Fixing Common JavaScript Bugs

Elijah Manor
http://elijahmanor.com
@elijahmanor

pluralsight
hardcore developer training

Prerequisites

JavaScript Fundamentals

Everything a beginner needs to get started programming with JavaScript

Authored by: Liam McLennan
Duration: 2h 53m
Level: Beginner
Released: 1/25/2011

JavaScript From Scratch

Learn JavaScript with no prior programming experience

Authored by: Jesse Liberty
Duration: 1h 52m
Level: Beginner
Released: 5/20/2013

Statements

Functions

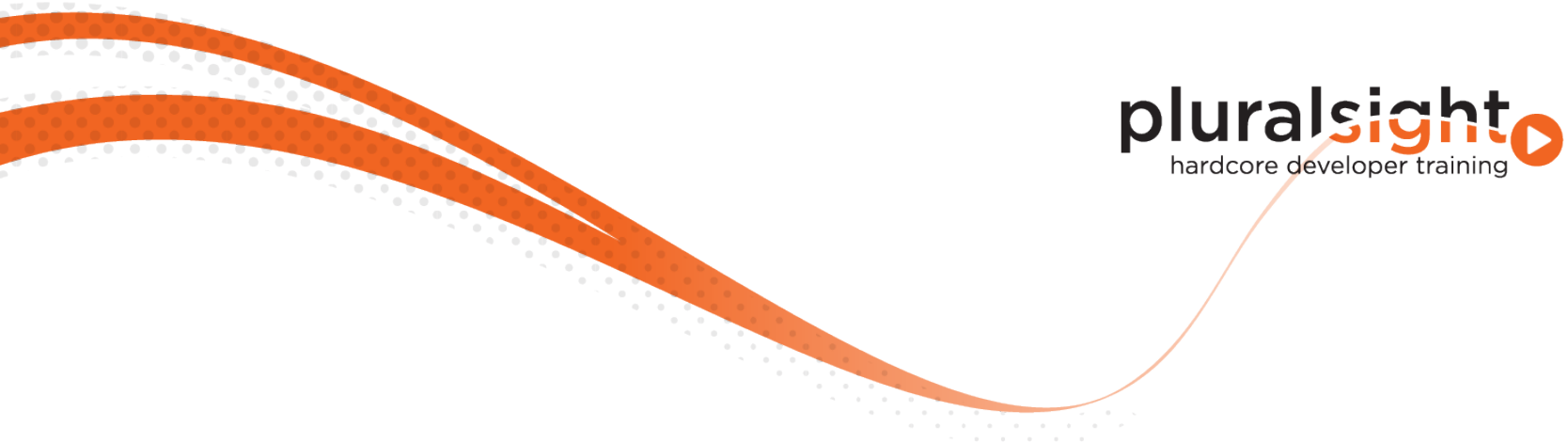
Expressions & Operators

Values, Variables, & Literals

Objects

Fixing Common JavaScript Bugs

Statements



Missing Mark Bug

Examine

```
function getNames() {
  var length = 0, names = ""

  ['John', 'Susan', 'Joe'].forEach(function (name, i) {
    length = i + 1
    names += name + ' '
  })

  return
  {
    length: length,
    names: names
  }
}
```

Missing Mark Bug

Expose

```
function getNames() {
  var length = 0, names = ""

  ['John', 'Susan', 'Joe'].forEach(function (name, i) {
    length = i + 1
    names += name + ' '
  })

  return
  {
    length: length,
    names: names
  }
}
```

Uncaught SyntaxError:
Unexpected token :

Missing Mark Bug

Explain

- **Automatic Semicolon Insertion (ASI)**
  - JavaScript needs semicolons in order to parse the language, however, there is a mechanism called automatic semicolon insertion to assist with parsing http://es5.github.io/#x7.9

- **ASI Rules**
  - Applied when new line or curly brace is followed by invalid token
  - Applied when new line comes before -- or ++ token
  - Applied when new line follows a continue, break, return or throw statement
  - Applied at end of a file if needed to parse

- **ASI Exceptions**
  - Not applied if would result in an empty statement
  - Not applied inside head of a for statement

Missing Mark Bug

Explain

```
function getNames() {
  var length = 0, names = ""

  ['John', 'Susan', 'Joe'].forEach(function (name, i) {
    length = i + 1;
    names += name + ' ';
  });

  return;
  {
    length: length,
    names: names
  };
}
```

Automatic Semicolon Insertion (ASI)

Missing Mark Bug

Explain

Returns `undefined`

```
func
  var length = 0; names = ""['John', 'Susan',
'Joe'].forEach(function (name, i) {
    length = i + 1;
    names += name + ' ';
  });

  return;
  {
    length: length,
    names: names
  };
}
```

Returns `undefined`

Missing Mark Bug

Exterminate

Semicolon-less JavaScript Rules

1. Don't end your statements with a semicolon
2. If statement starts with `[`, `(`, or a binary operator (+*/-,.) then insert a semicolon before it

Example:

```
function bootstrap(home) {
  var selector = typeof home === "string" ?
    "#home" + home : null
  if (selector) home = null
  ;(home || new HomeView(selector)).render()
}
```

Missing Mark Bug

Exterminate

```
function getNames() {
  var length = 0, names = ""

  ;['John', 'Susan', 'Joe'].forEach(function (name, i) {
    length = i + 1
    names += name + ' '
  })

  return {
    length: length,
    names: names
  }
}
```

Missing Mark Bug

Exterminate

Using Semicolons As Expected

1. Follow the official specification when semicolons are required
2. Use tools like JSLint or JSHint to give you feedback and integrate into your code editor

Example:

```
function bootstrap(home) {
  var selector = typeof home === "string" ?
    "#home" + home : null;
  if (selector) home = null;
  (home || new HomeView(selector)).render();
}
```

Missing Mark Bug

Exterminate

```
function getNames() {
  var length = 0, names = "";

  ['John', 'Susan', 'Joe'].forEach(function (name, i) {
    length = index + 1;
    names += name + ' ';
  });

  return {
    length: length,
    names: names
  };
}
```

Missing Mark Bug

Examine

DEMO

REQUIRED

COMPLETE

Examine

Fresh Function Bug

```
var people = [
  { fname: "John", lname: "Smith", bday: "2/2/1979" },
  { fname: "Jane", lname: "Smith", bday: "3/3/1981" },
  { fname: "Jack", lname: "Smith", bday: "4/4/1982" }
];

people.filter(function (person) {
  return new Date(person.bday).getFullYear() < 1980;
}).map(function (person) {
  return {
    name: person.fname + " " + person.lname,
    age: moment().diff(moment(person.bday), "years")
  };
});
```

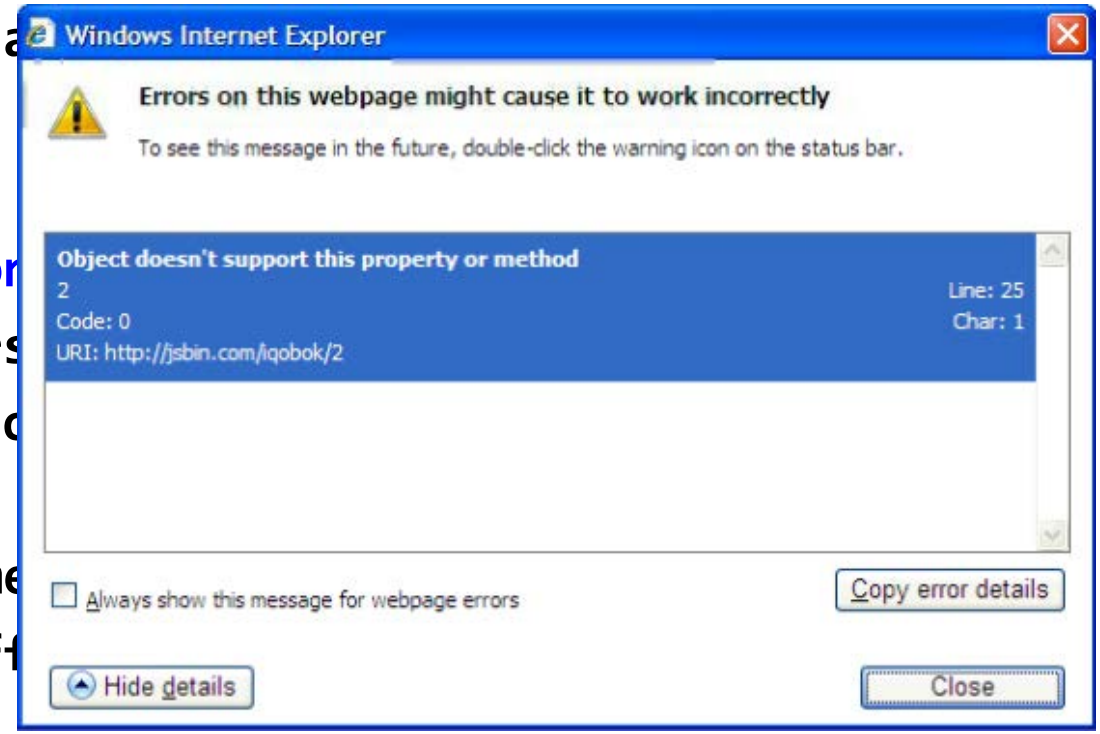
Expose

Fresh Function Bug

```
var people = [
  { fname: "John", lname: "Smith", bday: "2/2/1979" },
  { fname: "Jane", lname: "Smith", bday: "3/3/1981" },
  { fname: "Jack", lna
```

IE8 Throws an Error

```
people.filter(function
  return new Date(pers
}).map(function (perso
  return {
    name: person.fname
    age: moment().diff
  };
});
```



Fresh Function Bug

Explain

- **ECMAScript 5 array methods in Chrome, Firefox, Safari, Opera, IE9+**
  - map, reduce, reduceRight, filter, forEach, every, some, indexOf, lastIndexOf
- **Polyfill**
  - es5-shim - https://github.com/kriskowal/es5-shim/
- **Shim**
  - Underscore.js - http://underscorejs.org/
  - Lo-Dash - http://lodash.com/

Fresh Function Bug

Exterminate

```
<!DOCTYPE html>
<html xmlns="http://
<head><title>Extermi
<body>
  <script src="es5-shim.min.js"></script>
  <script>
    var people = [ /* ... */ ];

    people
      .filter(function (person) { return /* ... */; })
      .map(function (person) { return /* ... */; });
  </script>
</body>
</html>
```

Polyfilling ECMAScript 5 array methods

Fresh Function Bug

Extra

```
var people = [
  { fname: "John", lname: "Smith", bday: "2/2/1979" },
  { fna                                    "3/3/1981" },
  { fna                                    "4/4/1982" }
];
people.reduce(function (memo, person) {
  if (new Date(person.bday).getFullYear() < 1980) {
    memo.push({
      name: person.fname + " " + person.lname,
      age: moment().diff(moment(person.bday), "years")
    });
  }
  return memo;
}, []);
```

.reduce() can combine .filter() and .map()

Fresh Function Bug

DEMO

DEMO

REQUIRED

NOT COMPLETE

Examine

Tumble Through Bug

```
function getPrice(item) {
    var price = 0;
    switch (item) {
        case "apple": price = 1.25; break;
        case "banana": price = 0.75; break;
        case "orange": price = 1; break;
        case "passionfruit": price = 1.5;
        case "pear": price = 0.5; break;
        default: price = 0;
    }
    return price;
}

console.log(getPrice("passionfruit"));
```

Expose

Tumble Through Bug

```
function getPrice(item) {
    var price = 0;
    switch (item) {
        case "apple": price = 1.25; b
        case "banana": price = 0.75;
        case "orange": price = 1; bre
        case "passionfruit": price = 1.5;
        case "pear": price = 0.5; break;
        default: price = 0;
    }
    return price;
}

console.log(getPrice("passionfruit")); // 0.5
```

No `break` so falls through to "pear" price

JSHint: Line 7: case "passionfruit": price = 1.50;
--- Expected a 'break' statement before 'case'.

Tumble Through Bug

Explain

```
switch (item) {
  case "value1": /* code */ break;
  case "value2": /* code */ break;
  case "value3": /* code */ break;
  default:       /* code */
}
```

```
switch (item) {
  case "value1": /* co
  case "value2": /* co
  /* falls through */
  case "value3": /* code */ break;
  default:       /* code */
}
```

JSHint won't complain if you add
/* falls through */

Exterminate

Tumble Through Bug

```
function getPrice(item) {
    var price = 0;
    switch (item) {
        case "apple": price = 1.25; break;
        case "banana": price =
        case "orange": price =
        case "passionfruit": price = 1.5; break;
        case "pear": price = 0.5; break;
        default: price = 0;
    }
    return price;
}

console.log(getPrice("passionfruit")); // 1.5
```

Add `break` for "passionfruit"

Tumble Through Bug

Extra

```
var store = (function () {
  var prices = {
    apple: 1.25,
    banana: 0.75,
    orange: 1.0,
    passionfruit: 1.5,
    pear: 0.5
  }, getPrice = function (item, quantity) {
    return prices[item] * quantity;
  };
  return { getPrice: getPrice };
}());

console.log(store.getPrice("passionfruit", 2)); // 3
```

Keep price in an object and encapsulate in a module

Extra

Tumble Through Bug

Each fruit has their own function

```
var prices = {};

prices.apple  = function (num) { return 1.25 * num; };
prices.banana = function (num) { return 0.75 * num; };
prices.orange = function (num) { return 1.00 * num; };
prices.passionfruit = function (num) {
    var month = new Date().getMonth(),
        price = month < 4 && month > 10 ? 2.5 : 1.5;
    return price * num;
};
prices.pear   = function (num) { return 0.50 * num; };

console.log(prices["passionfruit"](2)); // 3 or 5
```

Tumble Through Bug

DEMO

DEMO

REQUIRED

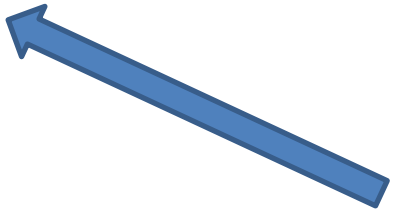
NOT COMPLETE

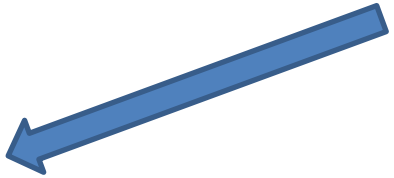
Strictly Stray Bug

Examine

```
// script1.js
"use strict";
var person = {
  fname: "John",
  lname: "Smith"
};

// script2.js
(function () {
  oops = "uhh-ohh";
  console.log(oops);
}());
```





Two Script Files

Expose

Strictly Stray Bug

```
"use strict";var
person={fname:"John",lname:"Smith"};(function(){oops="uhh
-ohh",console.log(oops)})();
//@ sourceMappingURL=bundle1.min.js.map
```

Error only when combined & minified

Uncaught ReferenceError: oops is not defined

Strictly Stray Bug

Explain

What is Strict Mode?

- Prevent accidental global variables
- Assignment to non-writable property will throw exception
- Deleting undeletable property will throw exception
- Requires properties to be unique in object literal
- Requires parameter names in function to be unique
- Prevent octal number literals
- Makes use of `with` a syntax error
- New variables aren't introduced in surrounding scope with `eval`
- Etc…

Strictly Stray Bug

Explain

Apply Strict Mode

JSHint: "use strict"; --- Use the function form of "use strict".

1. For Scripts

```
"use strict";
var person = { fname: "John" };
console.log("I'm Strict");
```

2. For Functions

```
function test1() {
  "use strict";
  console.log("I'm Strict");
}
function test2() { console.log("I'm Not Strict"); }
```

Strictly Stray Bug

Exterminate

```
// script1.js
(function () {
  "use strict";
  var person = {
    fname: "John",
    lname: "Smith"
  };
}());

// script2.js
(function () {
  oops = "uhh-ohh";
  console.log(oops);
}());
```

Wrap code in IFFE so "use strict"; contained to function

Strictly Stray Bug

DEMO

DEMO

NOT REQUIRED

NOT COMPLETE

Parsing Parenthesis Bug

Examine

```
var store = function () {
    "use strict";
    var price = 1.25,
      getPrice = function (num) {
          return num * price;
      };
    return { getPrice: getPrice };
}();

function () {
    "use strict";
    var numberOfItems = 4;
    console.log(store.getPrice(numberOfItems));
}();
```

Parsing Parenthesis Bug

Expose

```
var store = function () {
    "use strict";
    var price = 1.25,
      getPrice = function (num) {
          return num * price;
      };
    retu                                ice };
}();

function () {
    "use strict";
    var numberOfItems = 4;
    console.log(store.getPrice(numberOfItems));
}();
```

Uncaught SyntaxError: Unexpected token (

JSHint: Line 14: }(); --- Function Declarations are not invocable. Wrap the whole Function invocation in parens.

Parsing Parenthesis Bug

Explain

```
function () {
  /* code */
}();
```

JavaScript can't parse this correctly…

```
( function () {
  /* code */
}() );
```

A simple wrapper fixes the problem

```
!function () {
  /* code */
}();
```

Technically you can prepend a unary operator

Parsing Parenthesis Bug

Explain

```
var myObject = function () {
  /* code */
}();
```

Valid, but not consistent

JSHint: Line 3: }(); --- Wrap an immediate function invocation in parens to assist the reader in understanding that the expression is the result of a function, and not the function itself.

```
var myObject = ( function () {
  /* code */
}() );
```

A simple wrapper makes consistent

Parsing Parenthesis Bug

Exterminate

```
var store = (function () {
    "use strict";
    var price = 1.25,
      getPrice = function (num) {
          return num * price;
      };
    return { getPrice: getPrice };
}());

(function () {
    "use strict";
    var numberOfItems = 4;
    console.log(store.getPrice(numberOfItems));
}());
```

Added paren wappers to both IIFEs

Parsing Parenthesis Bug

DEMO

DEMO

NOT REQUIRED

NOT COMPLETE

Examine

Evil Eval Bug

```
var safe = (function () {
  var combinations = { main: "12345", fire: "67890" };
  var open = function (type, attempt) {
    var combination = eval("combinations." + type);
    if (attempt === combination) {
      console.log("safe opened");
    } else {
      console.log("incorrect combination");
    }
  };
  return { open: open };
}());

safe.open("main", "12345"); // safe opened
```

Evil Eval Bug

Expose

```
safe.open("_;console.log(JSON.stringify(combinations));",
"999");
```

```
{"main":"12345","fire":"67890"}
incorrect combination
```

```
safe.open("main='999';", "999");
```

```
safe opened
```

Evil Eval Bug

Explain

```
var safe = (function () {
  var combinations = { main: "12345", fire: "67890" };
  var open = function (type, attempt) {
    var combination = eval("combinations." + type);
    /* code */
  };
  return { open: open };
}());
```

JSHint: Line 4: var combination = eval("combinations." + type); --- eval can be harmful.

Eval() isn't evil, just misunderstood http://j.mp/19pvTsc

```
eval("combinations.main") //12345
combinations.main         //12345
combinations["main"]      //12345
```

Exterminate

Evil Eval Bug

```
var safe = (function () {
  var combinations = { main: "12345", fire: "67890" };
  var open = function (type, attempt) {
    var combination = combinations[type];
    if (attempt === combin
      console.log("safe op
    } else {
      console.log("incorrect combination");
    }
  };
  return { open: open };
}());

safe.open("main", "12345"); // safe opened
```

Don't use eval, use bracket notation

Evil Eval Bug

DEMO

DEMO

REQUIRED

NOT COMPLETE

Fickle Figure Bug

Examine

```
$(document).ready(function () {
  $("input.date").datepicker({
    minDate: -20,
    defaultDate: "+1w",
    maxDate: "+1M +5D",
    showWeek: true,
    numberOfMonths: 3,
  });
});
```

Fickle Figure Bug

Expose

```
$(document).ready(function() {
  $("input").datepicker({
    minDate: -20,
    defaultDate: "+1w",
    maxDate: "+1M +5D",
    showWeek: true,
    // numberOfMonths: 3,
  });
```

IE7 - Error: Expected identifier, string or number

```
$(document).ready(...
  $("input").datepicker({
    minDate: -20
    , defaultDate: "+1w"
    , maxDate: "+1M +5D"
    , showWeek: true
    , numberOfMonths: 3
  });
});
```

Fickle Figure Bug

Explain

Internet Explorer 7

```
var opts = { minDate: -20, showWeek: true, }; // Error
var numbers = [ 1, 2, 3, ]; // Error
```

Internet Explorer 8

```
var opts = { minDate: -20, showWeek: true, }; // Works
var numbers = [ 1, 2, 3, ]; // length 4
```

Internet Explorer 9

```
var opts = { minDate: -20, showWeek: true, }; // Works
var numbers = [ 1, 2, 3, ]; // length 3
```

Fickle Figure Bug

Exterminate

```
$(document).ready(function () {
  $("input
    minDat
    defaul
    maxDat
    showWe
    numberOfMonths: 3
  });
});
```

Errors:

- Line 7: `numberOfMonths: 3,`

  Extra comma. (it breaks older versions of IE)

Remove trailing comma or don't support IE7 or less

Extra

Fickle Figure Bug

If you do decide to drop IE7 support and use trailing commas keep in mind that…

- JSON.parse() has not changed and does not support trailing commas

Watch out!

```
JSON.parse('{"answer":42,}')
// SyntaxError: Unexpected token }

JSON.parse('[42,]')
// SyntaxError: Unexpected token ]
```

Fickle Figure Bug

DEMO

DEMO

NOT REQUIRED

NOT STARTED

Summary

- Be intentional about your semicolon placement
- If you are using ECMAScript 5 methods use a polyfill or shim
- Be careful of the switch fall through or use alternate approach
- Use JavaScript strict mode, but be careful to implement per function
- Make sure to wrap your IIFEs to be consistent & obvious to developer
- Eval can be dangerous and so be careful. Many times can rewrite
- IE7/8 don't like trailing commas in object and array literals