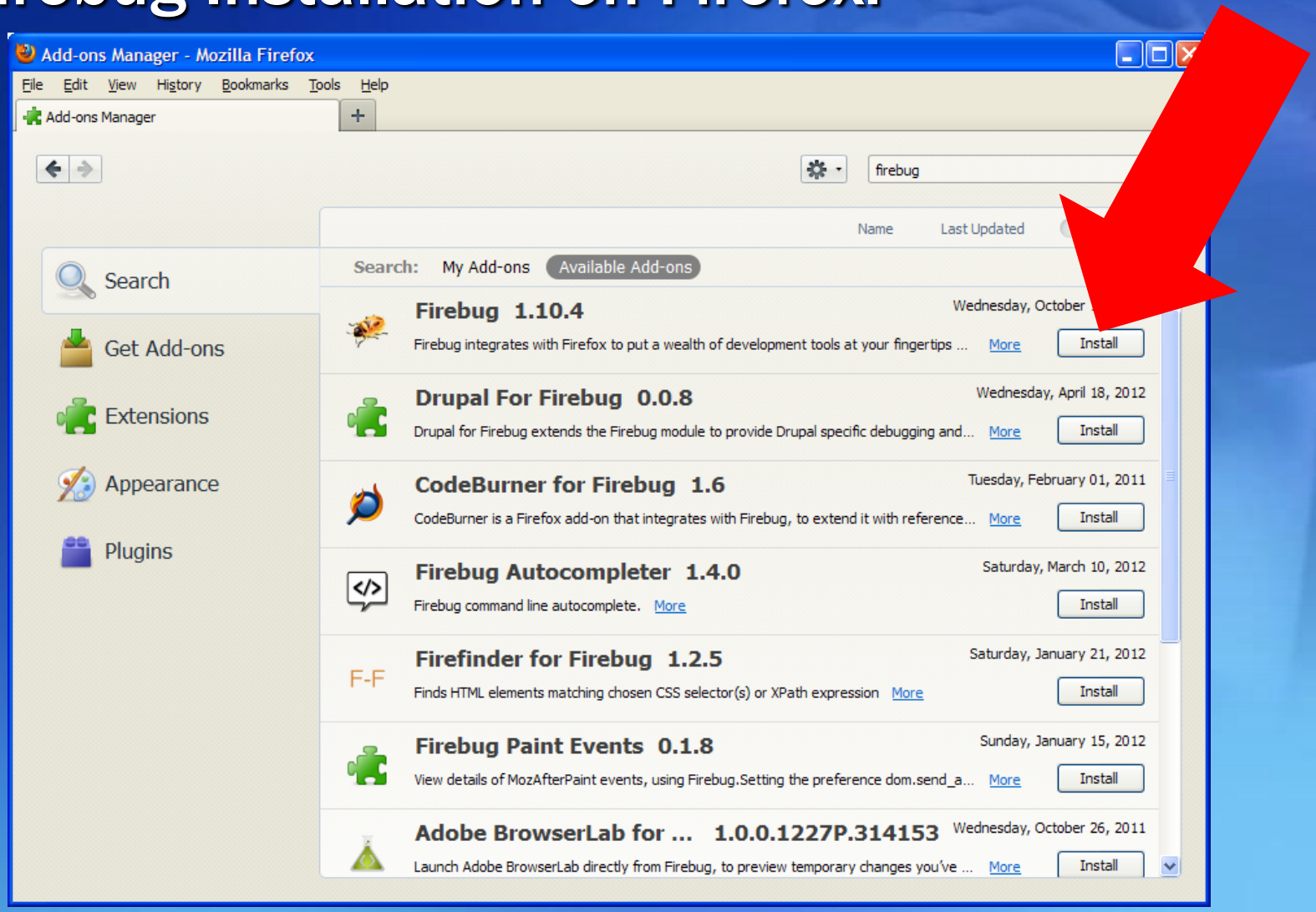


Firefox and Firebug

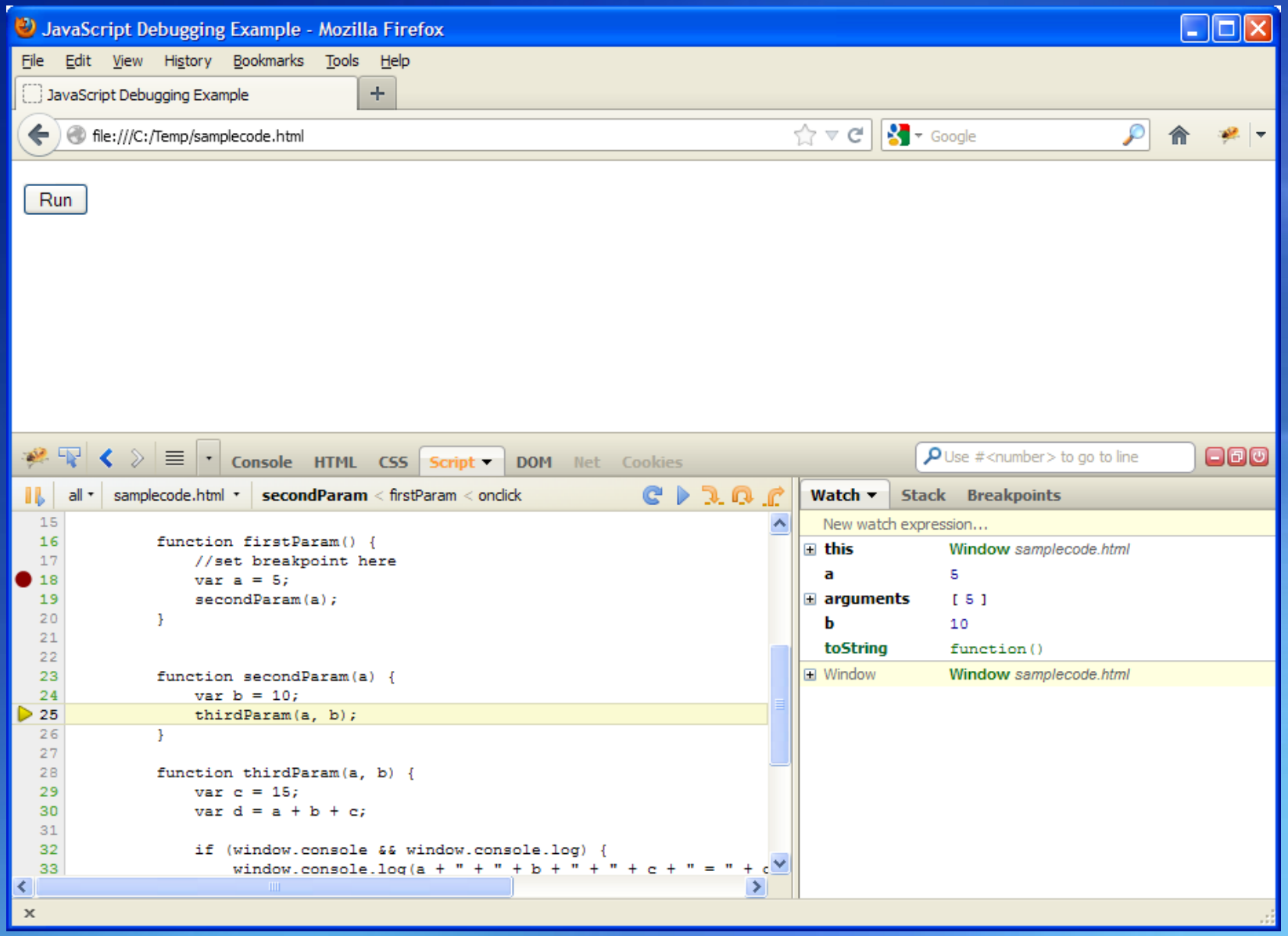
- http://getfirebug.com

Firefox and Firebug (Cont.)

- Firebug Installation on Firefox.



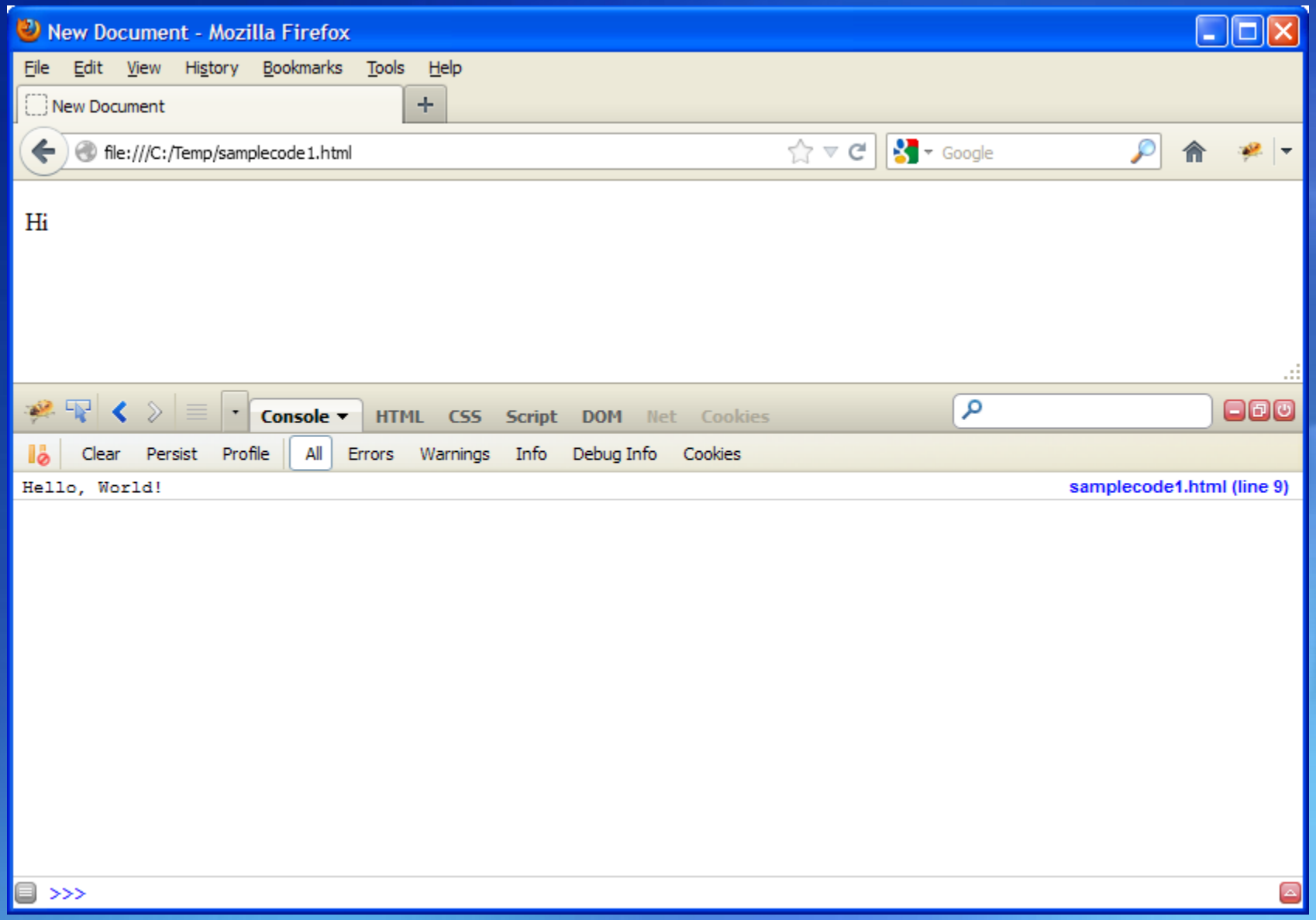
Debugging Test for Firefox Firebug



Debugging Test for Firefox Firebug (Cont.)

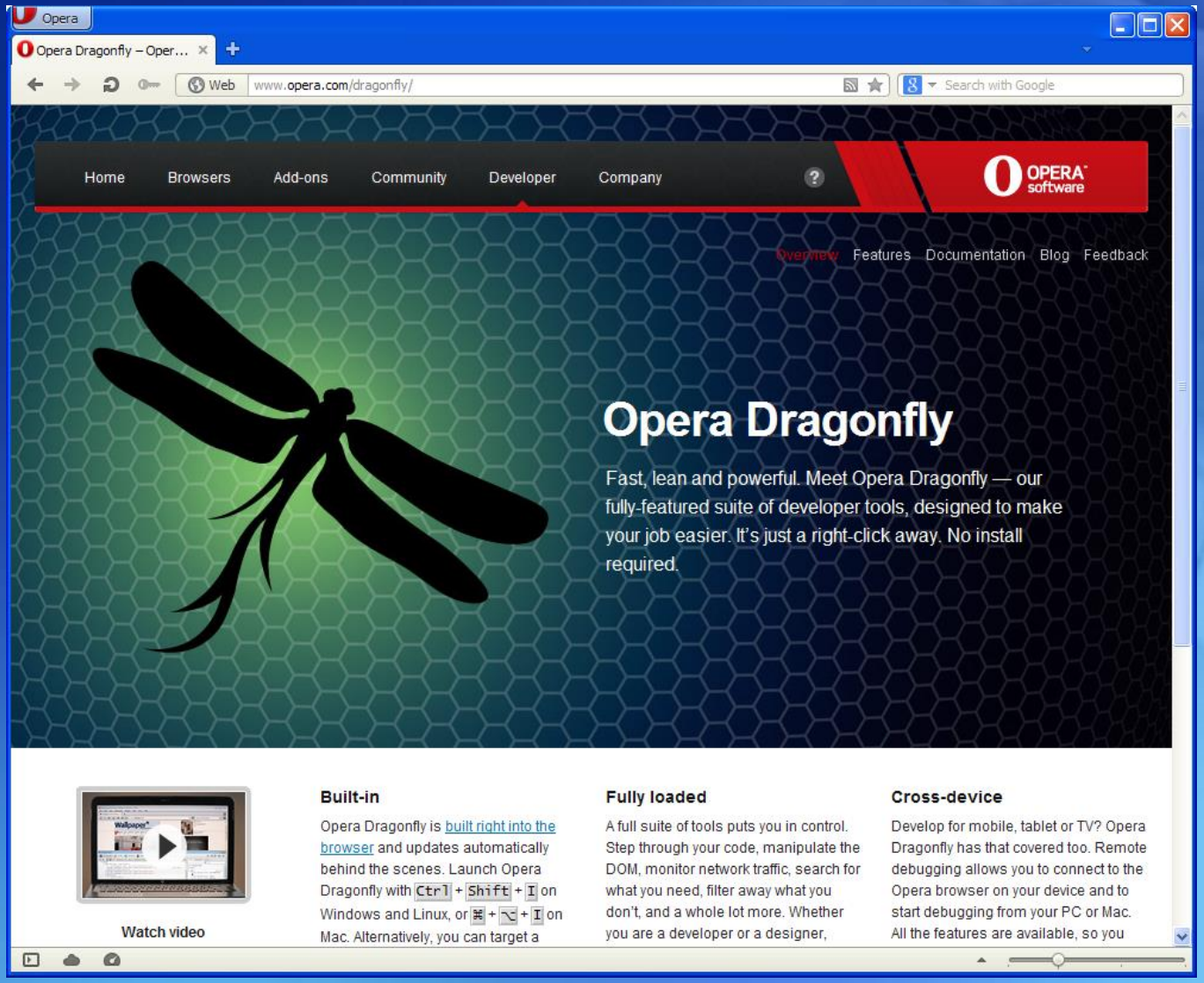
1. In Firefox Browser, load the example.
2. Press `F12` to open the `Open Firebug` tools, and click the `Script` tab.
3. In the bottom pane, scroll to the first function, click the line that says "`var a = 5;`“.
4. Click `Run` button in the web page, click `F11` to process `Step Into`.
5. Click the `Watch` tab on the right panel.

Using console.log in Firefox

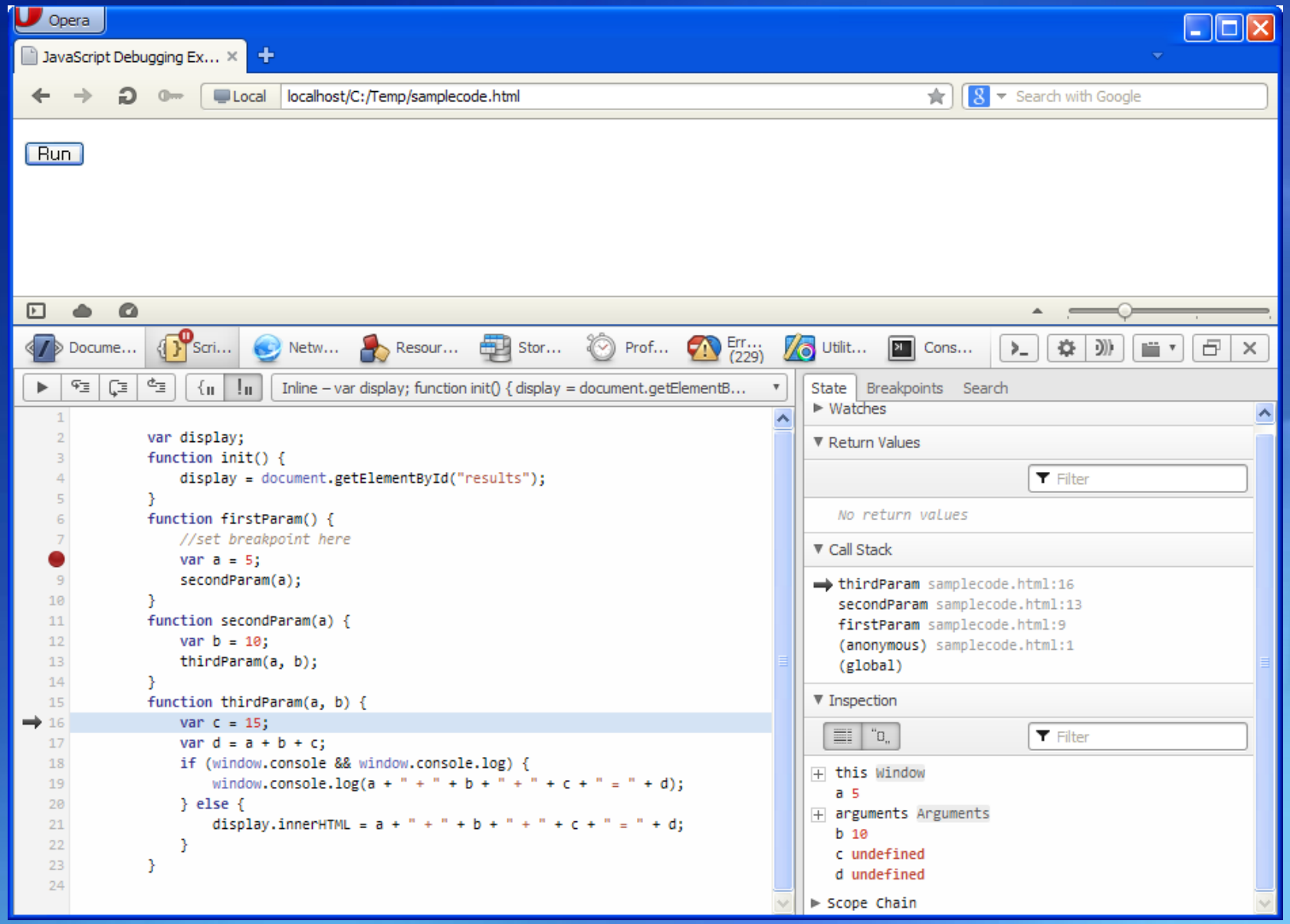


Opera and Dragonfly

- http://www.opera.com/dragonfly/



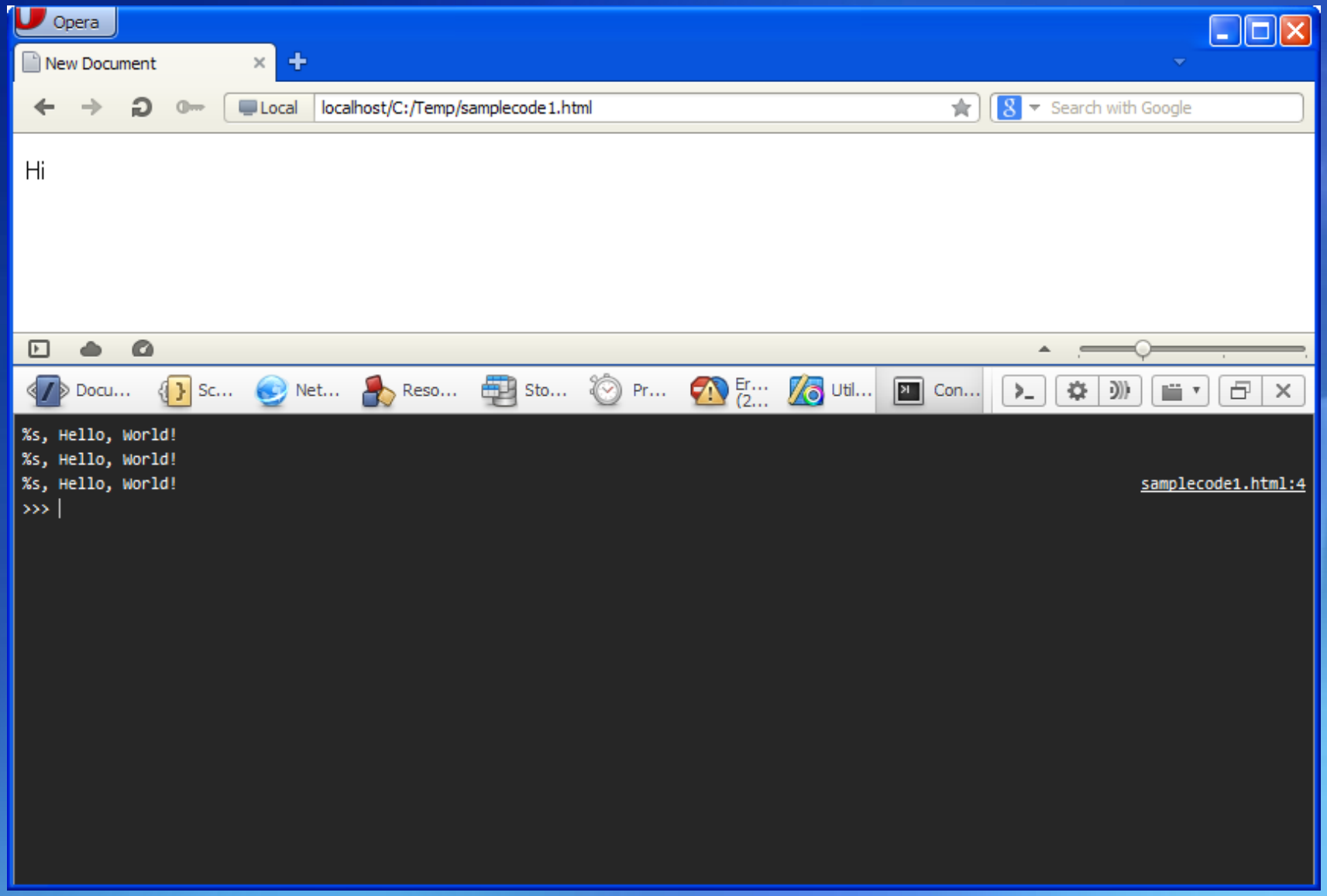
Debugging Test for Opera Dragonfly



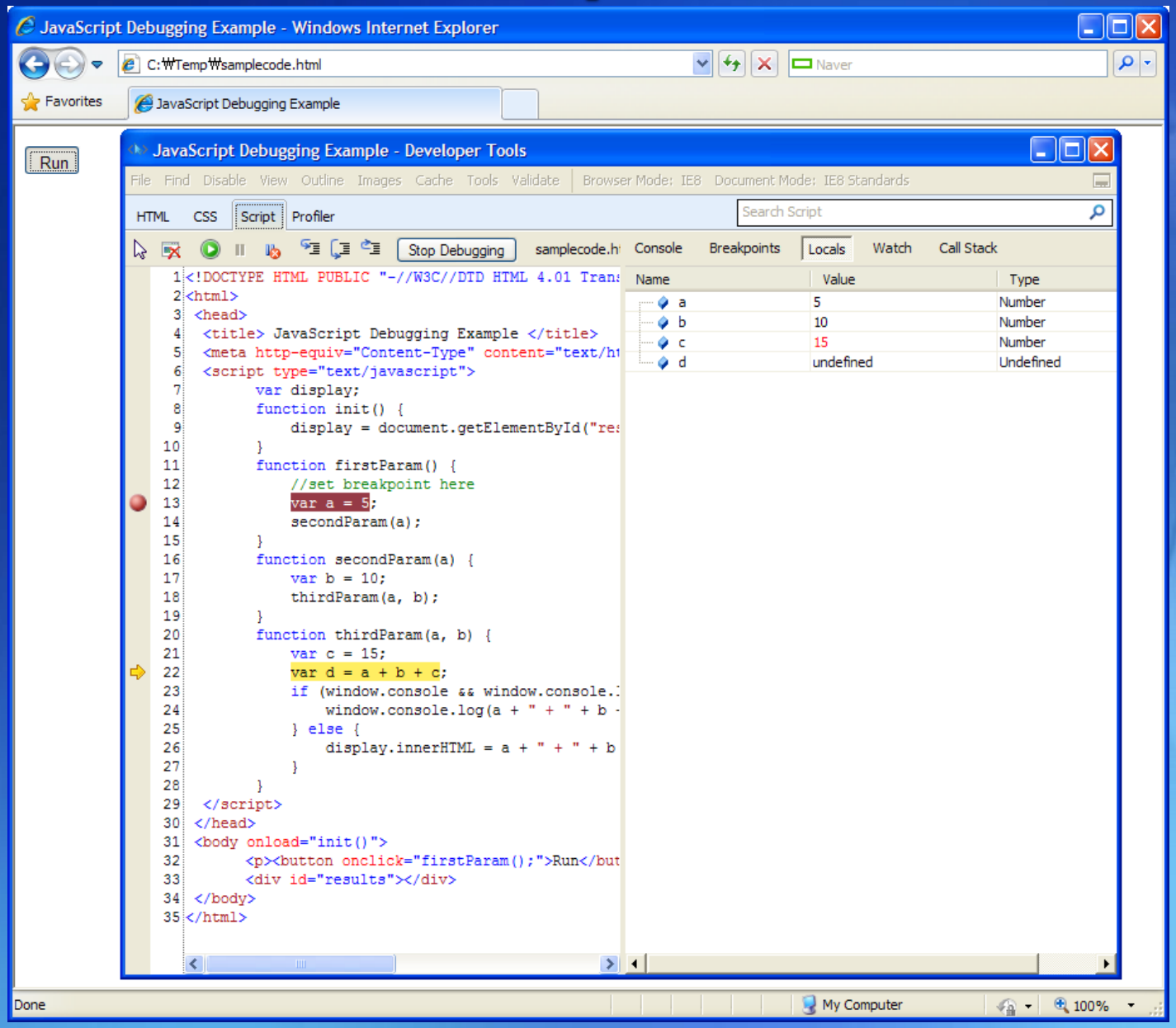
Debugging Test for Opera Dragonfly (Cont.)

1. In Opera Browser, load the example.
2. Press `Ctrl + Shitf + I` to open the `Opera Dragonfly` tools, and click the `Scripts` tab.
3. In the bottom pane, scroll to the first function, click the line that says "`var a = 5;`".
4. Click `Run` button in the web page, click `F11` to process `Step Into`.
5. Click the `Inspecition` tab on the right panel.

Using console.log in Opera



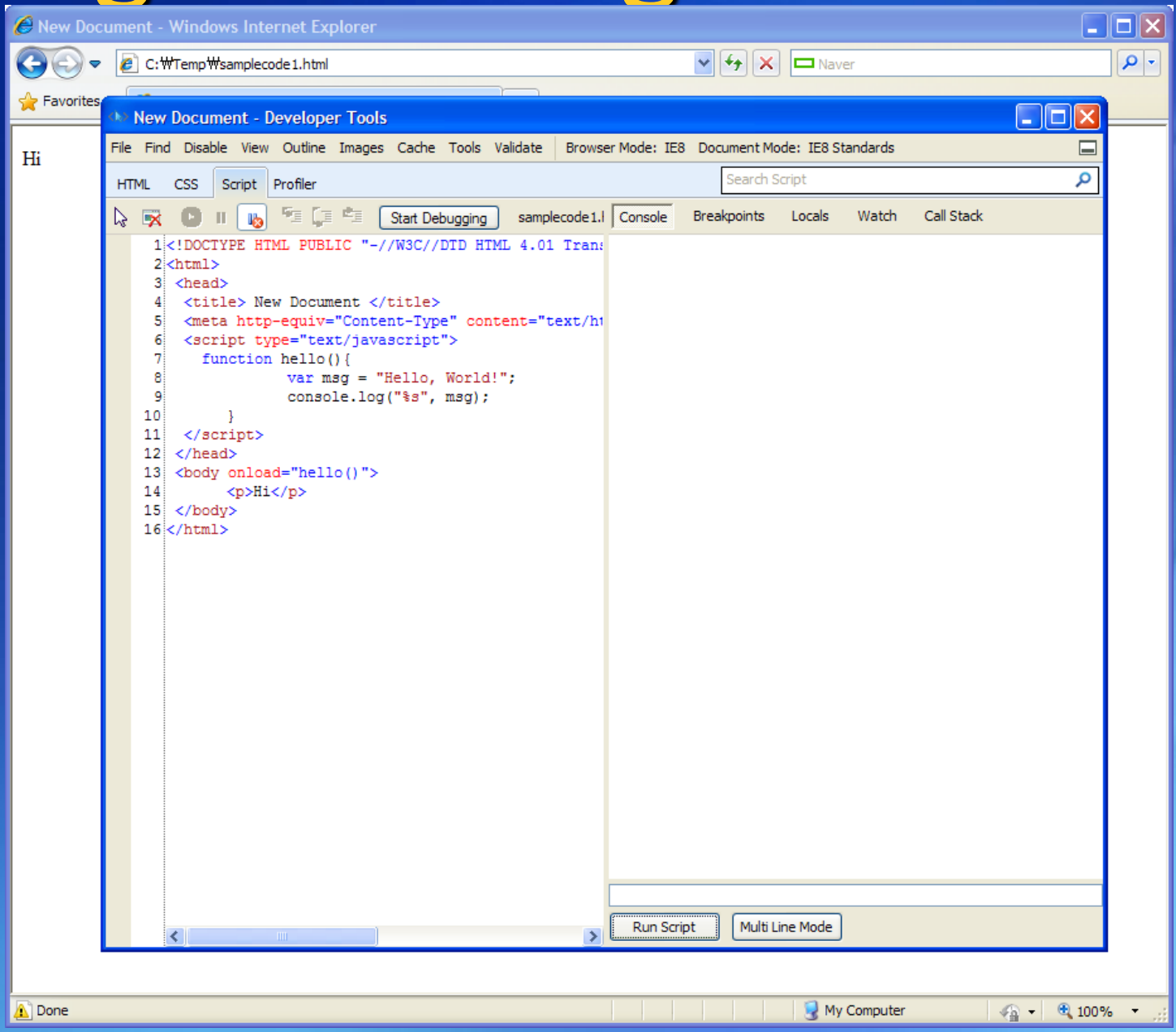
IE's F12 Developer Tools



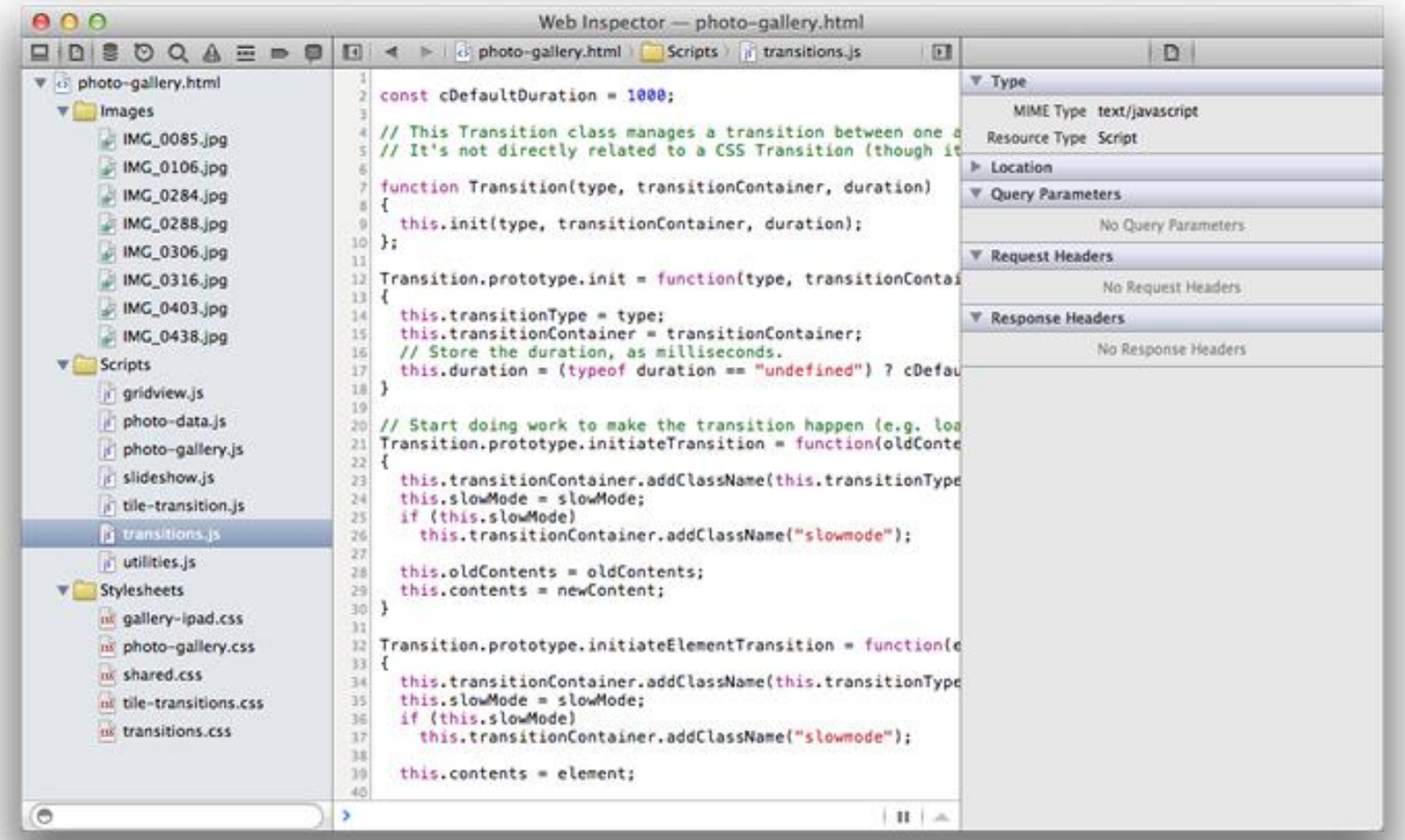
Debugging Test for IE (Cont.)

1. In Internet Explorer 9, load the example.
2. Press **F12** to open the **F12** tools, and click the **Script** tab.
3. In the left pane, scroll to the first function, right-click the line that says "var a = 5;", and click **Insert breakpoint**.
4. Click **Start debugging**, and then on the webpage in the browser, click the **Run** button.
5. In **F12** tools, click the **Watch** tab on the right side, and add the variables "a, b, c, and d.“.
6. Step through your code by pressing **F11**, or by clicking the **Step into** button, and watch the variables on the **Watch** tab.

Using console.log in IE

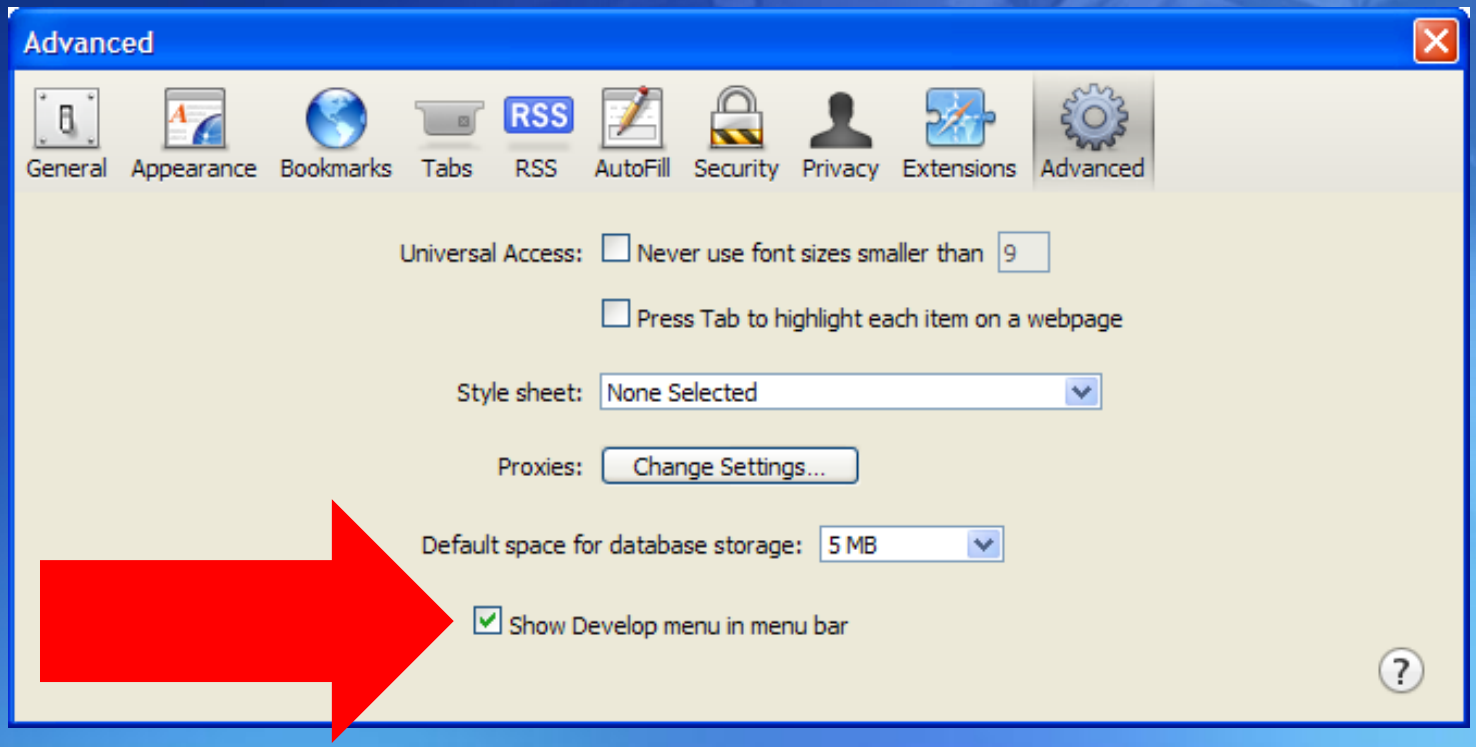


Safari Web Development Tools

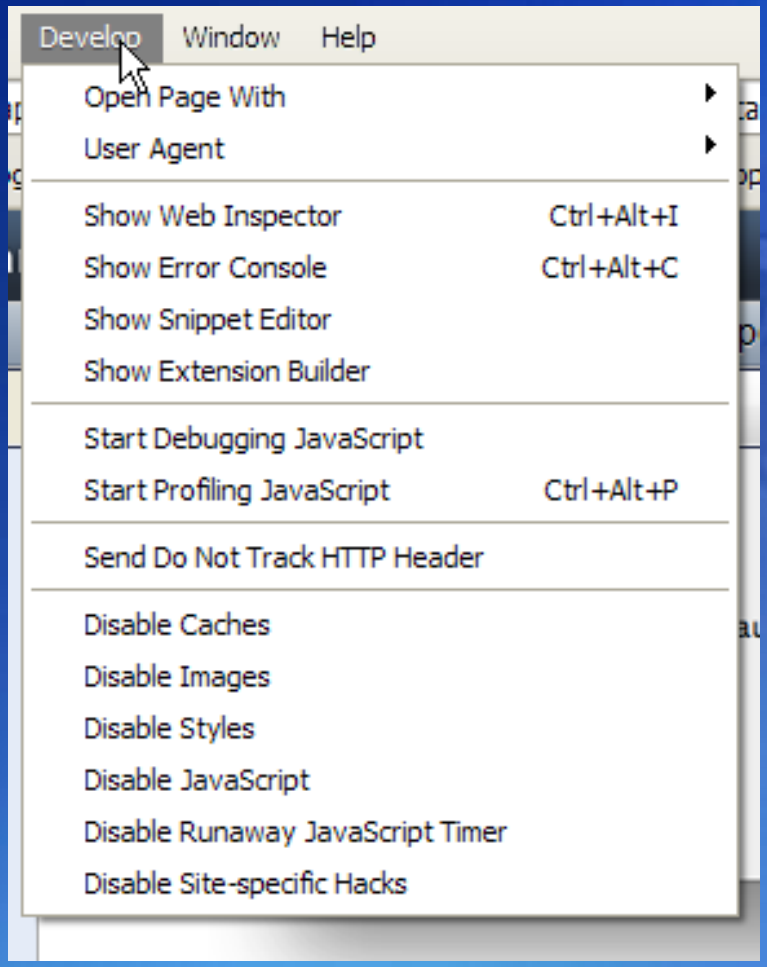


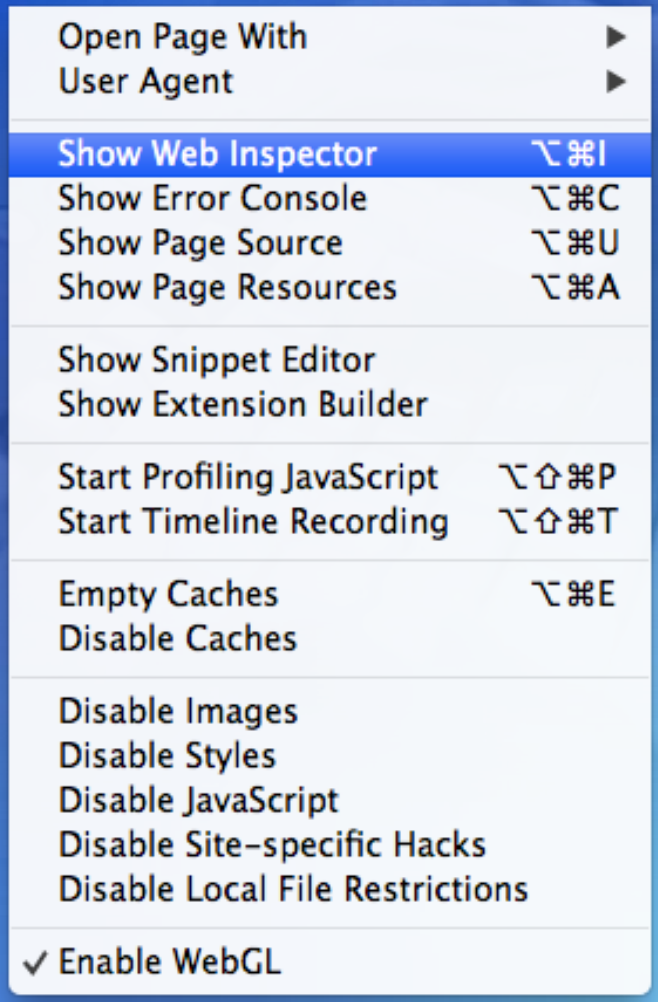
Safari Web Development Tools (Cont.)

1. In Safari `preferences`, click `Advanced`, then select "`Show Develop menu in menu bar`"

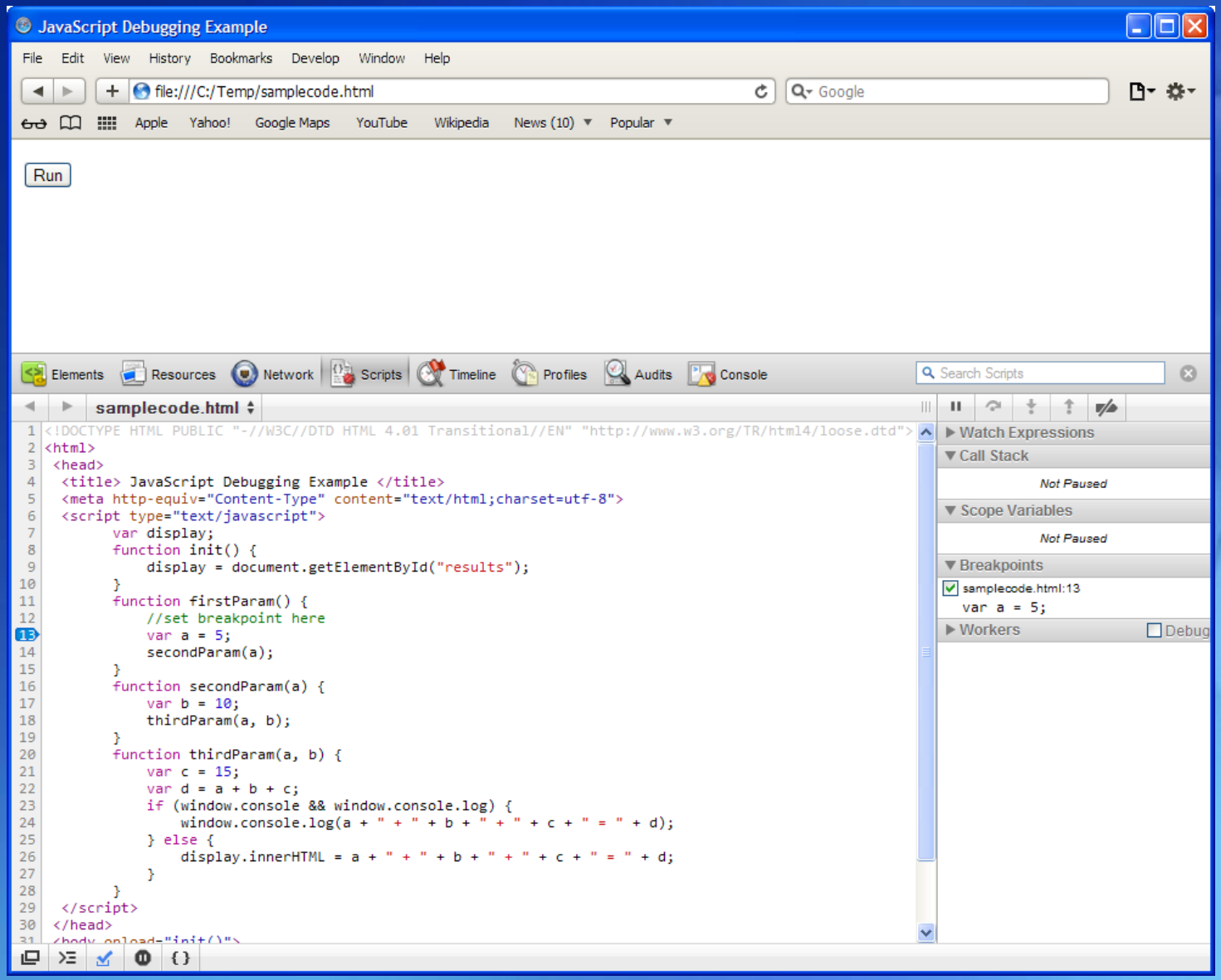


Safari Web Development Tools (Cont.)





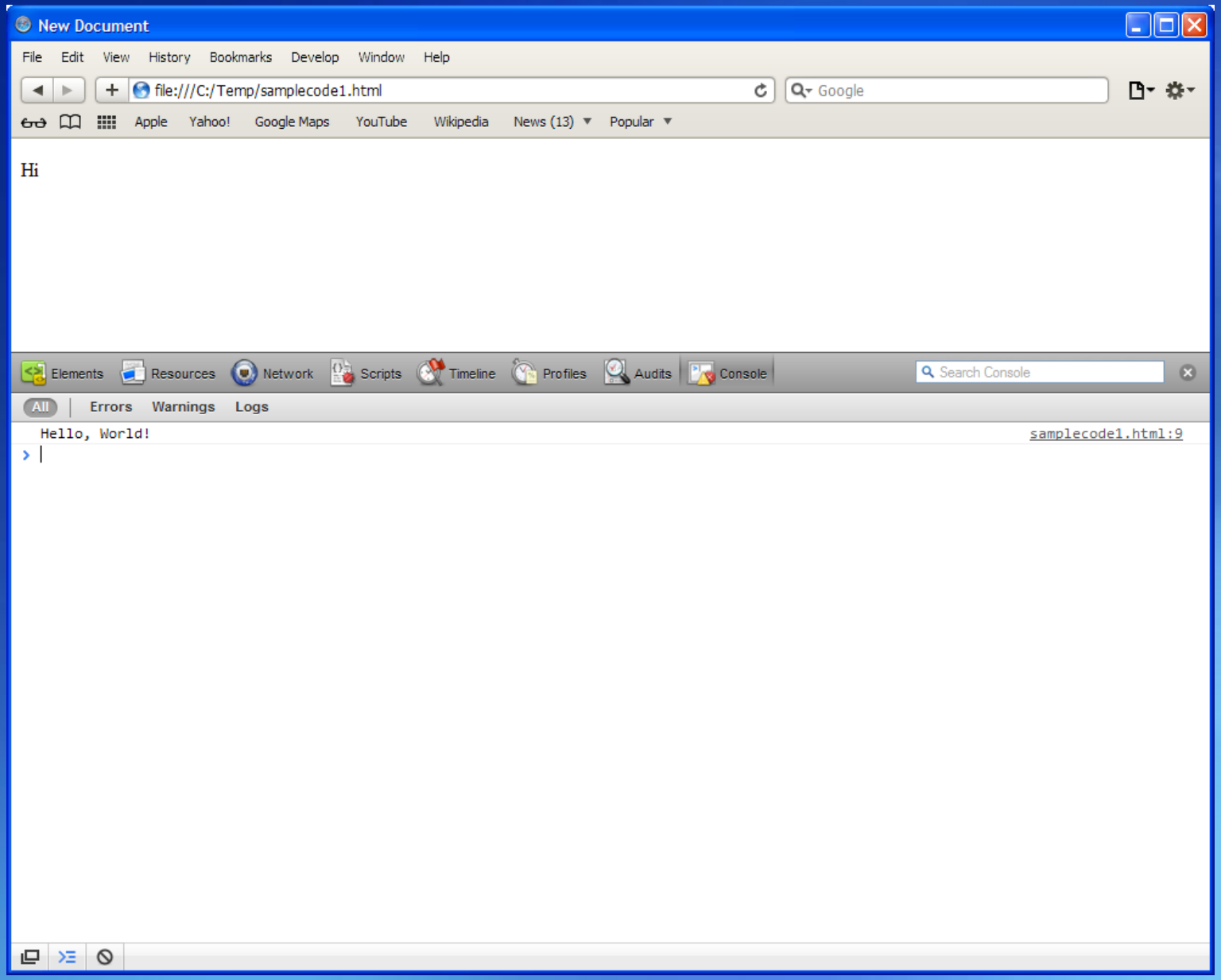
Debugging Test for Safari



Debugging Test for Safari (Cont.)

1. In Safari Browser, load the example.
2. Press `Show Web Inspector` in `Develop` Menu to open the `Safari Development Tool` tools, and click the `Scripts` tab.
3. In the bottom pane, scroll to the first function, click the line that says "`var a = 5;`", and click `Breakpoints` tab in the right pane.
4. Click `Pause script execution` button, and then click `Refresh` in the browser toolbar, and then click `Step into next function call` button on the right panel.
5. Click `Run` button in the web page, click the `Scope Variables` tab on the right panel.

Using console.log in Safari



Dealing with Cross-Browser Differences

- 15 years ago, dealing with cross-browser differences between the two dominant browsers at the time—Internet Explorer and Netscape's Navigator—was a heroic and painful process.
- Differences still exist, and all of the differences aren't just between Firefox-Opera-Safari-Chrome and Internet Explorer.

Object Detection

```
<script type="text/javascript">
   // *** BROWSER VERSION ***
   this.major = parseInt(navigator.appVersion)
   this.minor = parseFloat(navigator.appVersion)
   this.nav = ((agt.indexOf('mozilla')!=-1) && ((agt.indexOf('spoofer')==-1)
                && (agt.indexOf('compatible') == -1)))
   this.nav2 = (this.nav && (this.major == 2))
   this.nav3 = (this.nav && (this.major == 3))
   this.nav4 = (this.nav && (this.major == 4))
   this.nav4up = this.nav && (this.major >= 4)
   this.navonly = (this.nav && (agt.indexOf(";nav") != -1))
   this.ie = (agt.indexOf("msie") != -1)
   this.ie3 = (this.ie && (this.major == 2))
   this.ie4 = (this.ie && (this.major == 4))
   this.ie4up = this.ie && (this.major >= 4)
   this.opera = (agt.indexOf("opera") != -1)
</script>
```

Object Detection (Cont.)

- With object detection, the JavaScript application accesses the object being detected in a conditional statement.
- If the object doesn't exist, the condition evaluates to `false`.
- As an example, the following checks to ensure that the most basic level of object model support is provided in the browser or other user agent.

```
if (document.getElementById)...
```

Object Detection (Cont.)

```
<script type="text/javascript">
   function showBlock(evnt) {
      var theEvent = evnt ? evnt : window.event;
      var theSrc = theEvent.target ? theEvent.target : theEvent.srcElement;
      var itemId = "elements" + theSrc.id.substr(5,1);
      var item = document.getElementById(itemId);
      if (item.style.display=='none') {
         item.style.display='block';
      } else {
         item.style.display='none';
      }
   }
</script>
```

Object Detection (Cont.)

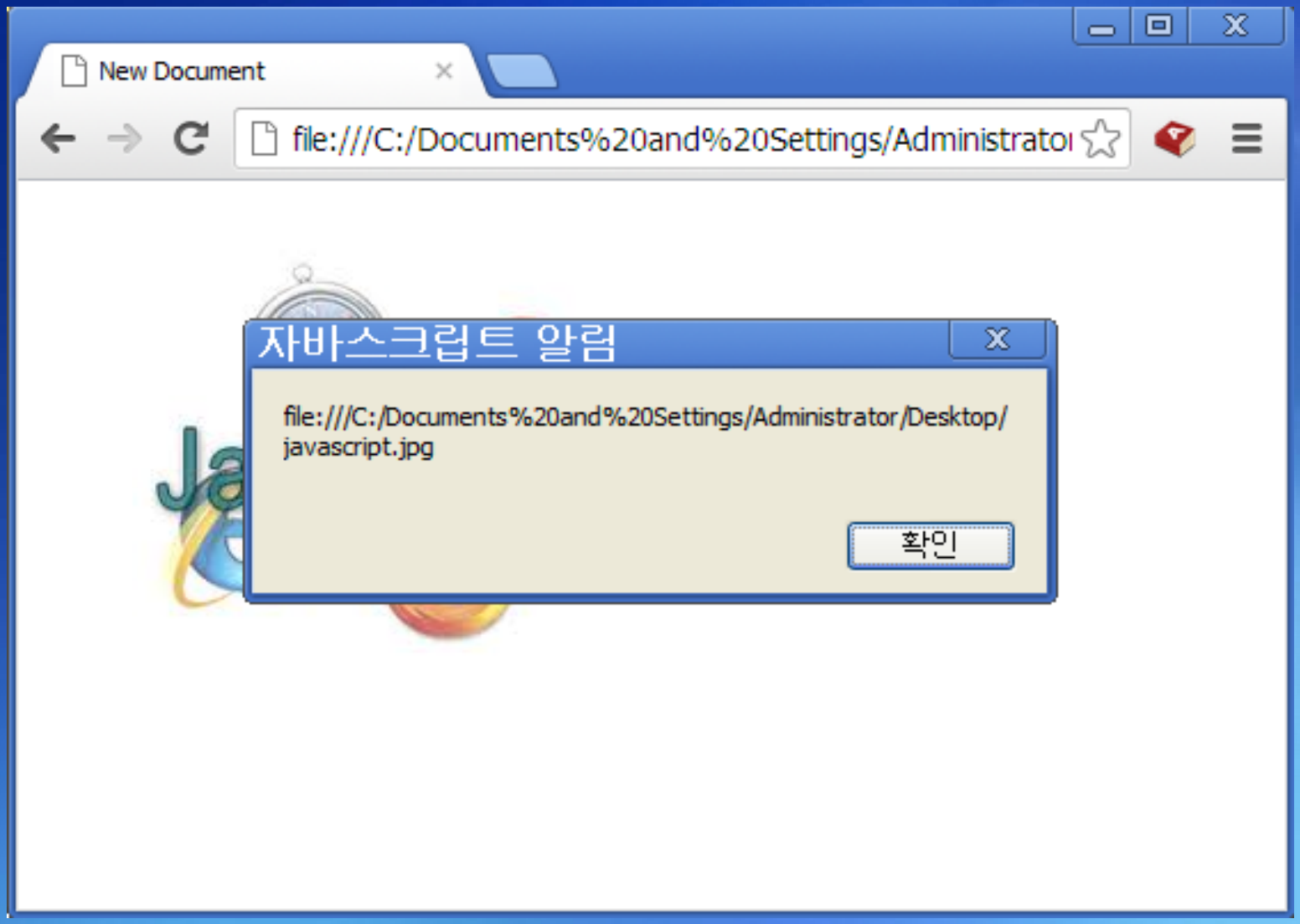
```
<style type="text/css">
    div {
        position: absolute;
        top: 30px;
        left: 50px;
    }
    #div1 img {
        filter:
        progid:DXImageTransform.Microsoft.AlphaImageLoader(src=javascript.jpg,
        sizingMethod='scale');
    }
</style>
```

```
<script type="text/javascript">
    window.onload=function() {
        document.getElementById("div1").onclick=getSrc;
    }
    function getSrc(evnt) {
        var theEvent = evnt ? evnt : window.event;
        var theSrc = theEvent.target ? theEvent.target : theEvent.srcElement;
        alert(theSrc.src);
    }
</script>
```

```
<body>
  <div id="div1">
  <img src="javascript.jpg" alt="" />
  </div>

</body>
```

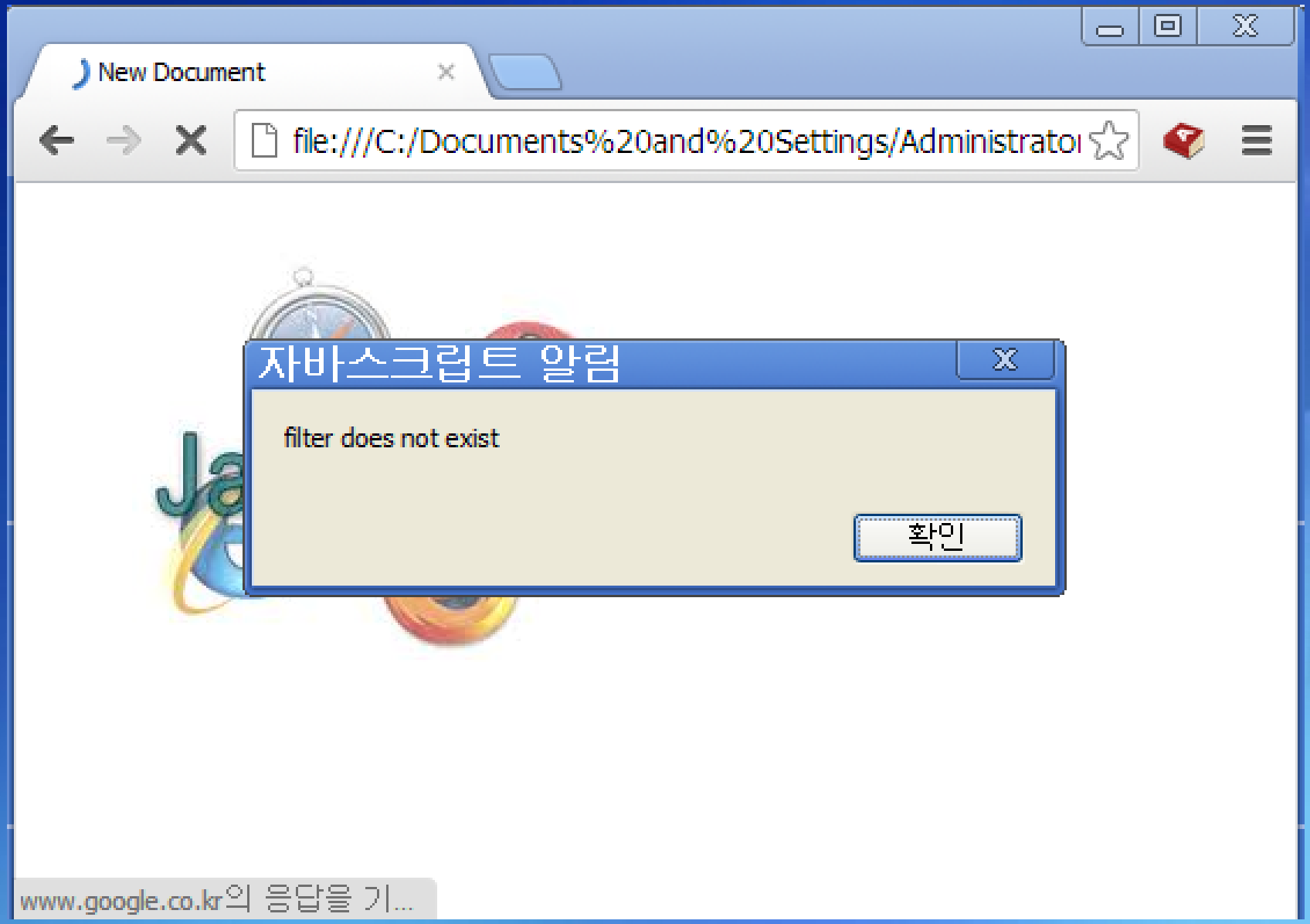
Object Detection (Cont.)



Object Detection (Cont.)

```
<script type="text/javascript">
 window.onload=function() {
    var divElement = document.getElementById("div1");
    var tst1 = (divElement.style.filter) ? "filter exists" : "filter does not exist";
    alert(tst1);
    var tst2 = (typeof(divElement.style.filter) !== "undefined") ? "filter exists" : "filter does
    not exist";
    alert(tst2);
    var tst3 = (divElement.style.filter !== undefined) ? "filter exists" : "filter does not exist";
    alert(tst3);
    var tst4 = (divElement.style.opacity) ? "opacity exists" : "opacity does not exist";
    alert(tst4);
    var tst5 = (typeof(divElement.style.opacity) !== "undefined") ? "opacity exists" :
    "opacity does not exist";
    alert(tst5);
 }
</script>
```

Object Detection (Cont.)



Test Your Knowledge : Quiz

1. Write the code to check to see what the value of a variable is, without recourse to browser-specific debugger.

Test Your Knowledge : Quiz

1. Write the code to check to see what the value of a variable is, without recourse to browser-specific debugger.

```
// test some variable
alert(firstName);
// value will either be set, null, or undefined
```

Test Your Knowledge : Quiz (Cont.)

2. The CSS attribute text-shadow has an interesting history. It was added to CSS2,but no browser implemented it, and it was removed in CSS 2.1. However, then browsers implemented it, and now it's being added into CSS3. At the time of this writing, this attribute was implemented in two of our four target browsers. Can you use object detection to successfully test for this style attribute? Write cross browser code that sets this CSS attribute for an existing header element.

Test Your Knowledge : Quiz (Cont.)

2. The CSS attribute text-shadow has an interesting history. It was added to CSS2,but no browser implemented it, and it was removed in CSS 2.1. However, then browsers implemented it, and now it's being added into CSS3. At the time of this writing, this attribute was implemented in two of our four target browsers. Can you use object detection to successfully test for this style attribute? Write cross browser code that sets this CSS attribute for an existing header element.

```
var headerElement = document.getElementById("pageHeader");
headerElement.style.textShadow="#ff0000 2px 2px 3px";
```