WHY ARE SOME PROPERTIES DIMMED?

Before we start, I just want to reinforce that this is a side-topic – it is not crucial to understand the DOM in any way

I just thought it was interesting enough to share

WHY ARE SOME PROPERTIES DIMMED?

Before we start, I just want to reinforce that this is a side-topic – it is not crucial to understand the DOM in any way

I just thought it was interesting enough to share

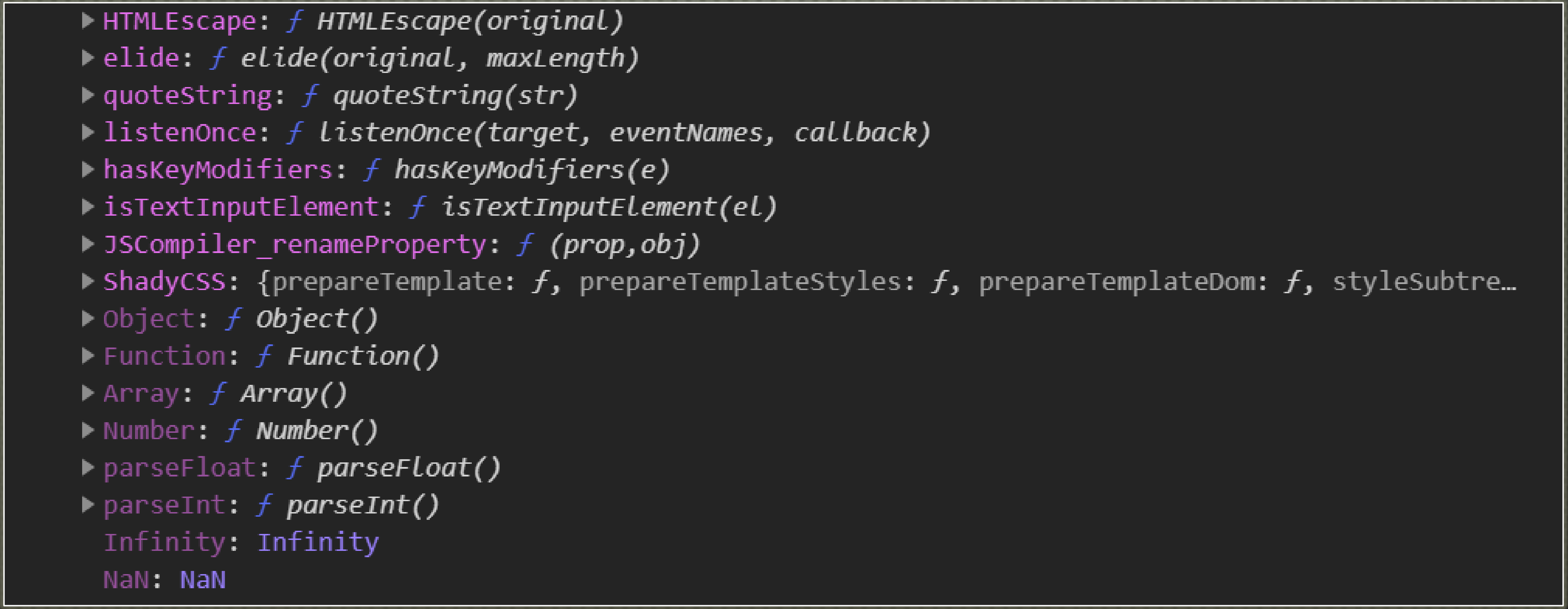
Feel free to skip this entire lecture if you want - I won’t be offended ;)

WHY ARE SOME PROPERTIES DIMMED?

Just type window into the console and you’ll see this

WHY ARE SOME PROPERTIES DIMMED?

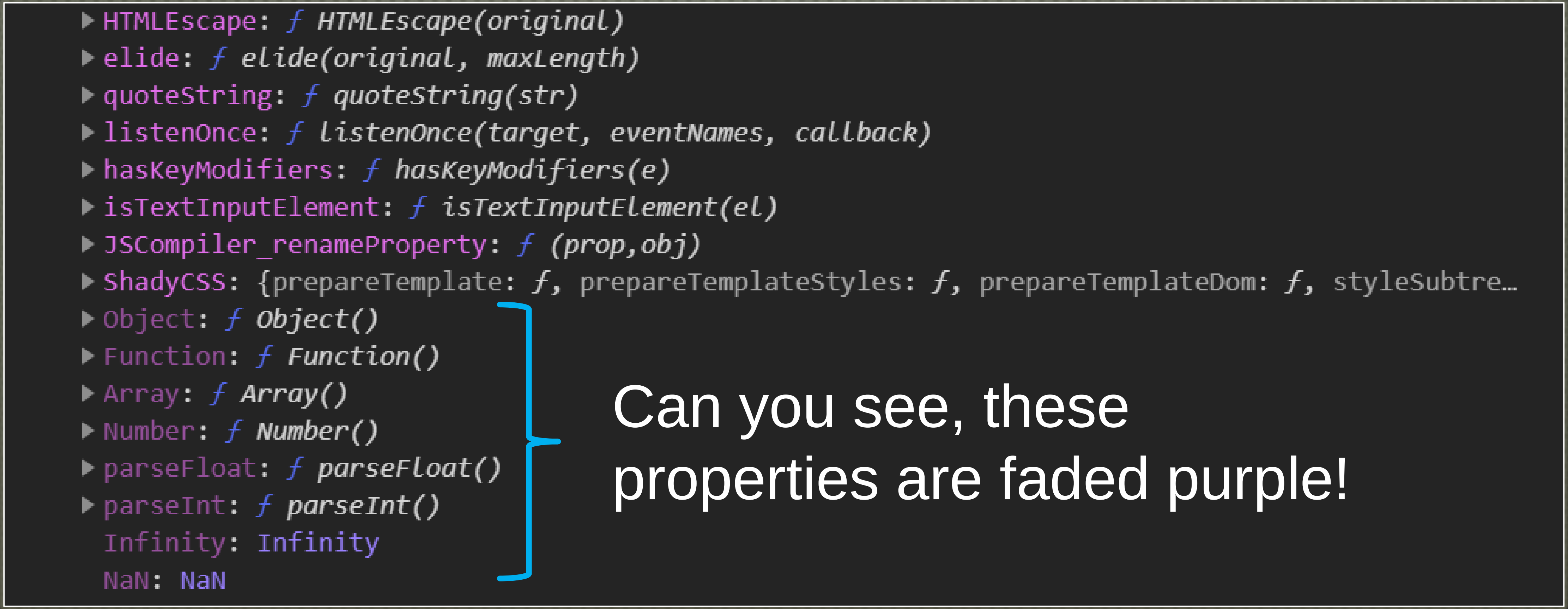
Just type window into the console and you'll see this



```
▶ HTMLEscape: ƒ HTMLEscape(original)
▶ elide: ƒ elide(original, maxLength)
▶ quoteString: ƒ quoteString(str)
▶ listenOnce: ƒ listenOnce(target, eventNames, callback)
▶ hasKeyModifiers: ƒ hasKeyModifiers(e)
▶ isTextInputElement: ƒ isTextInputElement(el)
▶ JSCompiler_renameProperty: ƒ (prop,obj)
▶ ShadyCSS: {prepareTemplate: ƒ, prepareTemplateStyles: ƒ, prepareTemplateDom: ƒ, styleSubtre…
▶ Object: ƒ Object()
▶ Function: ƒ Function()
▶ Array: ƒ Array()
▶ Number: ƒ Number()
▶ parseFloat: ƒ parseFloat()
▶ parseInt: ƒ parseInt()
  Infinity: Infinity
  NaN: NaN
```

WHY ARE SOME PROPERTIES DIMMED?

Just type window into the console and you'll see this



```
▶ HTMLEscape: ƒ HTMLEscape(original)
▶ elide: ƒ elide(original, maxLength)
▶ quoteString: ƒ quoteString(str)
▶ listenOnce: ƒ listenOnce(target, eventNames, callback)
▶ hasKeyModifiers: ƒ hasKeyModifiers(e)
▶ isTextInputElement: ƒ isTextInputElement(el)
▶ JSCompiler_renameProperty: ƒ (prop,obj)
▶ ShadyCSS: {prepareTemplate: ƒ, prepareTemplateStyles: ƒ, prepareTemplateDom: ƒ, styleSubtre…
▶ Object: ƒ Object()
▶ Function: ƒ Function()
▶ Array: ƒ Array()
▶ Number: ƒ Number()
▶ parseFloat: ƒ parseFloat()
▶ parseInt: ƒ parseInt()
  Infinity: Infinity
  NaN: NaN
```

Can you see, these properties are faded purple!

WHY ARE SOME PROPERTIES DIMMED?

I haven't been able to find the answer to this in any documentation or any official explanation

WHY ARE SOME PROPERTIES DIMMED?

I haven't been able to find the answer to this in any documentation or any official explanation

There is not much said about this on the web

WHY ARE SOME PROPERTIES DIMMED?

I haven't been able to find the answer to this in any documentation or any official explanation

There is not much said about this on the web

But with some tests, we can come up with a good guess as to why some properties are grayed

WHY ARE SOME PROPERTIES DIMMED?

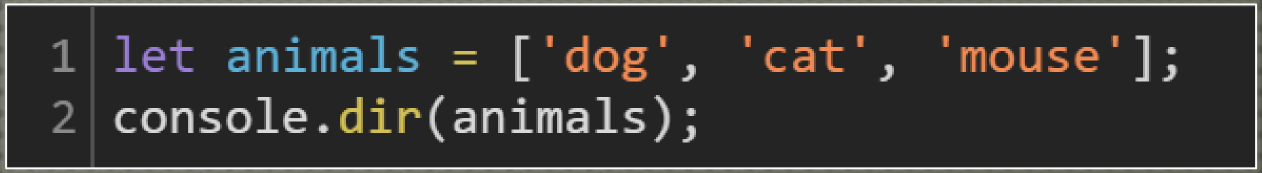
quick example

Lets create a simple array

WHY ARE SOME PROPERTIES DIMMED?

quick example

Lets create a simple array

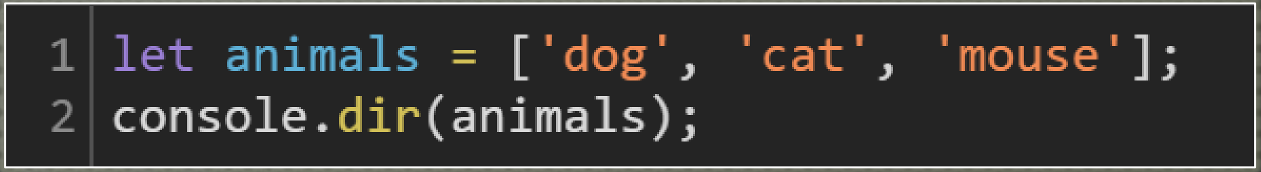


```
let animals = ['dog', 'cat', 'mouse'];
console.dir(animals);
```

WHY ARE SOME PROPERTIES DIMMED?

quick example

Lets create a simple array



```
let animals = ['dog', 'cat', 'mouse'];
console.dir(animals);
```

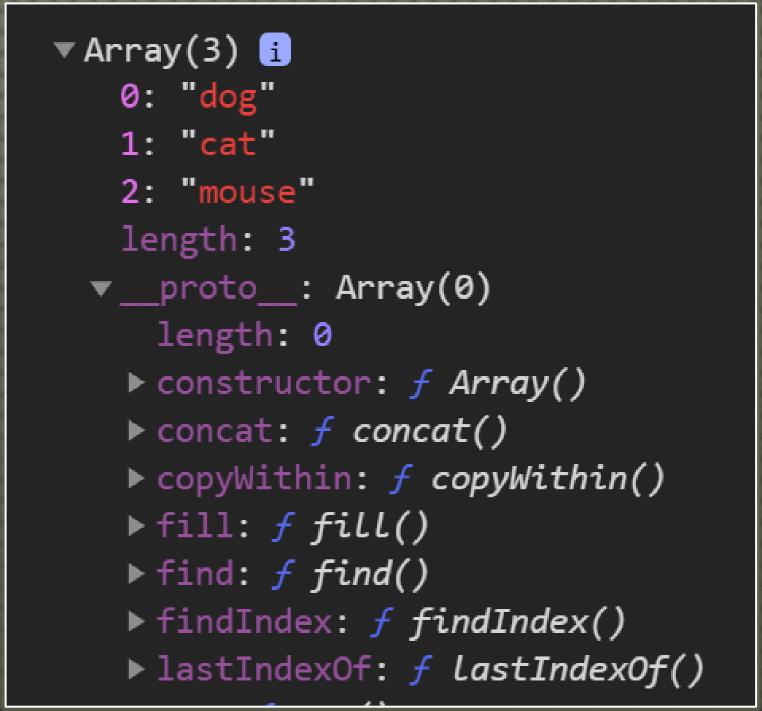
And if we console this out, we get:

WHY ARE SOME PROPERTIES DIMMED? quick example

Lets create a simple array

```
let animals = ['dog', 'cat', 'mouse'];
console.dir(animals);
```

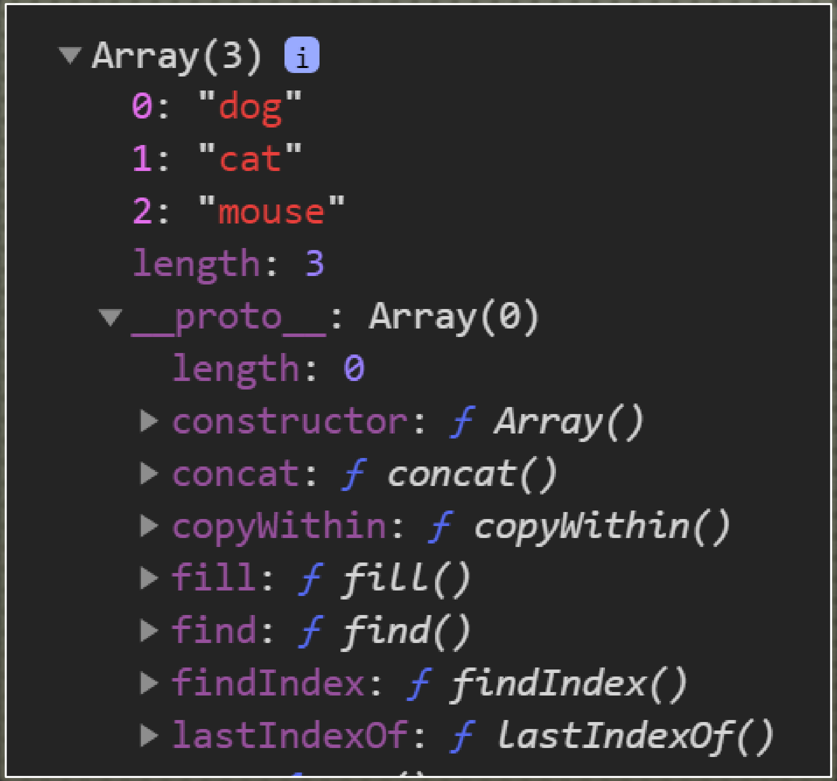
And if we console this out, we get:



```
▼ Array(3) i
    0: "dog"
    1: "cat"
    2: "mouse"
    length: 3
  ▼ __proto__: Array(0)
      length: 0
    ▶ constructor: ƒ Array()
    ▶ concat: ƒ concat()
    ▶ copyWithin: ƒ copyWithin()
    ▶ fill: ƒ fill()
    ▶ find: ƒ find()
    ▶ findIndex: ƒ findIndex()
    ▶ lastIndexOf: ƒ lastIndexOf()
```

WHY ARE SOME PROPERTIES DIMMED?

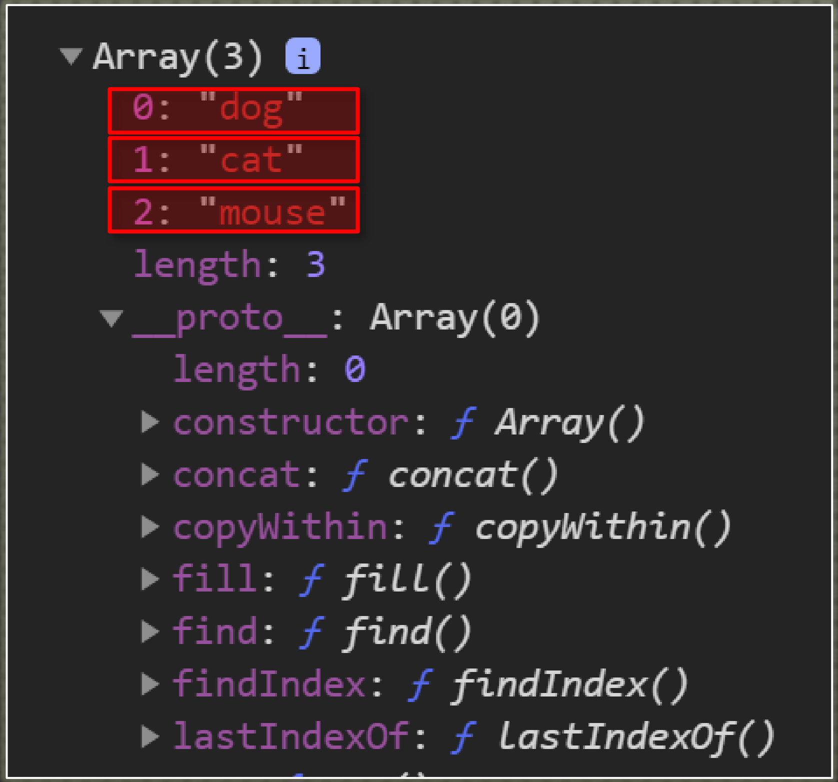
quick example



```
▼ Array(3) i
    0: "dog"
    1: "cat"
    2: "mouse"
    length: 3
  ▼ __proto__: Array(0)
      length: 0
    ▶ constructor: ƒ Array()
    ▶ concat: ƒ concat()
    ▶ copyWithin: ƒ copyWithin()
    ▶ fill: ƒ fill()
    ▶ find: ƒ find()
    ▶ findIndex: ƒ findIndex()
    ▶ lastIndexOf: ƒ lastIndexOf()
```

WHY ARE SOME PROPERTIES DIMMED?

quick example

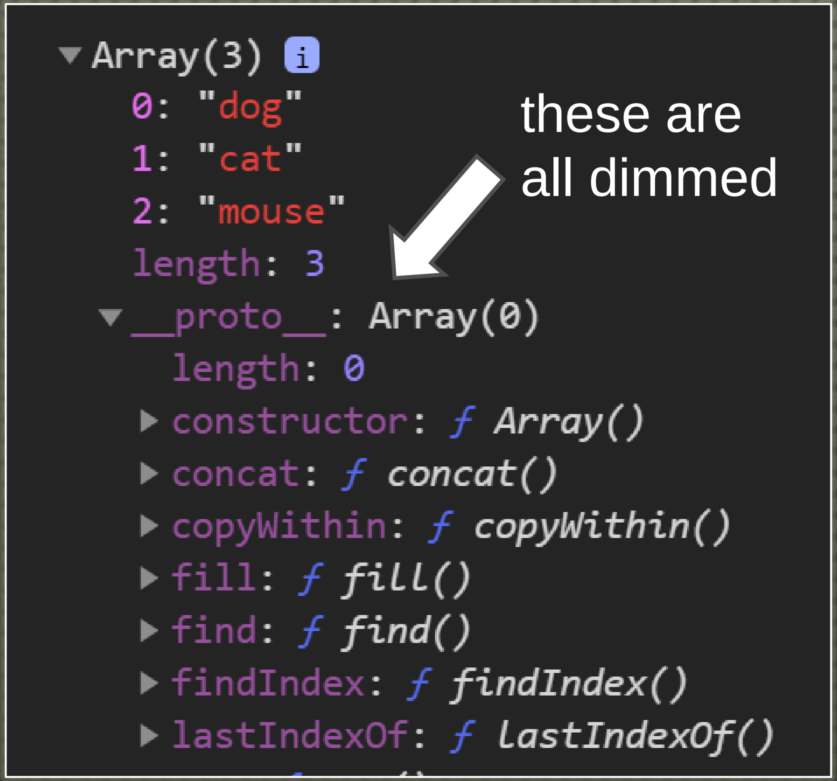


```
▼ Array(3) i
    0: "dog"
    1: "cat"
    2: "mouse"
    length: 3
  ▼ __proto__: Array(0)
      length: 0
    ▶ constructor: ƒ Array()
    ▶ concat: ƒ concat()
    ▶ copyWithin: ƒ copyWithin()
    ▶ fill: ƒ fill()
    ▶ find: ƒ find()
    ▶ findIndex: ƒ findIndex()
    ▶ lastIndexOf: ƒ lastIndexOf()
```

The indices are not grayed out

WHY ARE SOME PROPERTIES DIMMED?

quick example



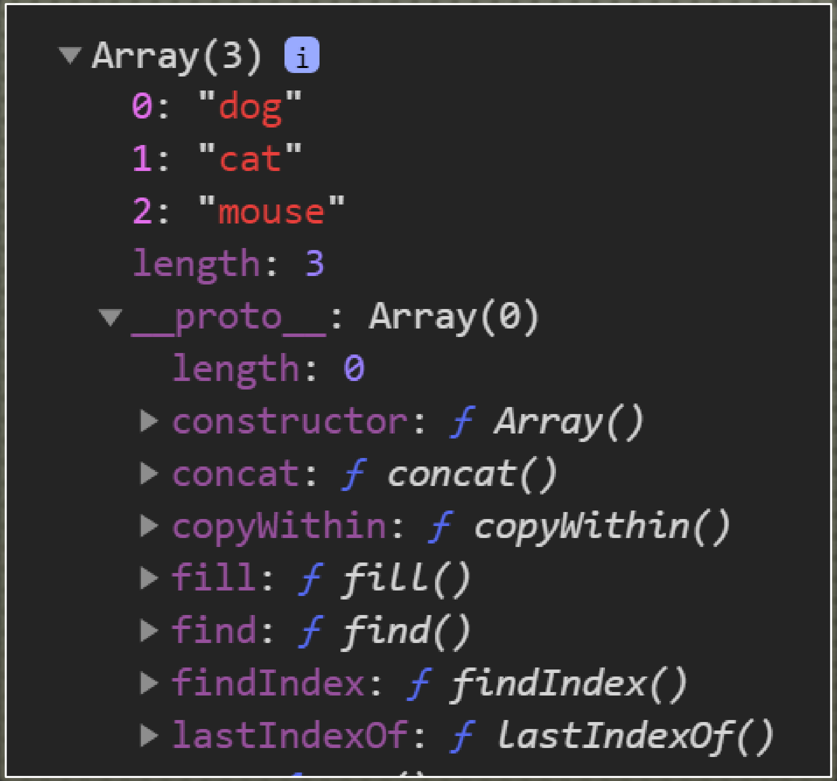
```
▼ Array(3) i
    0: "dog"
    1: "cat"
    2: "mouse"
    length: 3
  ▼ __proto__: Array(0)
      length: 0
    ▶ constructor: ƒ Array()
    ▶ concat: ƒ concat()
    ▶ copyWithin: ƒ copyWithin()
    ▶ fill: ƒ fill()
    ▶ find: ƒ find()
    ▶ findIndex: ƒ findIndex()
    ▶ lastIndexOf: ƒ lastIndexOf()
```

The indices are not grayed out

But the length and __proto__ properties are grayed out

WHY ARE SOME PROPERTIES DIMMED?

quick example



```
▼ Array(3) i
    0: "dog"
    1: "cat"
    2: "mouse"
    length: 3
  ▼ __proto__: Array(0)
      length: 0
    ▶ constructor: ƒ Array()
    ▶ concat: ƒ concat()
    ▶ copyWithin: ƒ copyWithin()
    ▶ fill: ƒ fill()
    ▶ find: ƒ find()
    ▶ findIndex: ƒ findIndex()
    ▶ lastIndexOf: ƒ lastIndexOf()
```

The indices are not grayed out

But the length and __proto__ properties are

Before we look at the __proto__, lets look at why the length property is dimmed

WHY ARE SOME PROPERTIES DIMMED?

enumerable

The starting point is to understand that in JavaScript, properties may be enumerable or not

WHY ARE SOME PROPERTIES DIMMED?

The starting point is to understand that in JavaScript, properties may be enumerable or not

Don’t stress

WHY ARE SOME PROPERTIES DIMMED?

enumerable

The starting point is to understand that in JavaScript, properties may be enumerable or not

Don’t stress

An enumerable property is one that can be included in and used in a for..in loop (or something similar)

WHY ARE SOME PROPERTIES DIMMED?

enumerable

The starting point is to understand that in JavaScript, properties may be enumerable or not

Don’t stress

An enumerable property is one that can be included in and used in a for..in loop (or something similar)

In JavaScript, many properties are non-enumerable, especially properties of prototypes

WHY ARE SOME PROPERTIES DIMMED?

enumerable

The starting point is to understand that in JavaScript, properties may be enumerable or not

Don’t stress

An enumerable propert

and used in a for..in loo

For example, this is why the for-in does not list all of the methods on Object.prototype for every object

In JavaScript, many properties are non-enumerable, especially properties of prototypes

WHY ARE SOME PROPERTIES DIMMED?

enumerable

Lets prove that a property that is dimmed means it's not enumerable

WHY ARE SOME PROPERTIES DIMMED?

enumerable

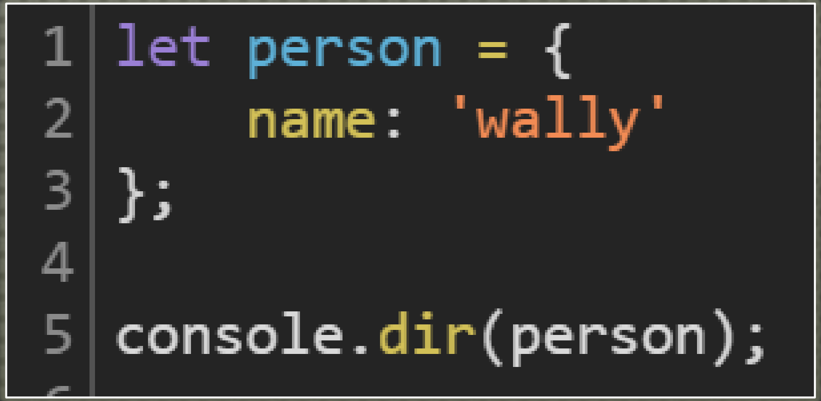
Lets prove that a property that is dimmed means it's not enumerable

First, lets create our own object

WHY ARE SOME PROPERTIES DIMMED? enumerable

Lets prove that a property that is dimmed means it's not enumerable

First, lets create our own object

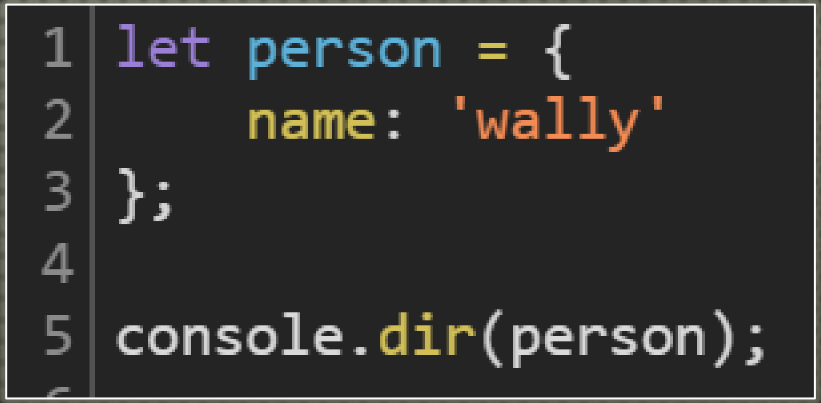


```
let person = {
    name: 'wally'
};

console.dir(person);
```

WHY ARE SOME PROPERTIES DIMMED? enumerable

Lets prove that a property that is dimmed means it's not enumerable

First, lets create our own object



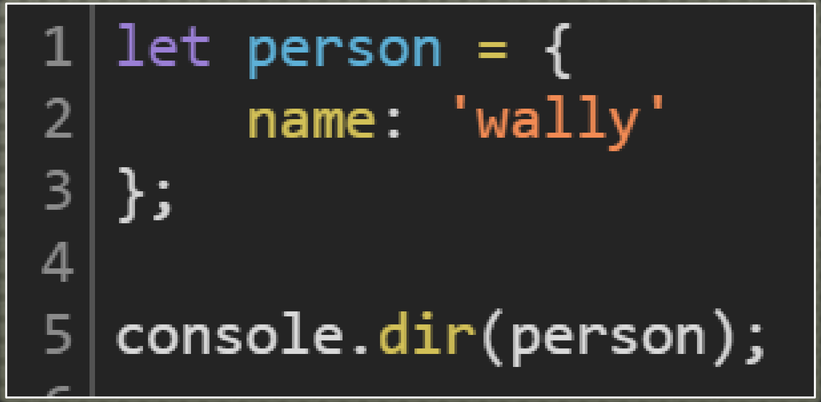
```
let person = {
    name: 'wally'
};

console.dir(person);
```

Then lets look at the console

WHY ARE SOME PROPERTIES DIMMED? enumerable

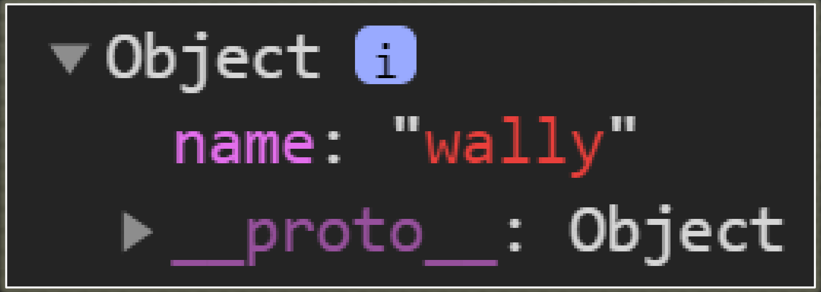
Lets prove that a property that is dimmed means it's not enumerable

First, lets create our own object



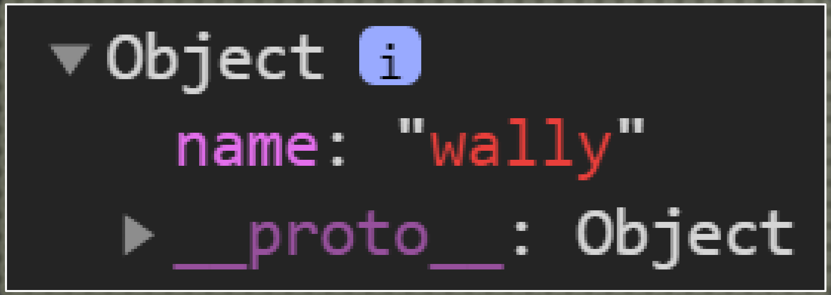
```
let person = {
    name: 'wally'
};

console.dir(person);
```

Then lets look at the console



```
▼ Object i
    name: "wally"
  ▶ __proto__: Object
```

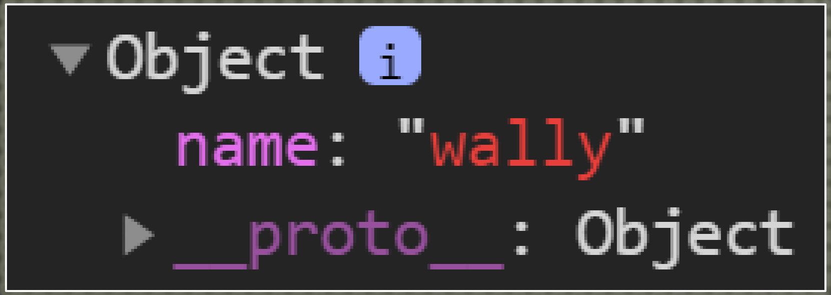
WHY ARE SOME PROPERTIES DIMMED? enumerable



```
▼ Object
    name: "wally"
  ▶ __proto__: Object
```

WHY ARE SOME PROPERTIES DIMMED?

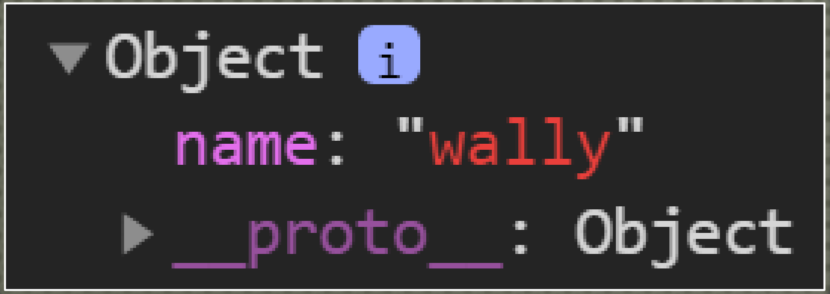
enumerable



```
▼ Object i
    name: "wally"
  ▶ __proto__: Object
```

The name is bright purple (this is because by default, custom properties are enumerable)

WHY ARE SOME PROPERTIES DIMMED? enumerable



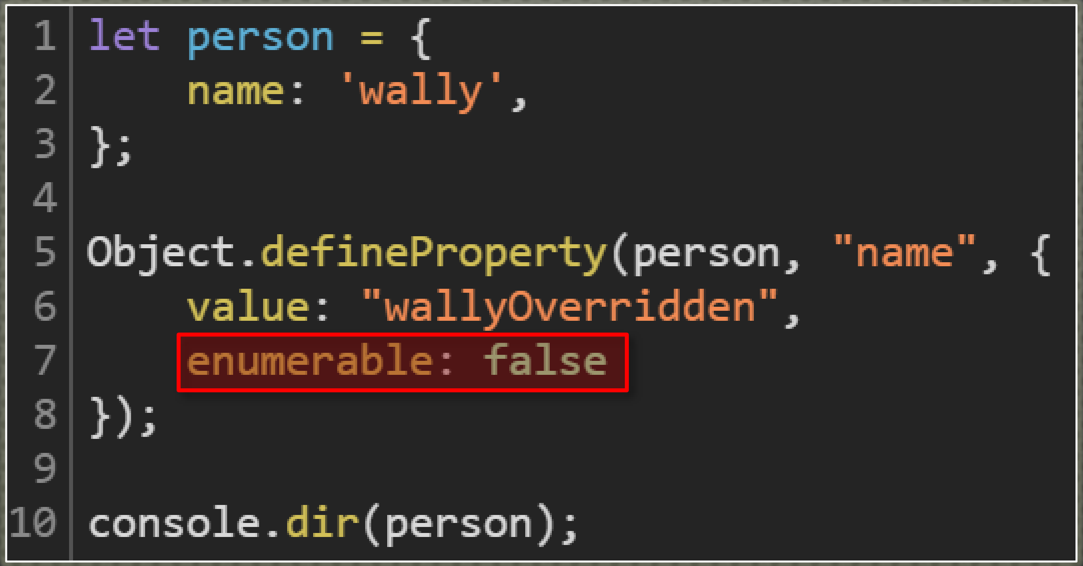
```
▼ Object ℹ
    name: "wally"
  ▶ __proto__: Object
```

The name is bright purple (this is because by default, custom properties are enumerable)

Lets change this by using the defineProperty method

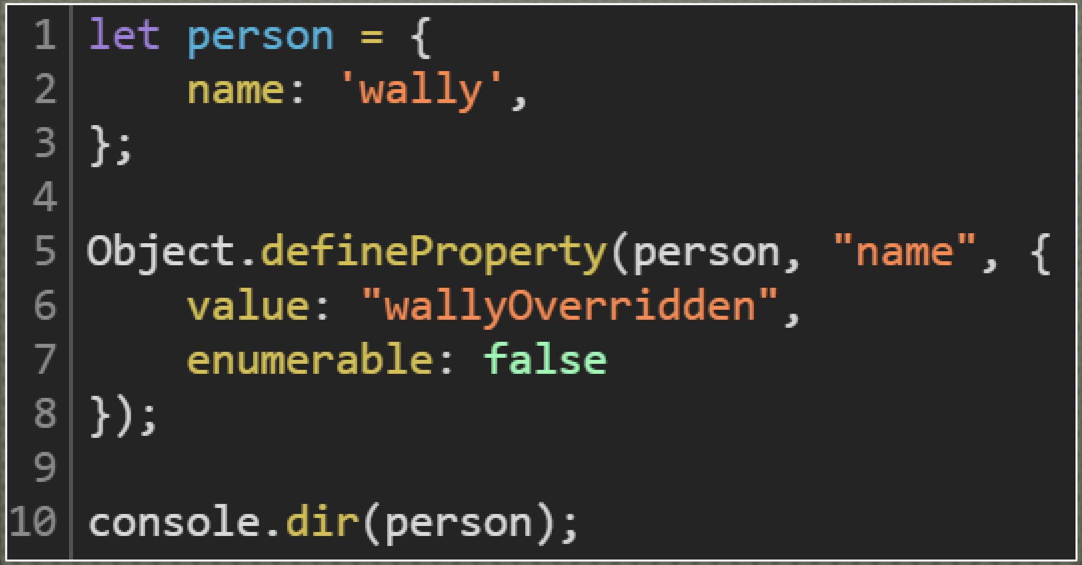
WHY ARE SOME PROPERTIES DIMMED?

enumerable



```
let person = {
    name: 'wally',
};

Object.defineProperty(person, "name", {
    value: "wallyOverridden",
    enumerable: false
});

console.dir(person);
```

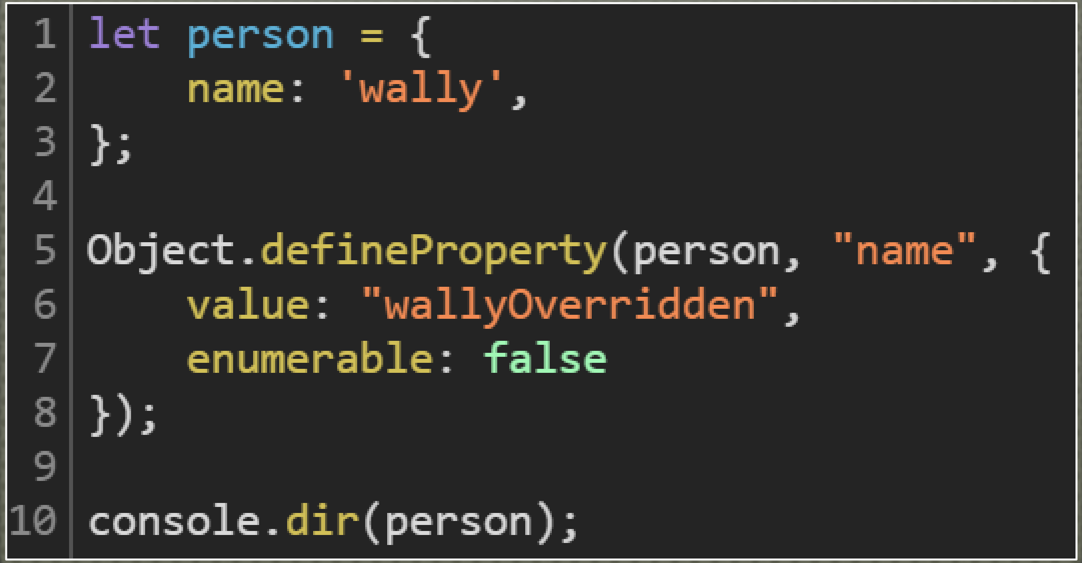
WHY ARE SOME PROPERTIES DIMMED? enumerable



```
let person = {
    name: 'wally',
};

Object.defineProperty(person, "name", {
    value: "wallyOverridden",
    enumerable: false
});

console.dir(person);
```

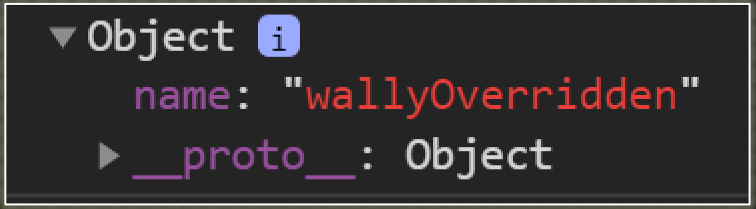
Now lets console this to the screen

WHY ARE SOME PROPERTIES DIMMED? enumerable



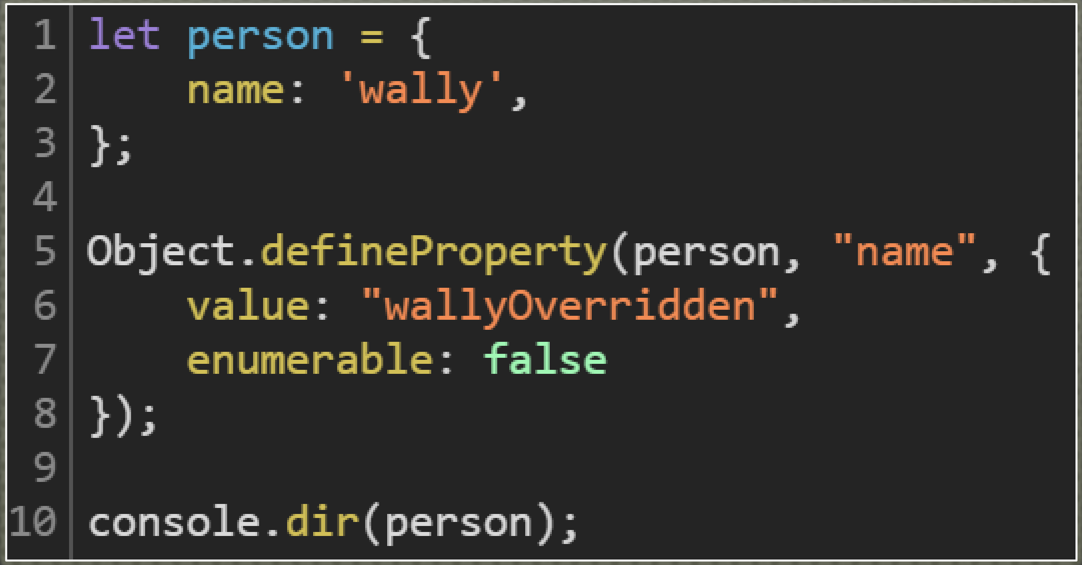
```
let person = {
    name: 'wally',
};

Object.defineProperty(person, "name", {
    value: "wallyOverridden",
    enumerable: false
});

console.dir(person);
```

Now lets console this to the screen



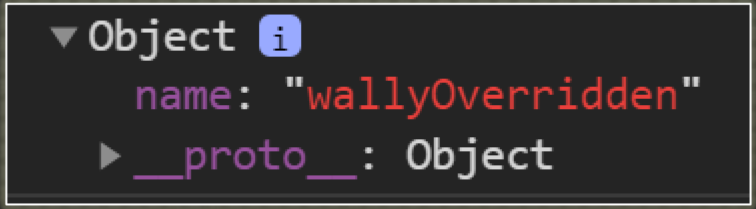
```
▼ Object i
    name: "wallyOverridden"
  ▶ __proto__: Object
```

WHY ARE SOME PROPERTIES DIMMED? enumerable



```
let person = {
     name: 'wally',
};

Object.defineProperty(person, "name", {
     value: "wallyOverridden",
     enumerable: false
});

console.dir(person);
```

Now lets console this to the screen



```
▼ Object i
    name: "wallyOverridden"
  ▶ __proto__: Object
```

The name property is now dimmed

WHY ARE SOME PROPERTIES DIMMED?

enumerable

Is this proof enough?

WHY ARE SOME PROPERTIES DIMMED?

enumerable

Is this proof enough?

No?

WHY ARE SOME PROPERTIES DIMMED? enumerable

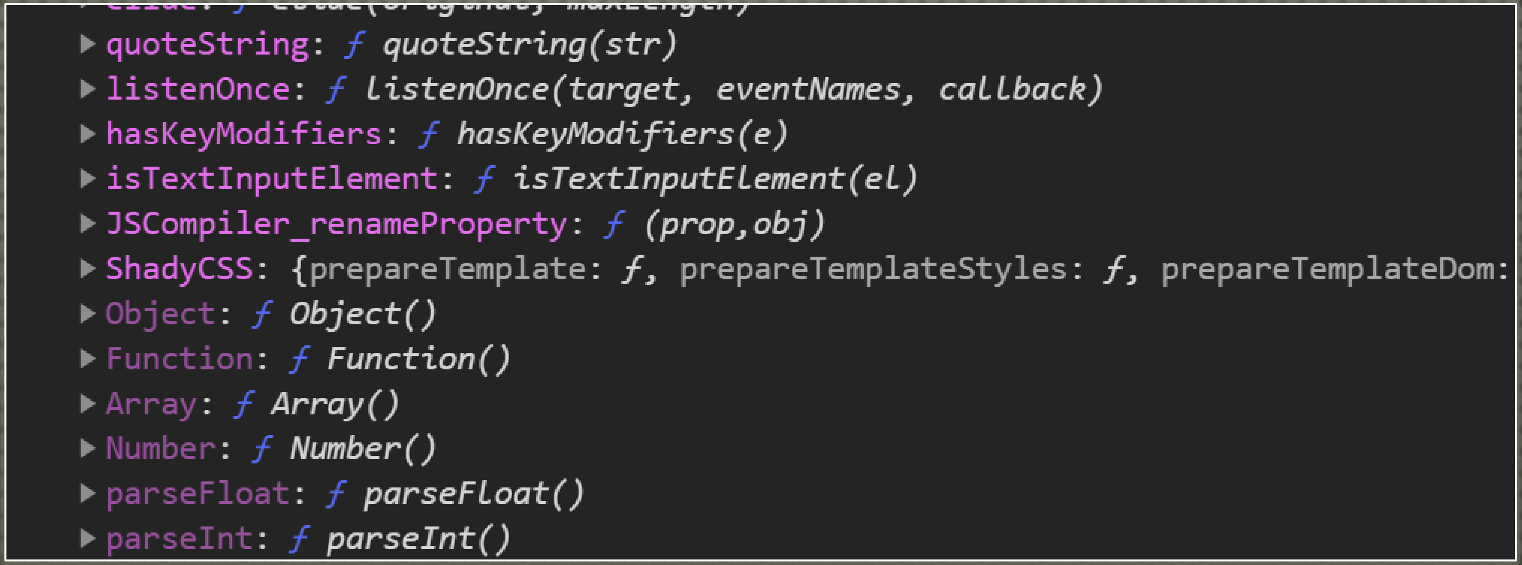
Is this proof enough?

No?

Then lets console the Window object and look at all of its properties

WHY ARE SOME PROPERTIES DIMMED?

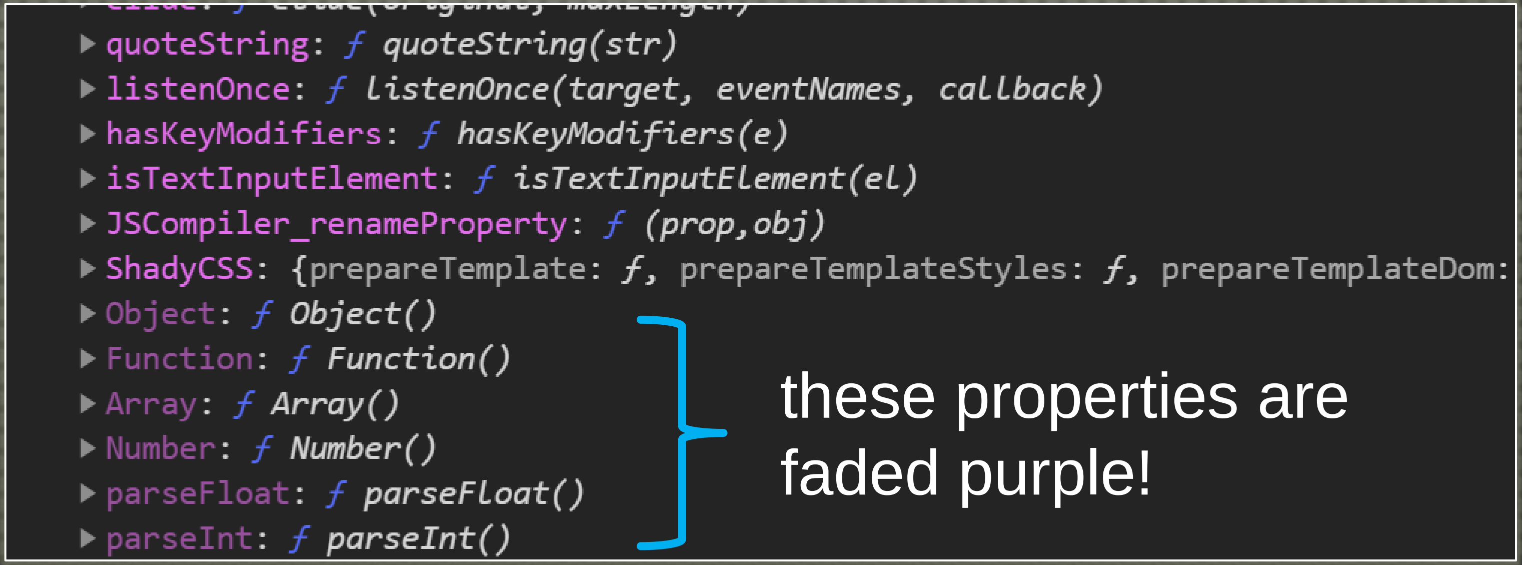
enumerable



```
▶ elide: ƒ elide(original, maxLength)
▶ quoteString: ƒ quoteString(str)
▶ listenOnce: ƒ listenOnce(target, eventNames, callback)
▶ hasKeyModifiers: ƒ hasKeyModifiers(e)
▶ isTextInputElement: ƒ isTextInputElement(el)
▶ JSCompiler_renameProperty: ƒ (prop,obj)
▶ ShadyCSS: {prepareTemplate: ƒ, prepareTemplateStyles: ƒ, prepareTemplateDom:
▶ Object: ƒ Object()
▶ Function: ƒ Function()
▶ Array: ƒ Array()
▶ Number: ƒ Number()
▶ parseFloat: ƒ parseFloat()
▶ parseInt: ƒ parseInt()
```

WHY ARE SOME PROPERTIES DIMMED?

enumerable



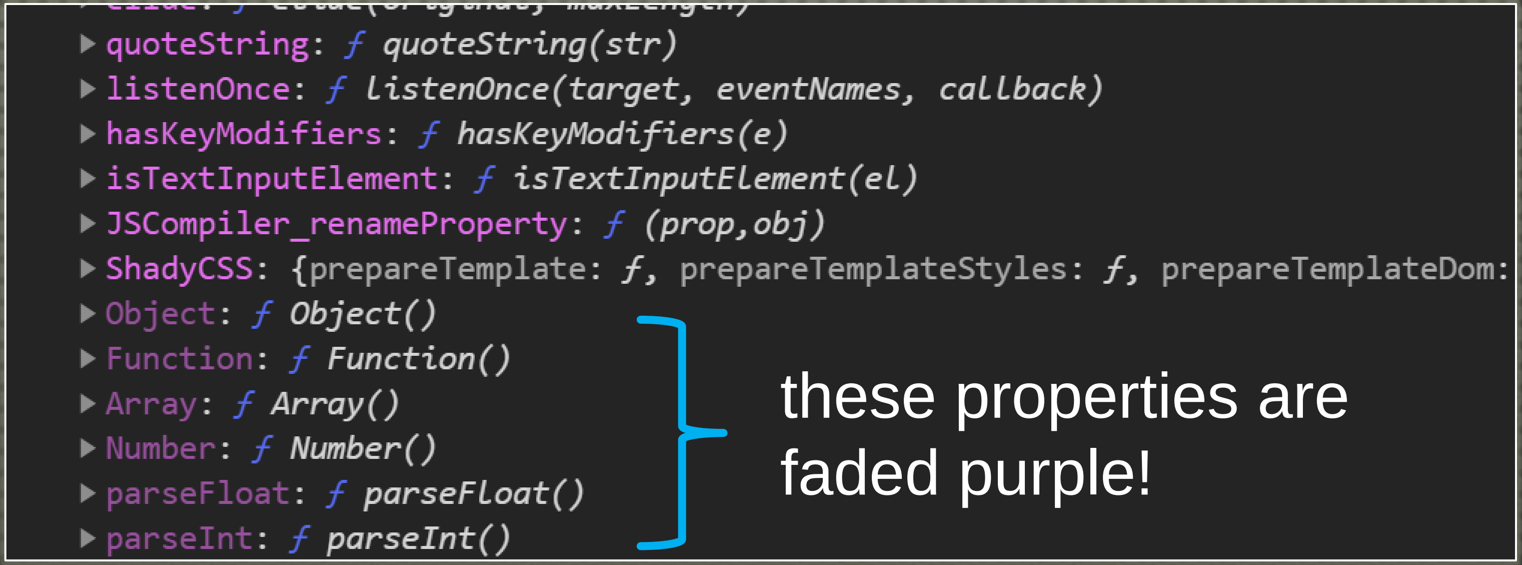
```
▸ elide: ƒ elide(original, maxLength)
▸ quoteString: ƒ quoteString(str)
▸ listenOnce: ƒ listenOnce(target, eventNames, callback)
▸ hasKeyModifiers: ƒ hasKeyModifiers(e)
▸ isTextInputElement: ƒ isTextInputElement(el)
▸ JSCompiler_renameProperty: ƒ (prop,obj)
▸ ShadyCSS: {prepareTemplate: ƒ, prepareTemplateStyles: ƒ, prepareTemplateDom:
▸ Object: ƒ Object()
▸ Function: ƒ Function()
▸ Array: ƒ Array()
▸ Number: ƒ Number()
▸ parseFloat: ƒ parseFloat()
▸ parseInt: ƒ parseInt()
```

these properties are faded purple!

Scroll down, and the bottom half are all dimmed out

WHY ARE SOME PROPERTIES DIMMED?

enumerable



```
▸ quoteString: ƒ quoteString(str)
▸ listenOnce: ƒ listenOnce(target, eventNames, callback)
▸ hasKeyModifiers: ƒ hasKeyModifiers(e)
▸ isTextInputElement: ƒ isTextInputElement(el)
▸ JSCompiler_renameProperty: ƒ (prop,obj)
▸ ShadyCSS: {prepareTemplate: ƒ, prepareTemplateStyles: ƒ, prepareTemplateDom:
▸ Object: ƒ Object()
▸ Function: ƒ Function()
▸ Array: ƒ Array()
▸ Number: ƒ Number()
▸ parseFloat: ƒ parseFloat()
▸ parseInt: ƒ parseInt()
```

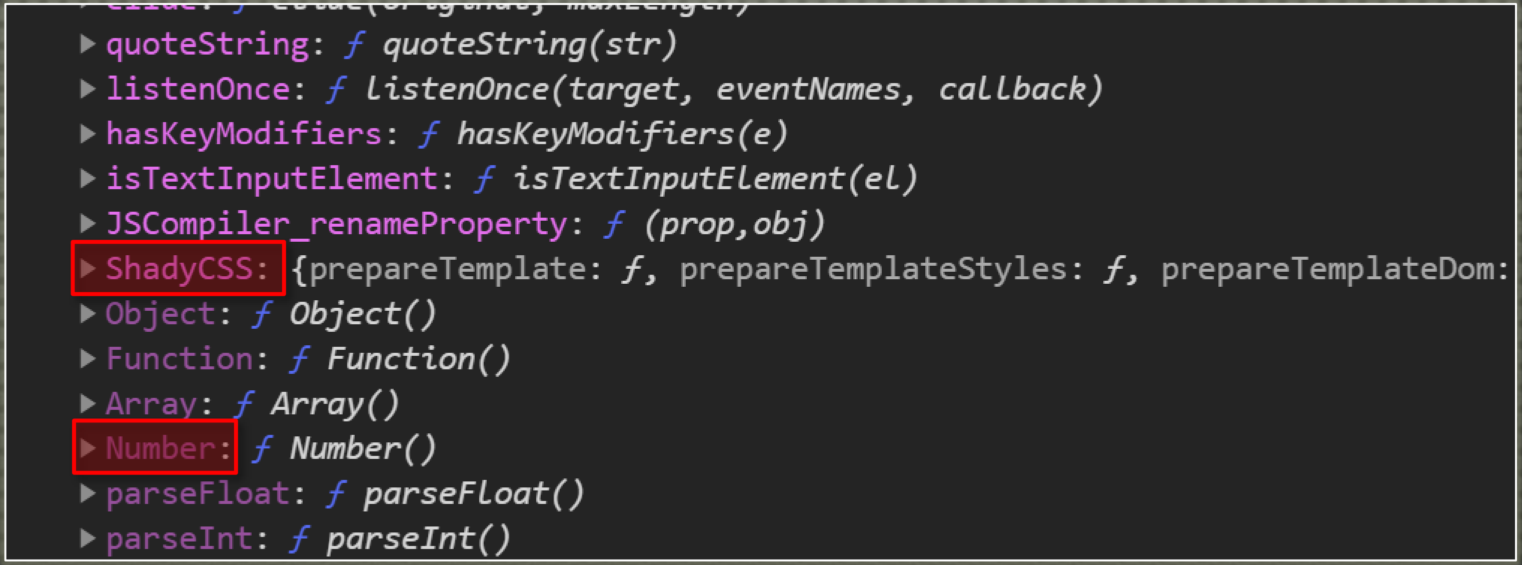
these properties are faded purple!

Scroll down, and the bottom half are all dimmed out

Lets test to see if they are enumerable …

WHY ARE SOME PROPERTIES DIMMED?

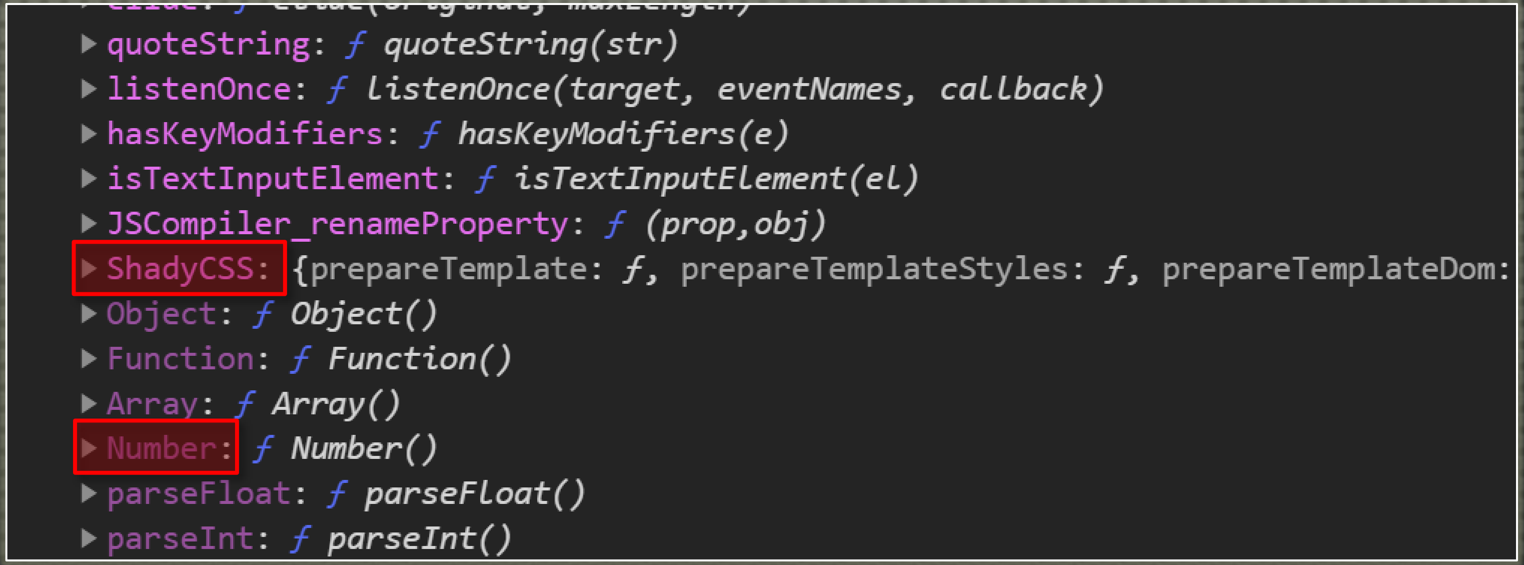
enumerable



```
▶ quoteString: ƒ quoteString(str)
▶ listenOnce: ƒ listenOnce(target, eventNames, callback)
▶ hasKeyModifiers: ƒ hasKeyModifiers(e)
▶ isTextInputElement: ƒ isTextInputElement(el)
▶ JSCompiler_renameProperty: ƒ (prop,obj)
▶ ShadyCSS: {prepareTemplate: ƒ, prepareTemplateStyles: ƒ, prepareTemplateDom:
▶ Object: ƒ Object()
▶ Function: ƒ Function()
▶ Array: ƒ Array()
▶ Number: ƒ Number()
▶ parseFloat: ƒ parseFloat()
▶ parseInt: ƒ parseInt()
```

Lets randomly just pick ShadyCSS and Number

WHY ARE SOME PROPERTIES DIMMED? enumerable



```
▶ quoteString: ƒ quoteString(str)
▶ listenOnce: ƒ listenOnce(target, eventNames, callback)
▶ hasKeyModifiers: ƒ hasKeyModifiers(e)
▶ isTextInputElement: ƒ isTextInputElement(el)
▶ JSCompiler_renameProperty: ƒ (prop,obj)
▶ ShadyCSS: {prepareTemplate: ƒ, prepareTemplateStyles: ƒ, prepareTemplateDom:
▶ Object: ƒ Object()
▶ Function: ƒ Function()
▶ Array: ƒ Array()
▶ Number: ƒ Number()
▶ parseFloat: ƒ parseFloat()
▶ parseInt: ƒ parseInt()
```

Lets randomly just pick ShadyCSS and Number

ShadyCSS should be enumberable, Number shouldn't

WHY ARE SOME PROPERTIES DIMMED?

enumerable

You can use propertyIsEnumerable to find out if the property will show up when you loop over the object (i.e. to find out if it is enumerable)

WHY ARE SOME PROPERTIES DIMMED? enumerable

You can use propertyIsEnumerable to find out if the property will show up when you loop over the object (i.e. to find out if it is enumerable)

```
window.propertyIsEnumerable("ShadyCSS")
// true
```

WHY ARE SOME PROPERTIES DIMMED? enumerable

You can use propertyIsEnumerable to find out if the property will show up when you loop over the object (i.e. to find out if it is enumerable)

```
window.propertyIsEnumerable("ShadyCSS")
// true

window.propertyIsEnumerable(“Number")
// false
```

WHY ARE SOME PROPERTIES DIMMED?

enumerable

yippee

WHY ARE SOME PROPERTIES DIMMED? enumerable

yippee

We’ve just shown that the dimmed out / grayed properties are all non-enumerable

WHY ARE SOME PROPERTIES DIMMED? enumerable

yippee

We’ve just shown that the dimmed out / grayed properties are all non-enumerable

But wait … there’s more

WHY ARE SOME PROPERTIES DIMMED?

Lets build our own custom object, and prototype and constructor

WHY ARE SOME PROPERTIES DIMMED?

Lets build our own custom object, and prototype and constructor

Don’t worry about all these concepts. I’m trying to prove a point so concentrate on the ‘bigger picture’

WHY ARE SOME PROPERTIES DIMMED?

prototype

Lets build our own custom object, and prototype and constructor

Don’t worry about all these concepts. I’m trying to prove a point so concentrate on the ‘bigger picture’

Are you ready?

WHY ARE SOME PROPERTIES DIMMED?

I don’t want to go into depth in the actual code itself

WHY ARE SOME PROPERTIES DIMMED?

prototype

I don’t want to go into depth in the actual code itself

I’ll show you what I did, so if you’re interested you can code it up yourself

WHY ARE SOME PROPERTIES DIMMED?

I don’t want to go into depth in the actual code itself

I’ll show you what I did, so if you’re interested you can code it up yourself

Again, don’t worry about not understanding the code. That’s not the point of this lecture

WHY ARE SOME PROPERTIES DIMMED?

prototype

Here’s the code

WHY ARE SOME PROPERTIES DIMMED? *prototype*

Here's the code

```
let Animal = function() {
    this.eatsGrass = true;
};

Animal.prototype.greet = function() {
    if(this.eatsGrass) {
        console.log(`Hi there. I eat grass.`)
    }
};

let Zebra = function(name, title) {
    Animal.call(this);
    this.name=name;
    this.title = title;
};
```

```
Zebra.prototype = Object.create(Animal.prototype);

Zebra.prototype.constructor = Zebra;

Zebra.prototype.greet = function() {
    if(this.eatsGrass) {
        console.log(`${this.title} ${this.name} eats grass`)
    }
};

let zebra = new Zebra('Harry', 'Mr');

console.dir(zebra);
```

WHY ARE SOME PROPERTIES DIMMED? prototype

Just in case you're wondering, here's what I did

defining our Animal constructor

```
let Animal = function() {
    this.eatsGrass = true;
};
```

adding a method to our prototype called 'greet'

```
Animal.prototype.greet =
    if(this.eatsGrass) {
        console.log(`Hi there
    }
};
```

creating a constructor function for our Zebra class

```
let Zebra = function(name, title) {
    Animal.call(this);
    this.name=name;
    this.title = title;
};
```

setting the prototype for Zebra to the Animal prototype

```
Zebra.prototype = Object.create(Animal.prototype);

Zebra.prototype.constructor = Z
```

lets override the Animal constructor with our own one

```
Zebra.prototype.greet = function() {
    if(this.eatsGrass) {
                              me} eats grass`)
    }
};
```

here we are also overriding the default 'greet' function set on the Animal class

```
let zebra = new Zebra('Harry', 'Mr');

console.dir(zebra);
```

Finally, we instantiate (i.e. create) a new variable called zebra, and console the object out

WHY ARE SOME PROPERTIES DIMMED?

Don’t worry about what this all means

WHY ARE SOME PROPERTIES DIMMED?

Don’t worry about what this all means

Our goal is to find out why some properties are dimmed and others aren’t

WHY ARE SOME PROPERTIES DIMMED?

Don’t worry about what this all means

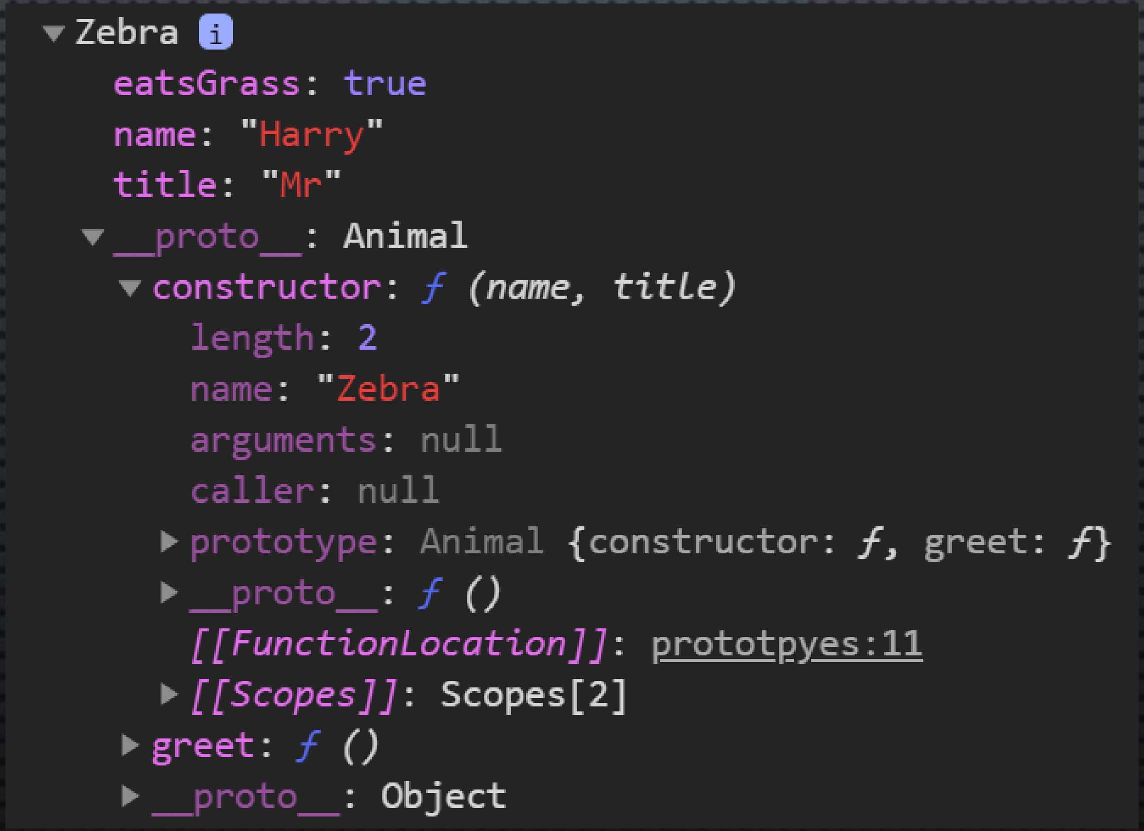
Our goal is to find out why some properties are dimmed and others aren’t

Lets log our zebra variable to the console and see what properties are grayed out and which aren’t'

WHY ARE SOME PROPERTIES DIMMED?

prototype

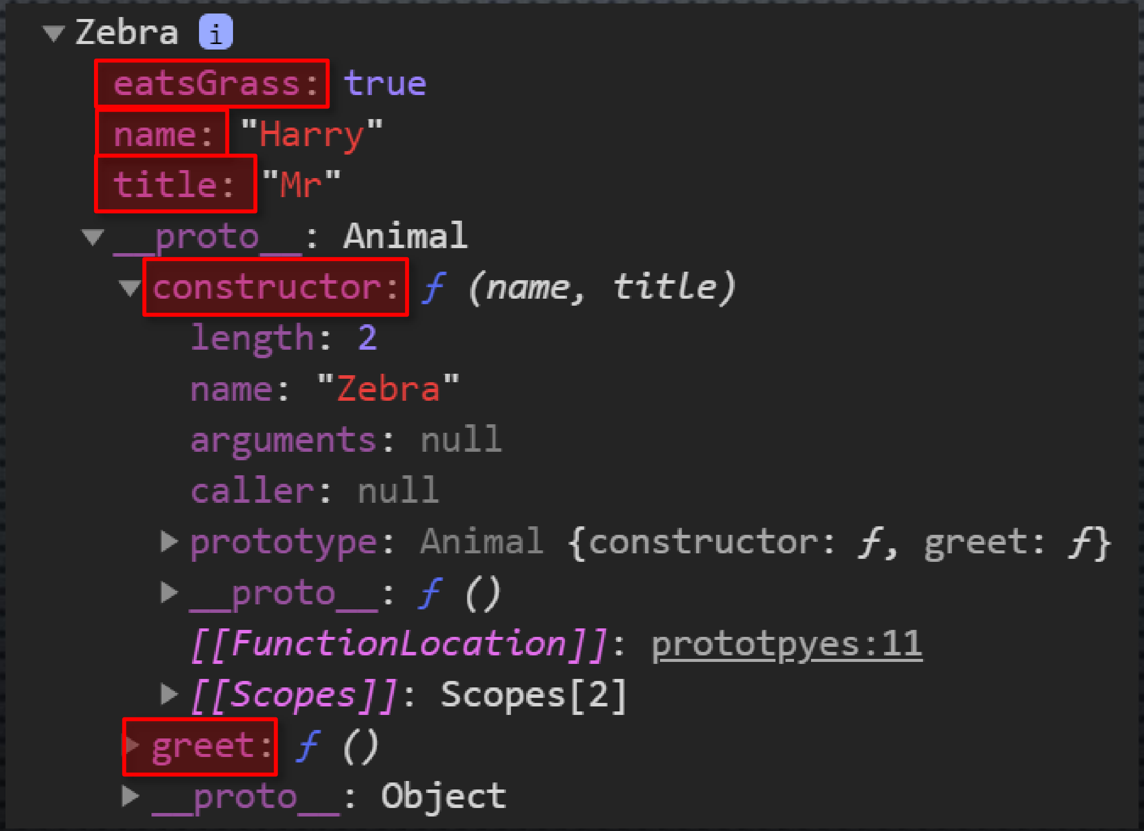
This is the result



```
▼ Zebra ℹ
    eatsGrass: true
    name: "Harry"
    title: "Mr"
  ▼ __proto__: Animal
    ▼ constructor: ƒ (name, title)
        length: 2
        name: "Zebra"
        arguments: null
        caller: null
      ▶ prototype: Animal {constructor: ƒ, greet: ƒ}
      ▶ __proto__: ƒ ()
        [[FunctionLocation]]: prototpyes:11
      ▶ [[Scopes]]: Scopes[2]
    ▶ greet: ƒ ()
    ▶ __proto__: Object
```

WHY ARE SOME PROPERTIES DIMMED?

This is the result



```
▼ Zebra ℹ
    eatsGrass: true
    name: "Harry"
    title: "Mr"
  ▼ __proto__: Animal
    ▼ constructor: ƒ (name, title)
        length: 2
        name: "Zebra"
        arguments: null
        caller: null
      ▶ prototype: Animal {constructor: ƒ, greet: ƒ}
      ▶ __proto__: ƒ ()
        [[FunctionLocation]]: prototpyes:11
      ▶ [[Scopes]]: Scopes[2]
    ▶ greet: ƒ ()
    ▶ __proto__: Object
```

WHY ARE SOME PROPERTIES DIMMED?

prototype

There are a lot of interesting things here

WHY ARE SOME PROPERTIES DIMMED?

prototype

There are a lot of interesting things here

All instances of greet are not grayed out

WHY ARE SOME PROPERTIES DIMMED?

prototype

There are a lot of interesting things here

All instances of greet are not grayed out

__proto__ is grayed

WHY ARE SOME PROPERTIES DIMMED?

prototype

There are a lot of interesting things here

All instances of greet are not grayed out

__proto__ is grayed

Also, the overridden constructor from Animal is grayed out, but the explicitly set constructor of Zebra is not grayed out

WHY ARE SOME PROPERTIES DIMMED?

So I think there is a bit of randomness here

WHY ARE SOME PROPERTIES DIMMED?

So I think there is a bit of randomness here

Chrom(e/ium) grays out properties that it thinks you're going to be less likely to care about, either because they were inherited or set by the engine as a construct of the language, but it seems its not perfect.

WHY ARE SOME PROPERTIES DIMMED?

So I think there is a bit of randomness here

Chrom(e/ium) grays out properties that it thinks you're going to be less likely to care about, either because they were inherited or set by the engine as a construct of the language, but it seems its not perfect.

But nobody is perfect, right?

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

Don’t get lost in all the detail

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

Don’t get lost in all the detail

I wanted to show you why some properties are grayed out by Chrome

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

Don’t get lost in all the detail

I wanted to show you why some properties are grayed out by Chrome

You’ve probably already forgot, so lets jump to the last summary slide next

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

The browser chooses to display some properties in light purple (dimmed out) in these circumstances:

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

The browser chooses to display some properties in light purple (dimmed out) in these circumstances:

1.
2.
3.
4.

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

The browser chooses to display some properties in light purple (dimmed out) in these circumstances:

1. Properties that are not enumerable are typically grayed out
2.
3.
4.

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

The browser chooses to display some properties in light purple (dimmed out) in these circumstances:

1. Properties that are not enumerable are typically grayed out
2. Prototypes seem to be greyed out
3.
4.

WHY ARE SOME PROPERTIES DIMMED? SUMMARY

The browser chooses to display some properties in light purple (dimmed out) in these circumstances:

1. Properties that are not enumerable are typically grayed out
2. Prototypes seem to be greyed out
3. Constructors seem to be greyed out, except when we explicitly define them
4.

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

The browser chooses to display some properties in light purple (dimmed out) in these circumstances:

1. Properties that are not enumerable are typically grayed out
2. Prototypes seem to be greyed out
3. Constructors seem to be greyed out, except when we explicitly define them
4. Properties that chrome thinks you’ll not care about that much

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

end.

WHY ARE SOME PROPERTIES DIMMED?

SUMMARY

end.

Please don’t forget to leave me a review – it helps me tremendously.