Associative map

...

Naming confusion

```
let x = [1,2,3]; x.map(x => x + 1);
```

This is NOT a lecture about list.map

What these slides are discussing is a COMPLETELY unrelated topic

Unfortunately, it has a the same name

Warmup (2 minutes)

Expression	Value
5 == 5	true
"hello" == "hello"	?
var x = {} x == x // this line	?
{} == {}	?

Difference between objects and primitives

5 == 5	true
"foo" == "foo"	true
false == false	true
{foo: 5} == {foo: 5}	false

Question

```
var x = {foo: "bar", bar: {}}

console.log(x == x);

console.log(x.foo == x.foo);

console.log(x.bar == x.bar);

console.log(x.foo == "bar");

console.log(x.bar == {});
```

Properties

if we write

```
user.name = "bob"
```

It will set a property name to "bob"

Bracket notation

We are all familiar with this kind of code

```
var x = {}

x.foo = 5
```

We can also write it like so

```
var x = {}

x['foo'] = 5
```

It is the SAME thing

Challenge question

Make an object with 10 000 properties in under 300 characters.

HINT

```
var x = {}

x['foo'] = 5 // same as x.foo = 5
```

Some javascript weirdness

```
var map = {}

map[5] = 6

map["6"] = 7

map[true] = 8

map[{foo: 10}] = 9
```

```
console.log(map["5"])

console.log(map[6])

console.log("true")

console.log(x[{bar: 12}])
```

What do you expect to happen?

Discuss in groups

```
var map = {}

map[5] = 6

map["6"] = 7

map[true] = 8

map[{foo: 10}] = 9
```

```
console.log(map["5"])

console.log(map[6])

console.log("true")

console.log(x[{bar: 12}])
```

There's a very simple rule that explains this behaviour. Get into your groups and try to figure it out.

Lightning round

For the following 11 questions, answer as a team

One answer sheet per team

Please discuss and come to a consensus

At the end, we'll tabulate results to see which team will dominate!

90 seconds per question

The teacher will use a stopwatch.

Warmup question

```
var x = {foo: {bar: {baz: [3]}}}

console.log(x.foo.bar.baz[0])
```

Question 1

What is the output of this code?

```
var x = {foo: 5}

console.log(x.foo)
```

Question 2

What is the output of this code?

```
var x = {}

x['foo'] = 6

console.log(x['foo'])
```

Question 3

What is the output of this code?

```
var x = {}

x['foo'] = 6

console.log(x.foo)
```

Question 4

What is the output of this code?

```
var x = {}

x['foo'] = 6

console.log(x[foo])
```

Question 5

What is the output of this code?

```
var x = {}

var bar = 'foo'

x['foo'] = 6

console.log(x[bar])
```

Question 6

What is the output of this code?

```
var x = {}

var bar = 'foo'

x[bar + 'z'] = 6

console.log(x.barz)
```

Question 7

What is the output of this code?

```
var x = {}

x[6 * 7] = 6

console.log(x[5 * 8])
```

Question 8

What is the output of this code?

```
var x = {}

x[2 + 2] = 6

console.log(x[4])
```

Question 9

What is the output of this code?

```
var x = {}

var y = {}

x[y] = 6

console.log(x[y])
```

Question 10

What is the output of this code?

```
var x = {}

x[{}] = 6

console.log(x[{}])
```

Question 11

What is the output of this code?

```
var x = {}

x[3 != 4] = 6

console.log(x["foo" == "foo"])
```

Done!

Hand over your sheets and let's go over them one by one.

Terminology review

What is this notation called?

```
a.foo
```

And what is this notation called?

```
a['foo']
```

Try and guess (no computer!)

bracket notation	dot notation
a['foo']	a.foo
a['bar']	?
a[0]	?
a[true]	?

Is there an error anywhere?

```
var x = []

x[0] = 1

x[1] = 2

console.log(x[1] - x[0])
```

```
var x = {}

x[0] = 1

x[1] = 2

console.log(x[1] - x[0])
```

indexOf

```
let x = 'foo'

x.indexOf('o');

x.indexOf('x');
```

Map example

```
var m = {} // we will use m as a map

for(var i = 0; i < 1100; i++) {

  var is = "" + i;

  m[is] = is.indexOf('1') != -1;

}

console.log(m['321'])

console.log(m['432'])
```

End of slides question

```
var x = {}

x[{bar: 123}] = 5

console.log(x[{foo:'abc'}])
```