Fixing Common JavaScript Bugs

Values, Variables, and Literals

Elijah Manor
http://elijahmanor.com
@elijahmanor

pluralsight
hardcore developer training

Booked Byword Bug

Examine

```
var collection = (function() {
  var items = [];
  var add = function(item) { items.push(item) };
  var get = function(index) { return items[index] };
  var delete = function(index) { items.splice(index, 1) };

  return {
    add: add,
    get: get,
    delete: delete
  };
}());
```

Booked Byword Bug

Expose

```
var collection = (function() {
  var items = [];
  var add = function(item) { items.push(item) };
  var get = function(index) { return items[index] };
  var remove = function(index) { items.splice(index, 1) };

  return {
    add:
    get:
    delete: remove
  };
}())
```

Uncaught SyntaxError:
Unexpected token delete

Expected identifier, string or number

Booked Byword Bug

Explain

Reserved Keywords

break	case	catch	continue	debugger
default	delete	do	else	finally
for	function	if	in	instanceof
new	return	switch	this	throw
try	typeof	var	void	while
with				

Reserved Keywords for the Future

class	enum	export	extends	implements
import	interface	let	package	private
protected	public	static	super	yield

Booked Byword Bug

Explain

In ECMAScript 3 reserved words can not be used as identifier names or identifiers.

```
// Identifier Names
a.impo
a["imp
a = {
```

Errors:

- Line 8: `var delete = function(index) { items.splice(index, 1); };`

  Expected an identifier and instead saw 'delete' (a reserved word).

```
}
```

```
// Identifier
function import() {}
var import = "test";
```

In ECMAScript 5 reserved words can be used as identifier names, but not identifiers.

```
// Identifier Names
```

```
// Identifier
function import() {}
var import = "test";
```

Booked Byword Bug


```
var collection = (function() {
  var items = [];
  var add = function(item) { items.push(item) };
  var get = function(index) { return items[index] };
  var remove = function(index) { items.splice(index, 1) };

  return {
    add: add,
    get: get,
    "delete": remove
  };
}());
```

```
collection["delete"](1);
```

Booked Byword Bug

Examine

```
DEMO

REQUIRED - 1

NOT COMPLETE
```

Revealing Recall Bug

Examine

```
function Bank(balance) {
  this.fee = 0.01;
  this.account = { balance: balance };
}

Bank.prototype.deposit = function (amount) {
  var amountWithFee = fee = this.fee;
  amountWithFee = amount - (amount * fee);
  this.account.balance += amountWithFee;
}

Bank.prototype.withdrawal = function (amount) {
  amountWithFee = amount + (amount * this.fee);
  this.account.balance -= amountWithFee;
};
```

Revealing Recall Bug

Two Global Variables Created!

```
function Bank(ba
  this.fee = 0.01;
  this.account = { balance: balance };
}

Bank.prototype.deposit = function (amount) {
  var amountWithFee = fee = this.fee;
  amountWithFee = amount - (amount * fee);
  this.account.balance += amountWithFee;
}

Bank.prototype.withdrawal = function (amount) {
  amountWithFee = amount + (amount * this.fee);
  this.account.balance -= amountWithFee;
};
```

If you don't declare your variables, JavaScript will for you… on the global object!

Explain

```
name = "John";

function greet(n) {
  greeting = "Hi " + n;
  /* ... more code ... */
}
greet(name);

function mangle(source) {
  var index = length = 0;
  /* ... more code ... */
}
mangle(name);
```

```
var name = "John";

function greet(n) {
  var greeting = "Hi " + n;
  /* ... more code ... */
}
greet(name);

function mangle(source) {
  var index = 0, length = 0;
  /* ... more code ... */
}
```

The functions are global too! Ahh

Revealing Recall Bug

Explain

Object Literal

```
var myObject = {
  name: "John",
  greet: function () {
    /* ... code ... */
  },
  mangle: function (src) {
    /* ... code ... */
  }
};
```

IIFE

```
var myObject = (function() {
  var name = "John",
    greet = function () {
      /* ... code ... */
    },
    mangle = function (src) {
      /* ... code ... */
    };
  return {
    greet: greet,
    mangle: mangle
  };
}());
```

Revealing Recall Bug

Exterminate

```
var Bank = (function () {
  function Bank(balance) {
    this.fee = 0.01; this.account = { balance: balance };
  }
  Bank.prototype.deposit = function (amount) {
    amount -= amount * this.fee;
    this.account.balance += amount;
  }
  Bank.prototype.withdrawal = function (amount) {
    amount += amount * this.fee;
    this.account.balance -= amount;
  };
  return Bank;
})();
```

Revealing Recall Bug

Extra

Errors:

- Line 9: `var amountWithFee = fee = this.fee;`

  Variable fee was not declared correctly.

- Line 12: `}`

  Missing semicolon.

- Line 9: `var amountWithFee = fee = this.fee;`

  'fee' is not defined.

- Line 10: `amountWithFee = amount - (amount * fee);`

  'fee' is not defined.

- Line 15: `amountWithFee = amount + (amount * this.fee);`

  'amountWithFee' is not defined.

- Line 16: `this.account.balance -= amountWithFee;`

  'amountWithFee' is not defined.

Revealing Recall Bug

Examine

DEMO

REQUIRED – 2

NOT COMPLETE

Relative Realism Bug

Examine

```
var element = document.getElementById("greeting");
function html(value) {
    if (value === undefined) {
        return element.innerHTML;
    } else if (typeof value === "string") {
        element.innerHTML = value;
    } else if (typeof value === "function") {
        element.innerHTML = value(element.innerHTML);
    }
}
html("Hello");
console.log(html());
html(function (text) { return text + " World!"; });
console.log(html());
```

Relative Realism Bug

```
undefined = true;
var element = document.getElementById("greeting");
function html(value) {
    if (value === undefined) {
        return element.innerHTML;
    } else if (typeof value === "string") {
        element.innerHTML = value;
    } else if (typeof value === "function") {
        element.innerHTML = value(element.innerHTML);
    }
}
html("Hello");
console.log(html());
html(function (text) { return text + " World!"; });
console.log(html());
```

```
undefined
undefined
```

Relative Realism Bug

Explain

Reserved Keywords

break	case	catch	continue	debugger
default	delete			finally
for	function			instanceof
new	return			throw
try	typeof			while
with				

WHERE DEFINED!?!

Errors:

- Line 4: `undefined = true;`

  Bad assignment.

Reserved Keyw

class				plements
import				ivate
protected				eld

In ECMAScript 3 undefined was not a reserved word & could be reassigned!

Thankfully in ECMAScript 5 undefined, NaN, & Infinity are all read-only

Relative Realism Bug

Explain

Use IIFE to Protect Undefined & Help with Minification

```
function sayHello(name, empty) {
  console.log("Hi" + name, empty);  // Hi John undefined
}
sayHello("John");

(function (name, undefined) {
  // forcing undefined variable to have undefined value
  if (name === undefined) { console.log("undefined") }
}("John"));

!function (n, o) { n === o && console.log("Name is
undefined") }("John")
```

Relative Realism Bug

Exterminate

```
undefined = true;
(function (undefined) {
  var element = document.getElementById("greeting");
  function html(value) {
    if (value === undefined) {
      return element.innerHTML;
    } else if (typeof value === "string") {
      element.innerHTML = value;
    } else if (typeof value === "function") {
      element.innerHTML = value(element.innerHTML);
    }
  }
  html(function (text) { return text + " World!"; });
  alert(html());
}());
```

Relative Realism Bug

Examine

DEMO

REQUIRED - 3

NOT COMPLETE

Tangled Tag Bug

Examine

```
<!DOCTYPE html>
<html>
<head>
  <!- jquery.js & jquery-ui.js, jquery-ui.css -->
</head>
<body>
  <input id="datePicker" class="date" />
  <script>
    datePicker = $(".date");
    datePicker.datepicker();
  </script>
</body>
</html>
```

Tangled Tag Bug

Expose

```
<!DOCTYPE html>
<html>
<head>
  <!- jquery.js & jquery-ui.js, jquery-ui.css -->
</head>
<body>
  <input id="datePicker" class="date" />
  <script>
    datePicker = $(".date");
    datePicker.datepicker();
  </script>
</body>
</html>
```

Object doesn't support this property or method

Tangled Tag Bug

Explain

Named acce

- Names of
- A, applet,
  `name` at
- HTML ele

amed-access-window

, frameset)

bject that have a

Errors:

- Line 1: `datePicker = $(".date");`

  'datePicker' is not defined.

- Line 2: `datePicker.datepicker();`

  'datePicker' is not defined.

```
<div id="h
<script>
  hello.innerHTML = "Howdy!";
</script>
```

Same thing as...

```
window.hello.innerHTML = "Howdy!";
```

Tangled Tag Bug

Exterminate

```
<!DOCTYPE html>
<html>
<head><!- jquery.js,jquery-ui.js,jquery-ui.css --></head>
<body>
  <input id="datePicker" class="date" />
  <script>
    (function () {
      var datePicker = $(".date");
      datePicker.datepicker();
    }());
  </script>
</body>
</html>
```

Tangled Tag Bug

Exterminate

```
<!DOCTYPE html>
<html>
<head><!- jquery.js,jquery-ui.js,jquery-ui.css --></head>
<body>
  <input id="datePicker" class="date"></input>
  <script>
    $(document).ready(function () {
      var datePicker = $(".date");
      datePicker.datepicker();
    });
  </script>
</body>
</html>
```

Tangled Tag Bug

Examine

DEMO

NOT REQUIRED - 1

NOT COMPLETE

Double Define Bug

Examine

```
(function() {

  var person = {
    name: "John Smith",
    phone: "555-123-4567",
    address: "123 White Ave.",
    city: "Nashville", state: "TN", zip: "90210",
    phone: "555-987-6543",
    toString: function() {
      return this.name + ": " + this.phone;
    }
  };
  console.log(person.toString());
}());
```

Double Define Bug


```
(function() {

  var person = {
    name: "John Smith",
    phone: "555-123-4567",
    address: "123 White Ave.",
    city: "Nashville", state: "TN", zip: "90210",
    phone: "555-987-6543",
    toString: function() {
      return this.name + ": " + this.phone;
    }
  };
  console.log(person);
}());
```

```
John Smith: 555-987-6543
```

Double Define Bug

Explain

Errors:

- Line 3: `key1: "Goodbye.1" // overwrites previous key1`

  Duplicate key 'key1'.

- Line 8: `function myFunction(param1, param1) {`

  'param1' is already defined.

```
var myO
  key1:
  key1:
};
console.log(myObject.key1); // Goodbye.1

function myFunction(param1, param1) {
  console.log(param1);    // Goodbye.2
}
myFunction("Hello.2", "Goodbye.2");
```

Double Define Bug

Explain

If we turn on strict mode, then these become exceptions! ☺

```
(function() {
  "use strict";

  var myObject = {
    key1: "Hello.1",
    key1: "Goodbye.1"
  };
  function myFunction(key1, key1) {
    console.log(key1);
  }
}());
```

Uncaught SyntaxError: Duplicate data property in object literal not allowed in strict mode

Uncaught SyntaxError: Strict mode function may not have duplicate parameter names

Double Define Bug

Exterminate

```
(function() {
  "use strict";

  var person = {
    name: "John Smith",
    phone: { home: "555-123-4567", cell: "555-987-6543" },
    address: "123 White Ave.",
    city: "Nashville", state: "TN", zip: "90210",
    toString: function() {
      return this.name + ": " + this.phone.home;
    }
  };
  console.log(person.toString());
}());
```

Double Define Bug

Examine

DEMO

NOT REQUIRED - 2

NOT COMPLETE

Transform Total Bug

Examine

```
function purchase(item, amount) {
    amount = parseInt(amount);
    console.log("Got " + item + ": $" + amount.toFixed(2));
}

purchase("Eggs", "01");
purchase("Bacon", "08");
```

Transform Total Bug

Expose

```
function purchase(item, amount) {
    amount = parseInt(amount);
    console.log("Got " + item + ": $" + amount.toFixed(2));
}

purchase("Eggs", "01");
purchase("Bacon", "08");
```


```
Got Eggs: $1.00
Got Bacon: $0.00
```

Transform Total Bug

Explain

Usage

```
var p                                    lix);
```

Errors:

- Line 4: `amount = parseInt(amount);`

  Missing radix parameter.

If radix is undefined or 0 then…

- If string & starts with "0x" or "0X" then radix is 16
- If string & starts with "0" then radix is 8 *
- If string & starts with something else then radix 10

* Exception: If ECMAScript 5 then radix is 10

```
console.log(parseInt("0xA")); // 10
console.log(parseInt("015")); // 13 (ES3) or 15 (ES5)
```

Transform Total Bug

Exterminate

```
function purchase(item, amount) {
    amount = parseInt(amount, 10);
    console.log("Got " + item + ": $" + amount.toFixed(2));
}

purchase("Eggs", "01");
purchase("Bacon", "08");
```

```
Got Eggs: $1.00
Got Bacon: $8.00
```

Transform Total Bug

Examine

DEMO

REQUIRED - 4

NOT COMPLETE

Amount Aware Bug

Examine

```
function purchase(item, amount) {
  amount = parseFloat(amount);
  if (amount === NaN) { throw "Amount is not a number" }
  console.log("Got " + item + ": $" + amount.toFixed(2));
}

try {
  purchase("Eggs", "1.75");
  purchase("Bacon", "priceless");
} catch (e) {
  console.log(e);
}
```

Amount Aware Bug


```
function purchase(item, amount) {
  amount = parseFloat(amount);
  if (amount === NaN) { throw "Amount is not a number" }
  console.log("Got " + item + ": $" + amount.toFixed(2));
}

try {
  purchase("Eggs", "1.75");
  purchase("Bacon", "priceless");
} catch (e) {
  console.log(e);
}
```

```
Got Eggs: $1.75
Got Bacon: $NaN
```

Amount Aware Bug

Explain

"Unlike all other possible values in JavaScript, it is not possible to rely on the equality operators (== and ===) to determine whether a value is NaN or not, because both NaN == NaN and NaN === NaN evaluate to false." --MDN

Errors:

- Line 7: `if (amount === NaN) { throw "Amount is not a number"; }`

  Use the isNaN function to compare with NaN.

```
cons
cons
```

```
console.log(isNaN(NaN));  // true
```

Amount Aware Bug

Exterminate

```
function purchase(item, amount) {
  amount = parseFloat(amount);
  if (isNaN(amount)) { throw "Amount is not a number" }
  console.log("Got " + item + ": $" + amount.toFixed(2));
}

try {
  purchase("Eggs", "1.75");
  purchase("Bacon", "priceless");
} catch (e) {
  console.log(e);
}
```

```
Got Eggs: $1.75
Amount is not a number
```

Amount Aware Bug

Examine

DEMO

NOT REQUIRED - 3

NOT COMPLETE

Summary

- Beware of using reserved words
- Be careful of making global variables
- Protect undefined and help minification
- Watch out for global DOM elements off of the window
- Don't repeat the same object key or parameter name
- Always provide a radix when parsing an integer
- Use the isNaN() function when comparing against NaN