Objects Revision

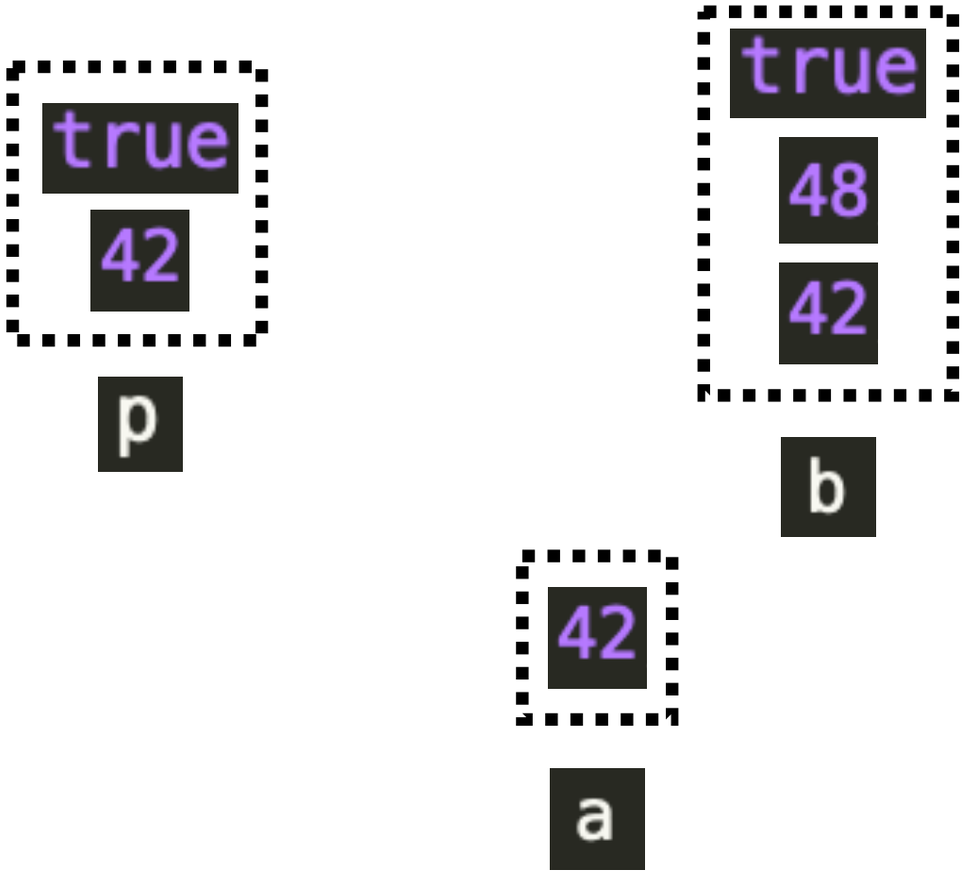
Problem Solving for Computer Science

Primitive Data Types in JavaScript

Type	Literals (Values)	
Boolean	`true` `false`	
Number (float)	`1` `3.14` `NaN`	
String	`"ps_for_cs"` `""`	← Empty string
Undefined	`undefined`	Variables without values
Null	`null`	Nothing

All other data types are **objects**

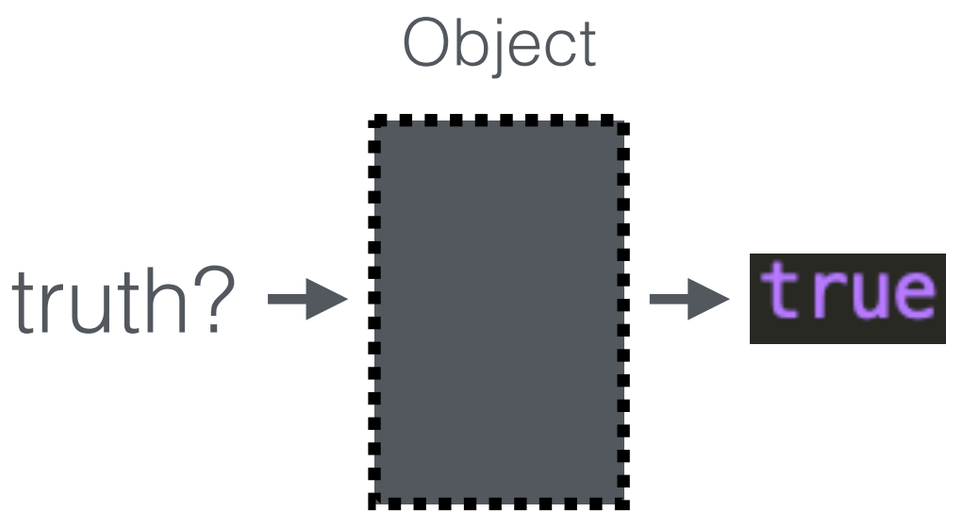
Objects



Variables can identify objects

Objects

Names are assigned to each of the properties




```
b
```

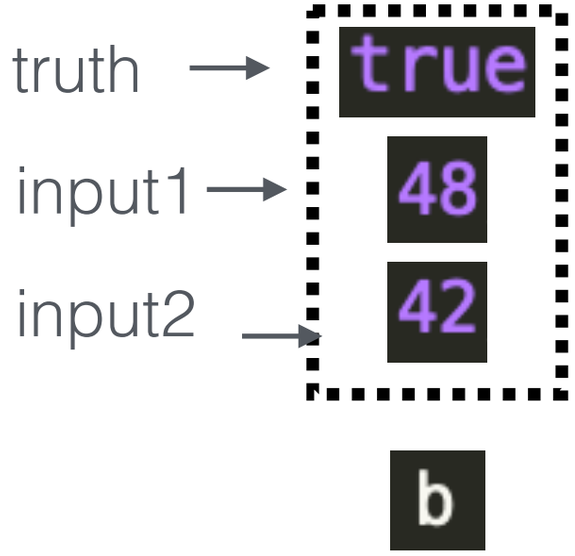
We can think of objects as black boxes

- We access data in particular ways using full stops
- We can ask particular questions, e.g. *what is truth?*
- We can alter values one at a time
- Can have its own ways to alter its own data: methods

Nice Computerphile video: https://www.youtube.com/watch?v=KyTUN6_Z9TM

Objects

Names are assigned to each of the properties



Creation of object:

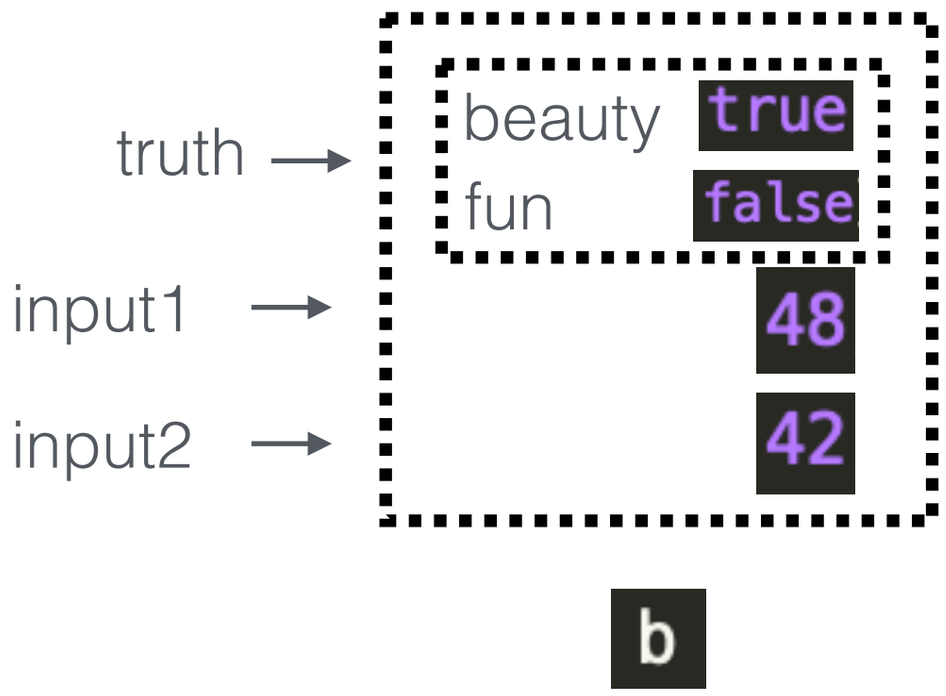
names

```
var b = {
    truth : true,
    input1 : 48,
    input2 : 42
};
```

values

Objects

Names are assigned to each of the properties

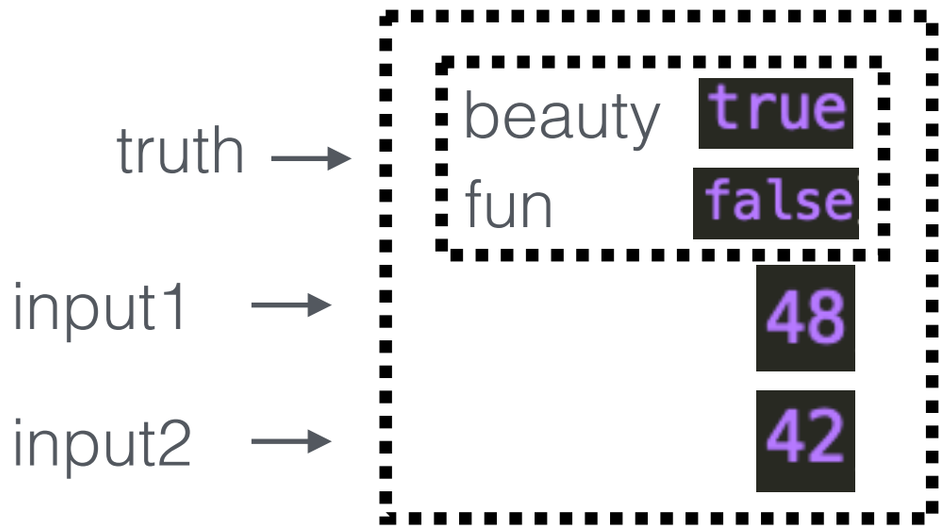


Objects can store other objects

```
var b = {
    truth : {beauty : true, fun: false},
    input1 : 48,
    input2 : 42
};
```

Objects

Names are assigned to each of the properties



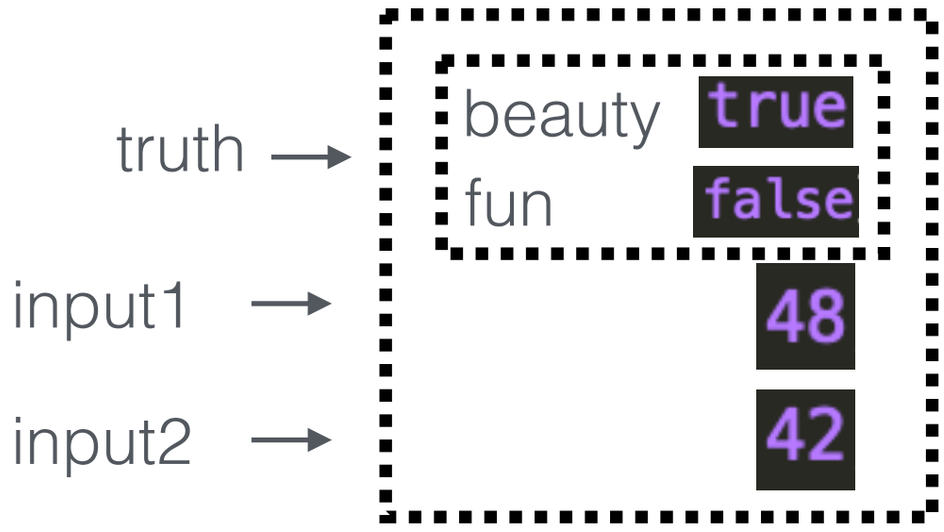

```
b
```

To access the contents of an object we need a syntax

```
objectName.propertyName
```

Objects

Names are assigned to each of the properties



`b`

To access the contents of an object we need a syntax

```
objectName.propertyName
```

```
console.log(b.truth);
console.log(b.input1);
console.log(b.truth.beauty);
```

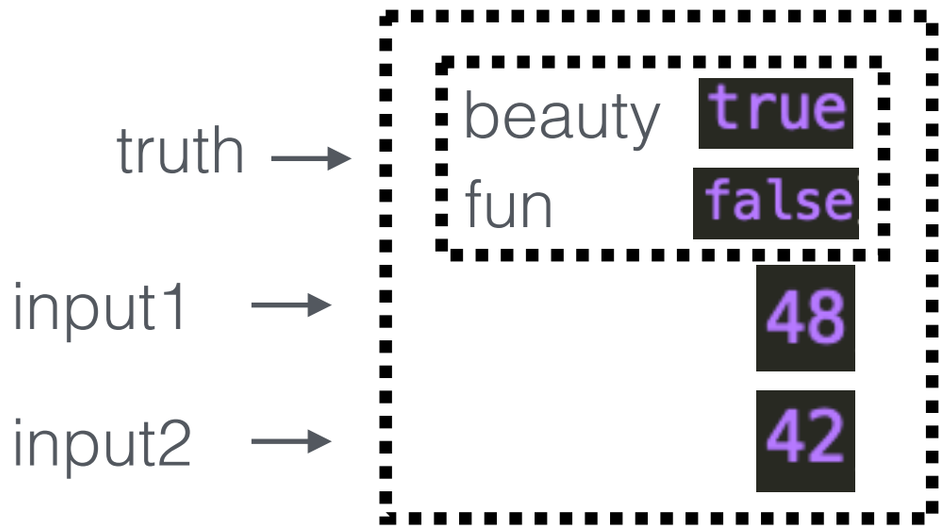
→ `{ beauty: true, fun: false }`

→ `48`

→ `true`

Objects

Names are assigned to each of the properties




```
b
```

Altering properties uses similar syntax

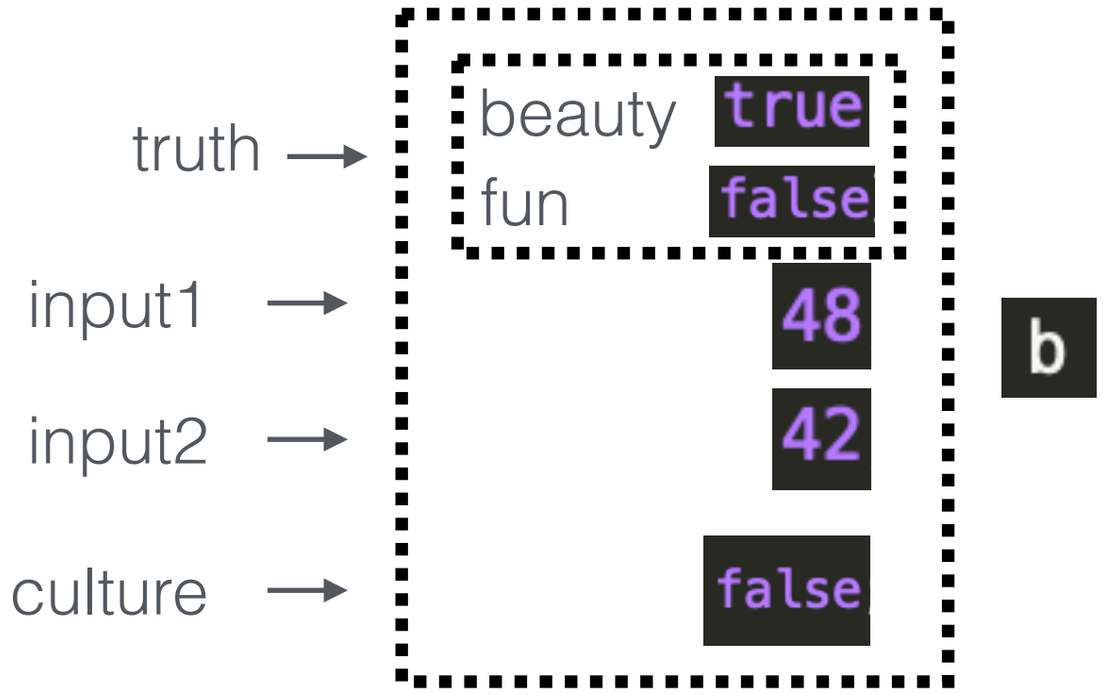
```
b.input1 = 50;
```

Creating new properties uses similar syntax

```
b.culture = false;
```

Objects

Names are assigned to each of the properties



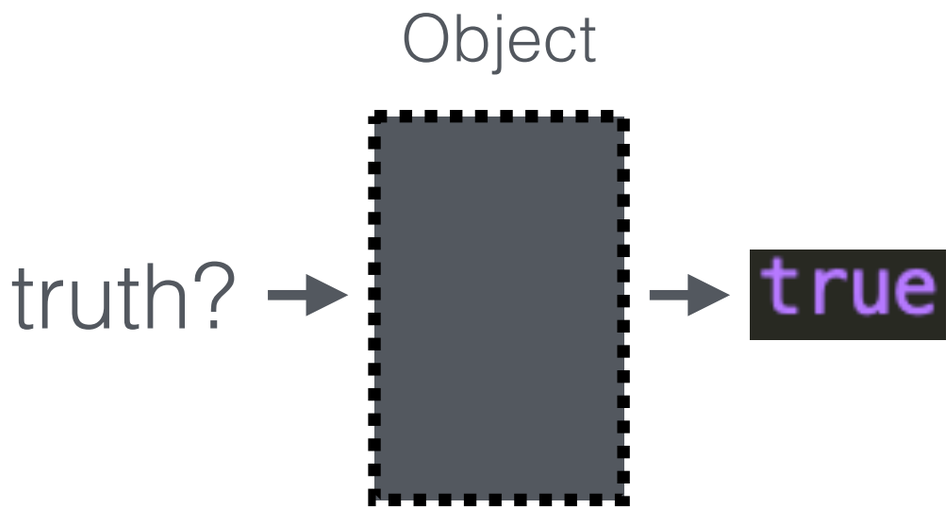
Altering properties uses similar syntax

```
b.input1 = 50;
```

Creating new properties uses similar syntax

```
b.culture = false;
```

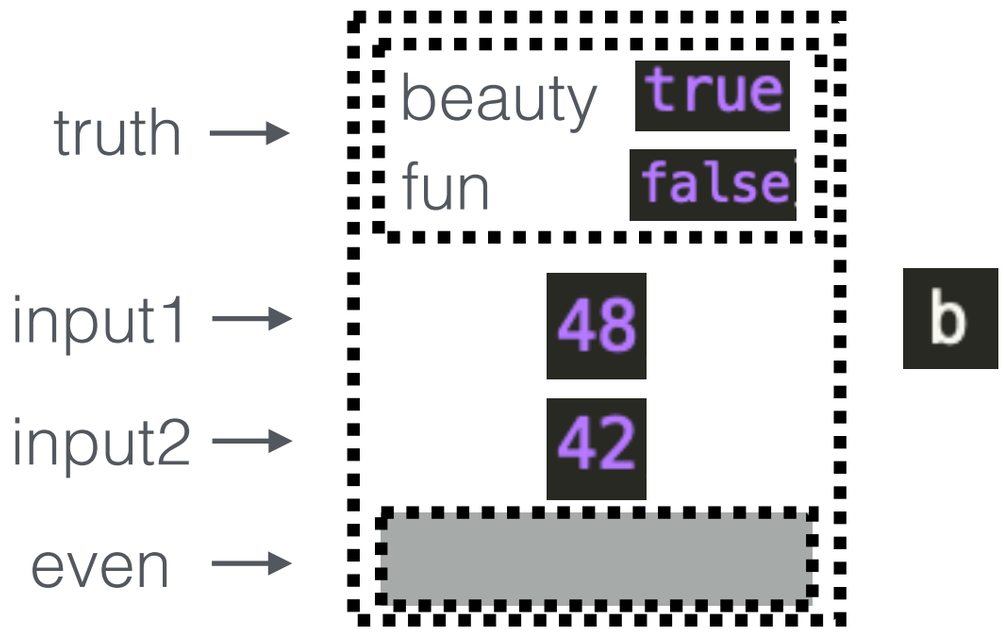
Objects



b

- Functions are objects
- Objects can contain objects
- Objects can contain functions: **methods**

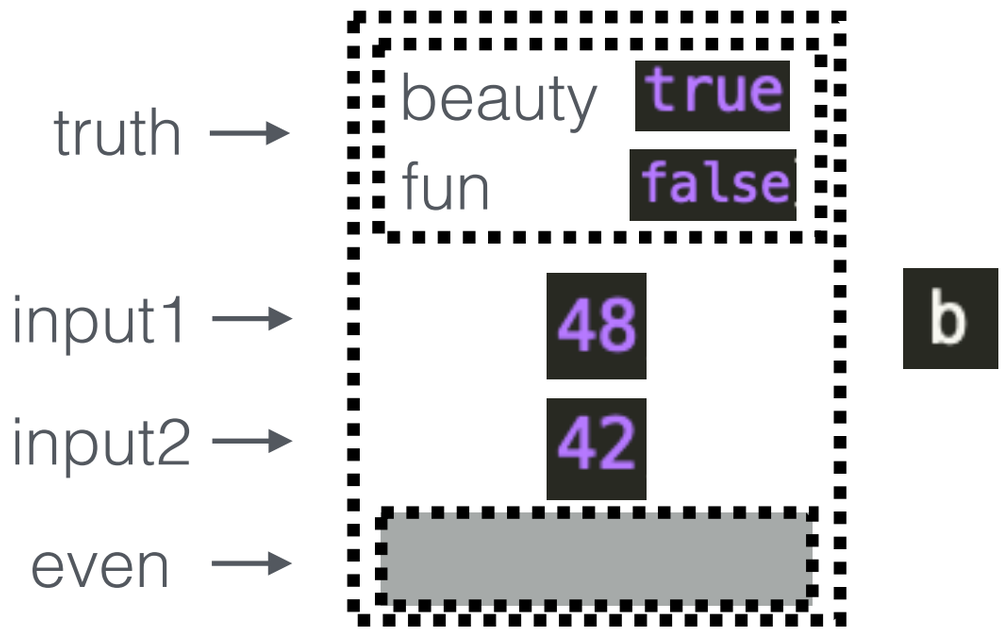
Methods




```
var b = {
    truth : {beauty : true, fun: false},
    input1 : 48,
    input2 : 42,
    even : function() {
        if ((this.input1 + this.input2) % 2 == 0) {
            return true;
        }
        return false;
    }
};
```

this refers to the “owner” object of function: the object outside of the function object

Methods



Alternative form

```
var b = {
    truth : {beauty : true, fun: false},
    input1 : 48,
    input2 : 42,
    even() {
        if ((this.input1 + this.input2) % 2 == 0) {
            return true;
        }
        return false;
    }
};
```

this refers to the “owner” object of function: the object outside of the function object

this

this is used mostly within the context function and is a specific keyword

Outside of this context, it refers to something called the global object, and things get complicated

Methods

```
var b = {
    truth : {beauty : true, fun: false},
    input1 : 48,
    input2 : 42,
    even : function() {
        if ((this.input1 + this.input2) % 2 == 0) {
            return true;
        }
        return false;
    }
};
```

To call method:

```
b.even();
```

Input parameters

Same syntax from before, but acknowledging it's a function

Methods

```
var b = {
    truth : {beauty : true, fun: false},
    input1 : 48,
    input2 : 42,
    even : function(n) {
        if (n % 2 == 0) {
            return true;
        }
        return false;
    }
};
```

To call method:

```
b.even(3);
```

Input parameters

Same syntax from before, but acknowledging it's a function

Methods

```
var b = {
    truth : {beauty : true, fun: false},
    input1 : 48,
    input2 : 42,
    even : function(n) {
        if (n % 2 == 0) {
            return true;
        }
        return false;
    }
};
```

Altering methods uses same syntax for altering properties

Creating new methods uses same syntax for creating new properties