

Debugging JavaScript Code

CSE 190 M (Web Programming), Spring 2008
University of Washington

References: Dr. Dobb's

Except where otherwise noted, the contents of this presentation are © Copyright 2008 Marty Stepp and Jessica Miller and are licensed under the Creative Commons Attribution 2.5 License.



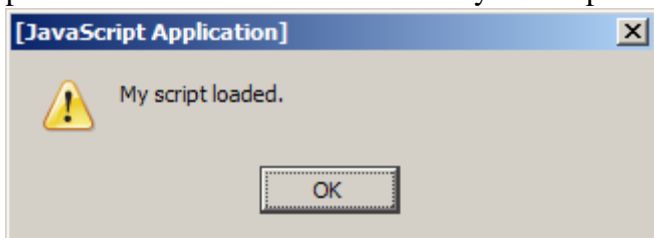
"My program does nothing"


Since Javascript has no compiler, many errors will cause your Javascript program to just "do nothing." Some questions you should ask when this happens:

- Is the browser even loading my script file?
- If so, is it reaching the part of the file that I want it to reach?
- If so, what is it doing once it gets there?

Is my JS file loading?

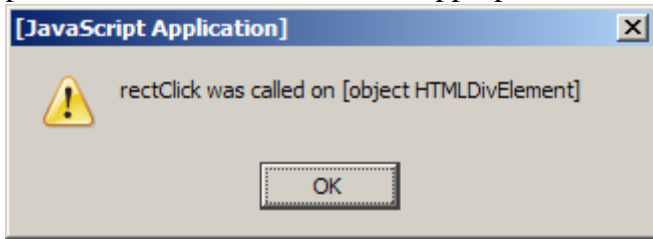
- put an `alert` at the VERY TOP of your script:



- if it shows up, good!
- if it doesn't show up:
 - maybe your HTML file isn't linking to the script properly
 - double-check file names and directories
 - maybe your script has a syntax error
 - check bottom-right for Firebug error text 
 - comment out the rest of your script and try it again
 - run your script through JSLint to find some syntax problems

Is it reaching the code I want it to reach?

- put an `alert` at the start of the appropriate function:



- write a descriptive message, not just "hello" or "here"
- if it shows up, good!
- if it doesn't show up:
 - if it's an event handler, maybe you didn't attach it properly; check the code to attach the handler
 - maybe your script has a syntax error; run JSLint

COMMON ERROR: 'foo' has no properties

- If you see the common "has no properties" error, it means you are trying to utilize an undefined value
- some possible causes:
 - you're trying to access a variable that is out of scope
 - you're accessing a DOM element with `$` with an invalid id
 - you've run off the bounds of an array
 - you've spelled the variable's name incorrectly

COMMON BUG: spelling error

```
window.onload = initalizeBody; // spelled wrong
...
function initializeBody() {
  ...
}
```

JS

- if you misspell an identifier, the value `undefined` is used
- if you set `undefined` as an event handler, nothing happens (fails silently)
- **Manifestation of bug:** function doesn't get called, or a value is unexpectedly `undefined`
- **Fix:** JSLint warns you if you use an undeclared identifier

COMMON BUG: bracket mismatches

```
function foo() {  
  ... // missing closing curly brace!  
  
function bar() {  
  ...  
}
```

JS

- JS unfortunately doesn't always tell us when we have too many / too few brackets in our JS code
 - unfortunately this is legal in JavaScript, to declare one function inside another
- **Manifestation of bug:** script often becomes (fully or partially) non-functional
- **Detection:** bracket matching in TextPad (highlight bracket, press Ctrl-M)
- **Detection:** using an Indenter tool can highlight such problems (second function will be unexpectedly indented)
- **Detection:** JSLint sometimes catches this

COMMON BUG: misuse of `.style`

```
var theDiv = document.getElementById("puzzlearea");  
theDiv.left = "100px"; // BAD!  
theDiv.style.onclick = myClickFunction; // BAD!
```

JS

- DOM objects have internal `style` object that represents CSS styles
 - setting styles: `object.style.property = value;`
- the DOM objects themselves also have properties of their own
 - setting DOM properties: `object.property = value;`
- **Manifestation of bug:** "I set the property, but it didn't do anything."
- **Fix:** JSLint now tries to catch this and shows an error
- **Avoidance:** if you're setting something that you would have set in the CSS file, use `.style`. If you would have set it in the HTML file, don't.

COMMON BUG: incorrect units on styles

```
theDiv.style.left = x; // BAD! should be x + "px"  
theDiv.style.backgroundColor = x + "px" + y + "px"; // BAD! missing space
```

JS

- all CSS property values must be Strings, and many require units and/or a specific format
- **Manifestation of bug:** code fails silently; style is not set
- **Detection:** use Firebug debugger, step through code and look at `style`
- **Detection:** use an `alert` immediately after style property is set

```
theDiv.style.left = 100; // BAD!  
alert("div left is " + theDiv.style.left);
```

JS

COMMON BUG: incorrect usage of existing styles

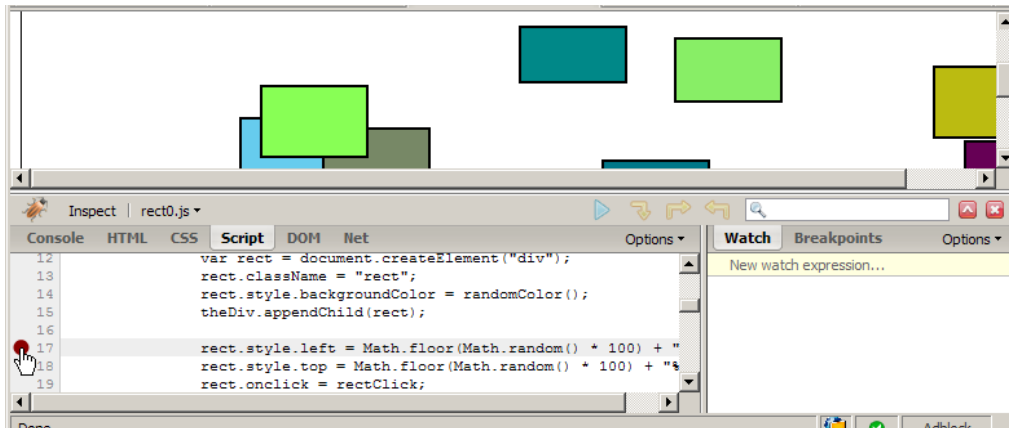
```
theDiv.style.top = this.getStyle("top") + 100 + "px"; // BAD! String + Number  
theDiv.style.top = parseInt(this.getStyle("top")) + 100 + "px";
```

JS

- the first example is equivalent to something like:
"200px" + 100 + "px", which evaluates to:
"200px100px"

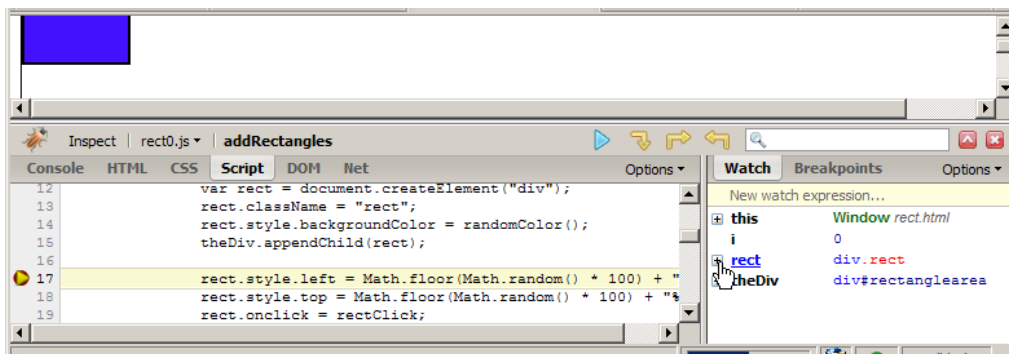
Debugging in Firebug


Firebug's debugger



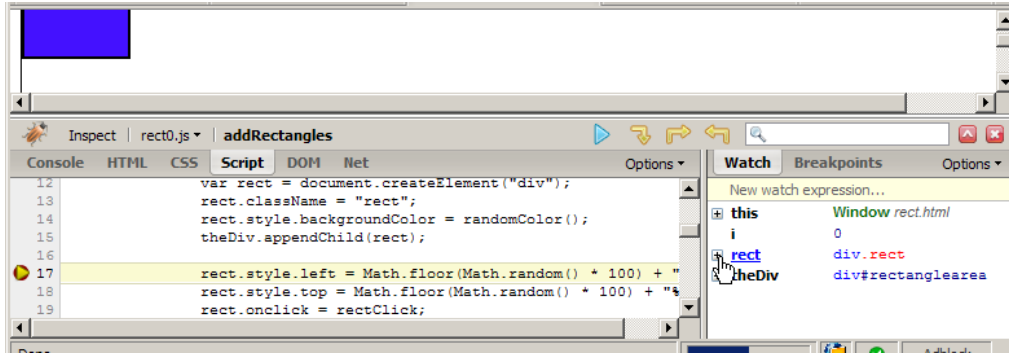
- open Firebug, click **Script** tab
- click to the left of a line to set a **breakpoint**
- refresh page
- when page runs, if it gets to that line in the JS code, program will halt





Breakpoints



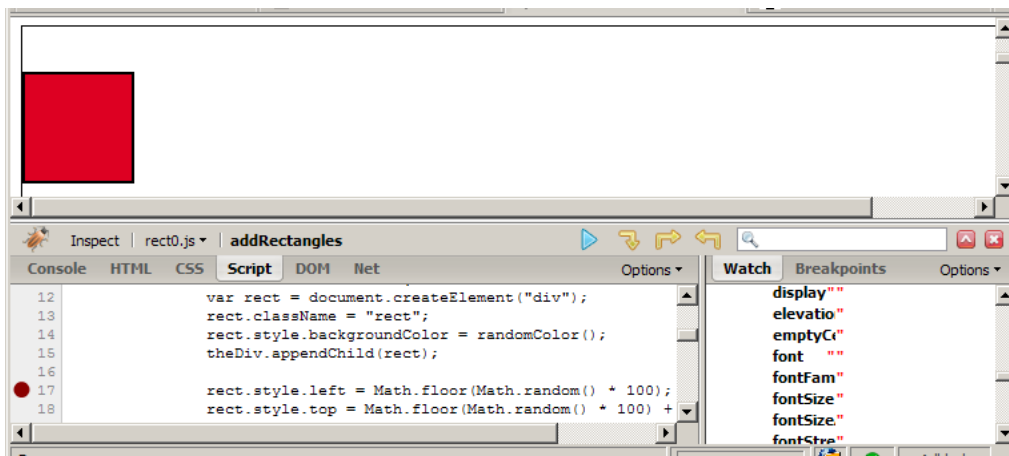
- **data**: once you've stopped at a breakpoint, you can examine any variables in the **Watch** tab at right
 - can click  to see properties/methods inside any object
 - **this** variable holds data about current object, or global data
 - if the object is global or not listed, type its name in the "New watch expression..." box
 - make sure Options → Show DOM Properties is checked, so you can see any DOM-related values


Stepping through code



- **code:** once stopped at a breakpoint, you can continue execution:
 -  **continue** (F8): start the program running again
 -  **step over** (F10): run the current line of code completely, then stop again
 -  **step into** (F11): run the current line of code, but if it contains any calls to other methods, jump into those and stop
 -  **step out** (Shift-F11): run the current function to completion and return, then stop

Debugging CSS property code



- expand DOM object with , and expand its `style` property to see all styles
- also look at HTML (left) tab, Style (right) tab to see styles

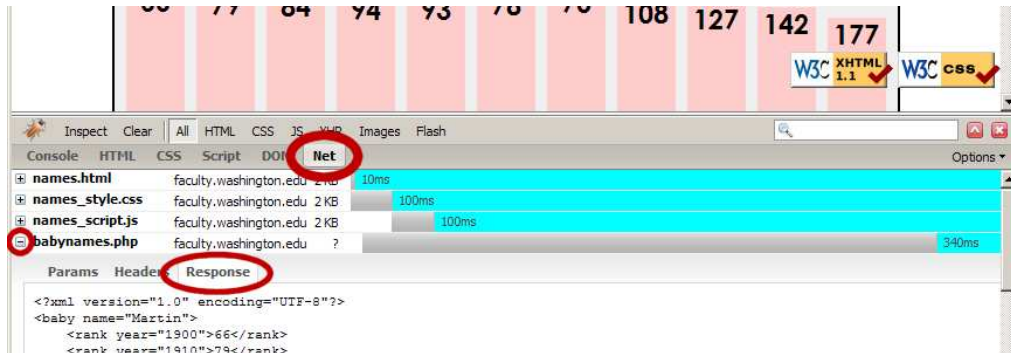
Ajax code bugs

When writing Ajax programs, there are new kinds of bugs that are likely to appear.

- Nothing happens!
- The `responseText` or `responseXML` has no properties.
- The data isn't what I expect.

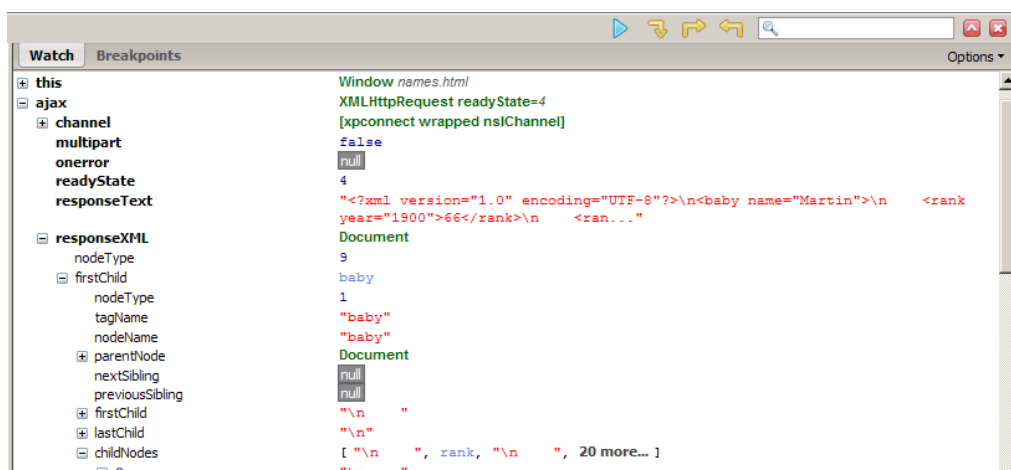
How do we find and fix such bugs?

Debugging Ajax code



- Net tab shows each request, its parameters, response, any errors
- expand a request with `+` and look at **Response** tab to see Ajax result

Debugging responseXML



- can examine the entire XML document, its node/tree structure

General good coding practices

- ALWAYS code with Firebug installed
- code a little, test a little
- follow good general coding principles
 - remove redundant code
 - make each line short and simple
 - always use { } even when not needed on if, for, etc.
- use lines and variables liberally
 - it's good to save parts of a complex computation as variables
 - helps see what part of a big expression was bad/undefined/etc.
 - blank lines and profuse whitespace make code easier to read
- don't fear the Firebug debugger