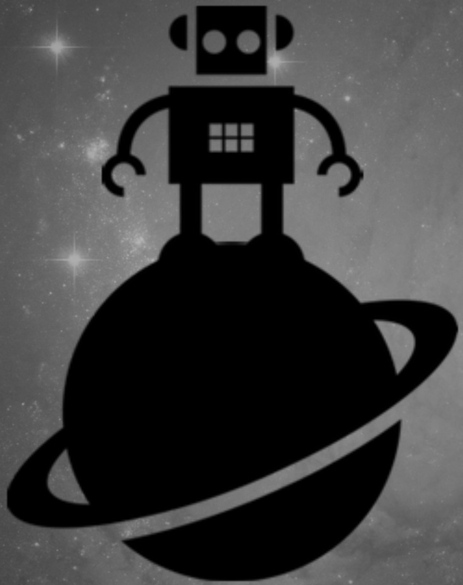
PlanetProto



Cleveland JavaScript Group

November 21, 2016

Thanks to

LeanDog AN AGILE SOFTWARE STUDIO

Rockwell Automation LISTEN. THINK. SOLVE.℠

HOUSEKEEPING

Bathrooms

Food

No meetup in December

Vote for sessions on meetup.com

Call for presenters

Planet Proto

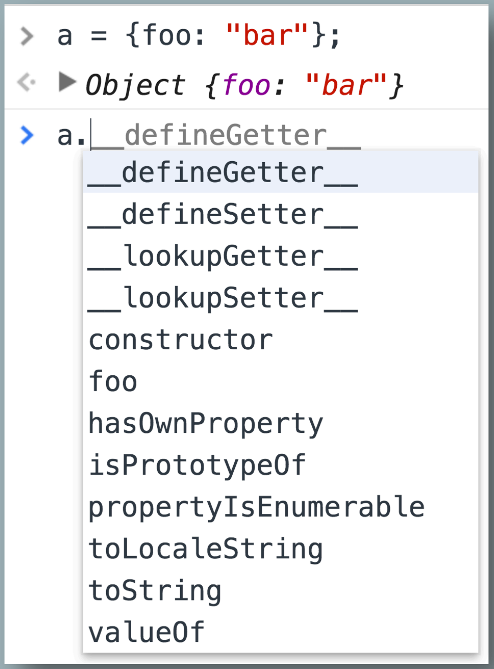
• product-image.php (single-product) — Brackets
KEEP CALM AND DRINK COFFEE

```
                    //Get the Thumbnail URL
                    $src = wp_get_attachment_image_src( get_post_thumbnail_id($post->ID), false, '' );
                    $src_small = wp_get_attachment_image_src( get_post_thumbnail_id($post->ID),  apply_filters( 'single_product_large_thumbnail_size',
'shop_single' ));
                    $src_title = get_post(get_post_thumbnail_id())->post_title;

                ?>

                <div class="slide easyzoom">
                    <img itemprop="image" src="<?php echo $src_small[0]; ?>" alt="<?php echo $src_title; ?>" title="<?php echo $src_title; ?>" />
                </div>

                <?php } else { echo '<div class="slide easyzoom"><img src="'.wc_placeholder_img_src().'" title="'.get_the_title().'" alt="'.get_the_title().'"/>
</div>';} ?>

                <?php

                    if ( $attachment_ids ) {

                        $loop = 0;
                        $columns = apply_filters( 'woocommerce_product_thumbnails_columns', 3 );

                        foreach ( $attachment_ids as $attachment_id ) {

                            $classes = array( 'zoom' );

                            if ( $loop == 0 || $loop % $columns == 0 )
                                $classes[] = 'first';

                            if ( ( $loop + 1 ) % $columns == 0 )
                                $classes[] = 'last';

                            $image_link = wp_get_attachment_url( $attachment_id );

                            if ( ! $image_link )
                                continue;

                            $image       = wp_get_attachment_image( $attachment_id, apply_filters( 'single_product_small_thumbnail_size', 'shop_thumbnail' ) );
                            $image_class = esc_attr( implode( ' ', $classes ) );
                            $image_title = esc_attr( get_the_title( $attachment_id ) );

                            printf( '<div class="slide easyzoom"><a href="%s" title="%s"><span>%s</span><div class="zoom-button large icon-expand tip-top hide-
for-small" title="Zoom"></div></a></div>', wp_get_attachment_url( $attachment_id ),get_post($attachment_id)->post_title, wp_get_attachment_image( $attachment_id,
apply_filters( 'single_product_large_thumbnail_size', 'shop_single' ) ) );

                            $loop++;
                        }


                    }
                ?>
```

THE BASICS - OBJECTS

Objects are maps/dictionaries of key-value pairs

If that is true, then what is this?

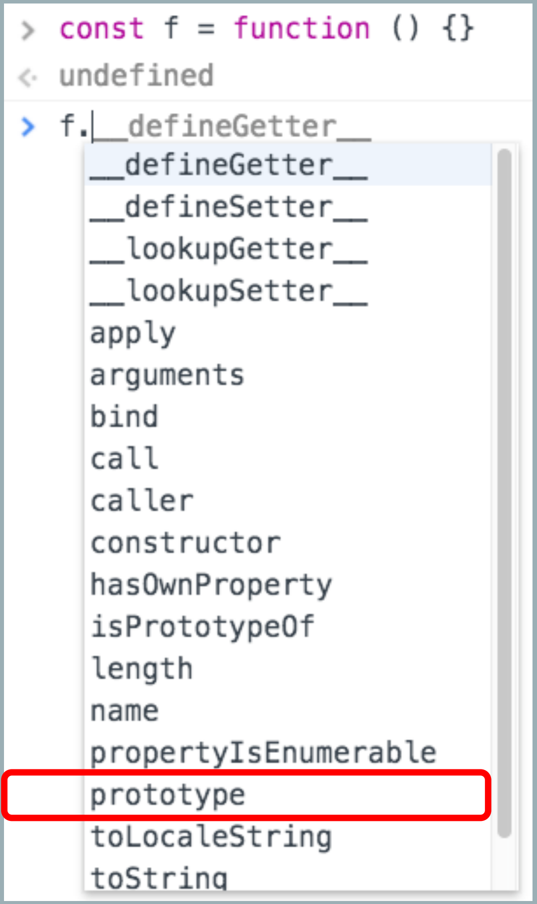


```
> a = {foo: "bar"};
<· ▶Object {foo: "bar"}
> a.__defineGetter__
    __defineGetter__
    __defineSetter__
    __lookupGetter__
    __lookupSetter__
    constructor
    foo
    hasOwnProperty
    isPrototypeOf
    propertyIsEnumerable
    toLocaleString
    toString
    valueOf
```

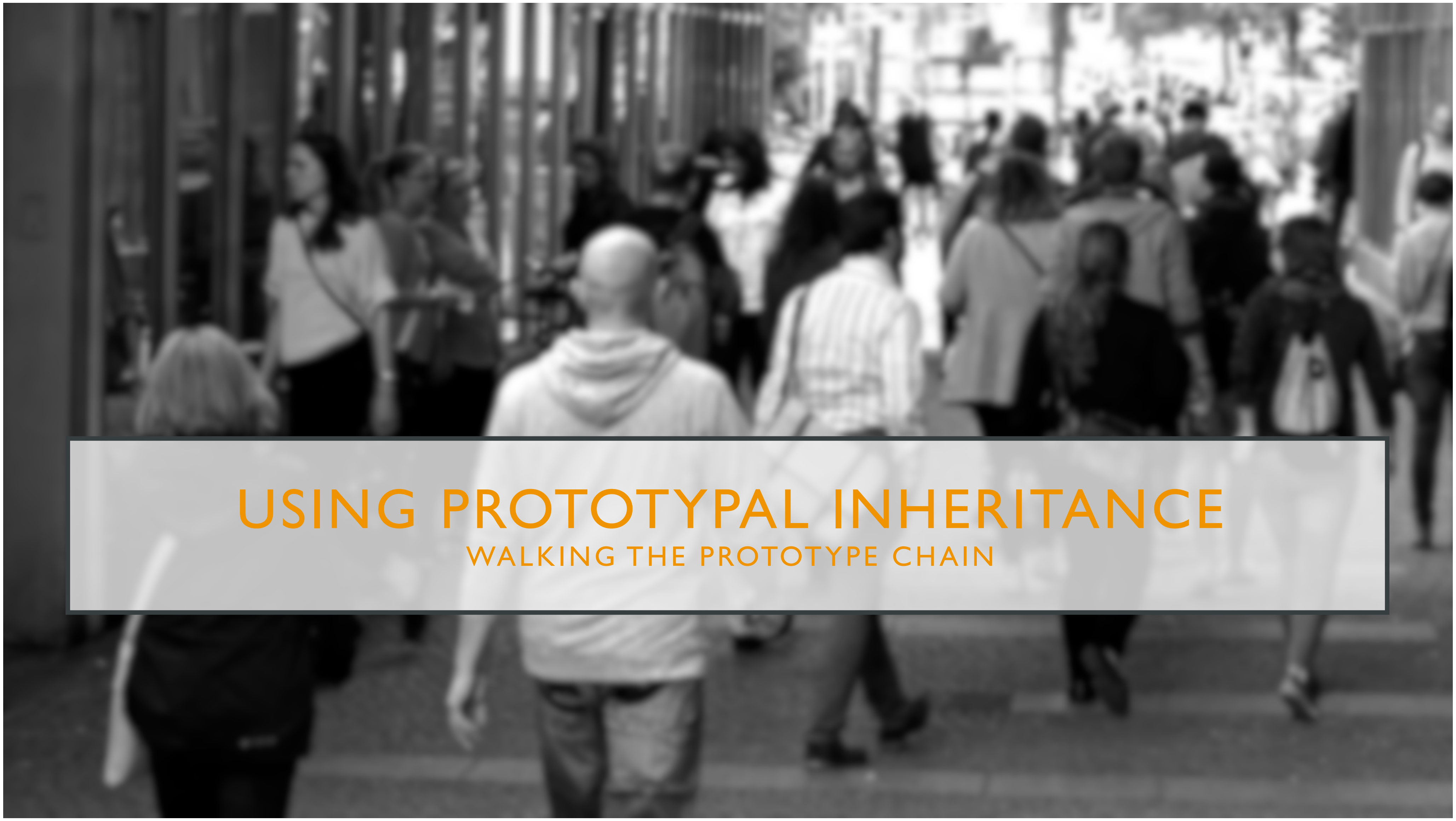
THE BASICS - FUNCTIONS

- Functions are objects

- Because they are objects, arbitrary properties can be assigned to them



```
> const f = function () {}
< undefined
> f.__defineGetter__
  __defineGetter__
  __defineSetter__
  __lookupGetter__
  __lookupSetter__
  apply
  arguments
  bind
  call
  caller
  constructor
  hasOwnProperty
  isPrototypeOf
  length
  name
  propertyIsEnumerable
  prototype
  toLocaleString
  toString
```

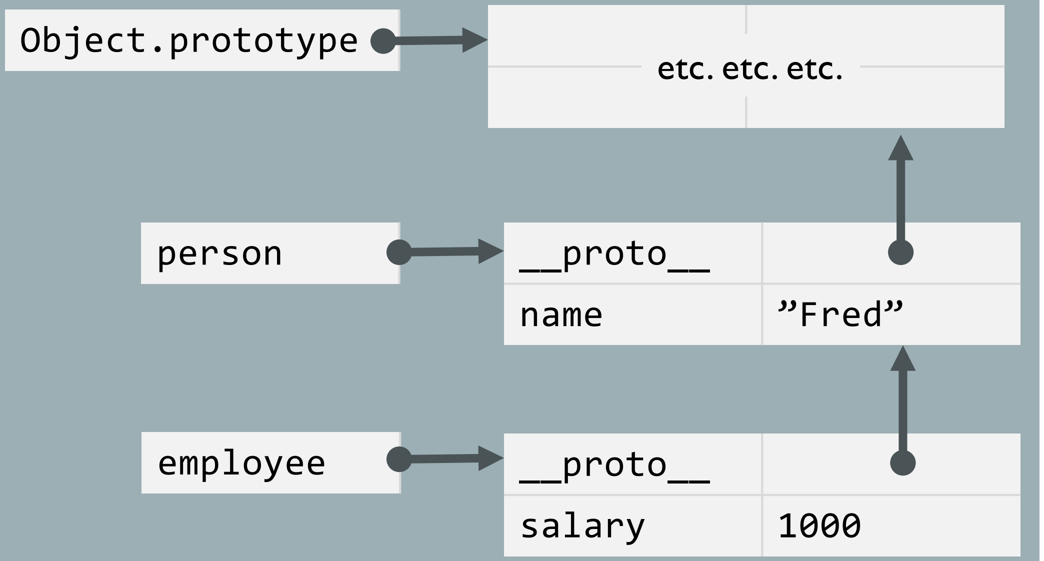


USING PROTOTYPAL INHERITANCE

WALKING THE PROTOTYPE CHAIN

THE PROTOTYPE CHAIN

```
const person = {name: "Fred"};

const employee = Object.create(person);
employee.salary = 1000;

console.log(employee.name);     // Fred
console.log(employee.salary);   // 1000
```



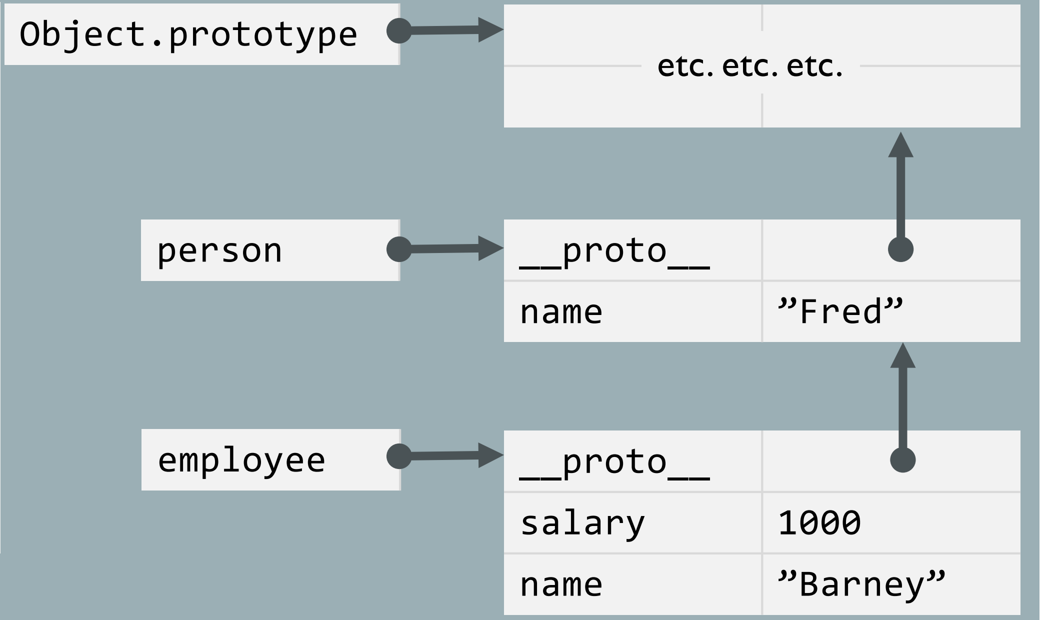
PROTOTYPE FUNCTIONS

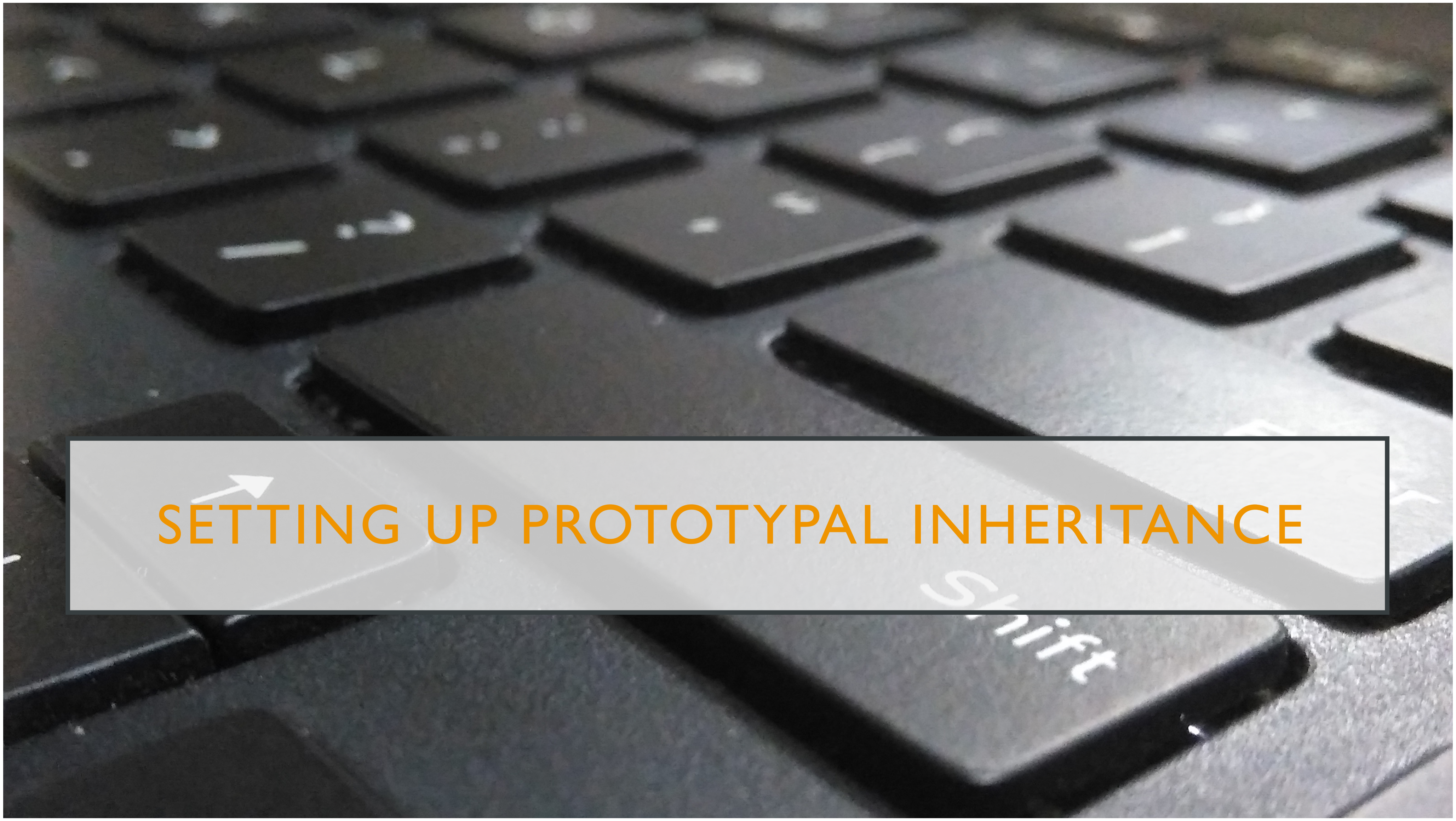
`Object.create()`	Creates a new object with a specified prototype
`Object.getPrototypeOf()`	Gets the specified object's prototype (__proto__)
`Object.setPrototypeOf()`	Sets an object's prototype (after creation). Caution: major performance hit (ES2015)
`Object.prototype.isPrototypeOf()`	Checks whether an object exists in another object's prototype ***chain***
`object` `instanceof` *`constructor`*	Tests whether constructor.prototype appears anywhere in object's prototype chain

SETTING A PROPERTY (HIDING)

Assignments do not search the prototype chain. Instead, they hide/mask properties higher up in the prototype chain.

```
const person = {name: "Fred"};

const employee = Object.create(person);
employee.salary = 1000;
employee.name = "Barney";

console.log(employee.name);     // Barney
console.log(employee.salary);   // 1000
console.log(person.name);       // Fred
```





SETTING UP PROTOTYPAL INHERITANCE

INVOKING A CONSTRUCTOR WITH NEW

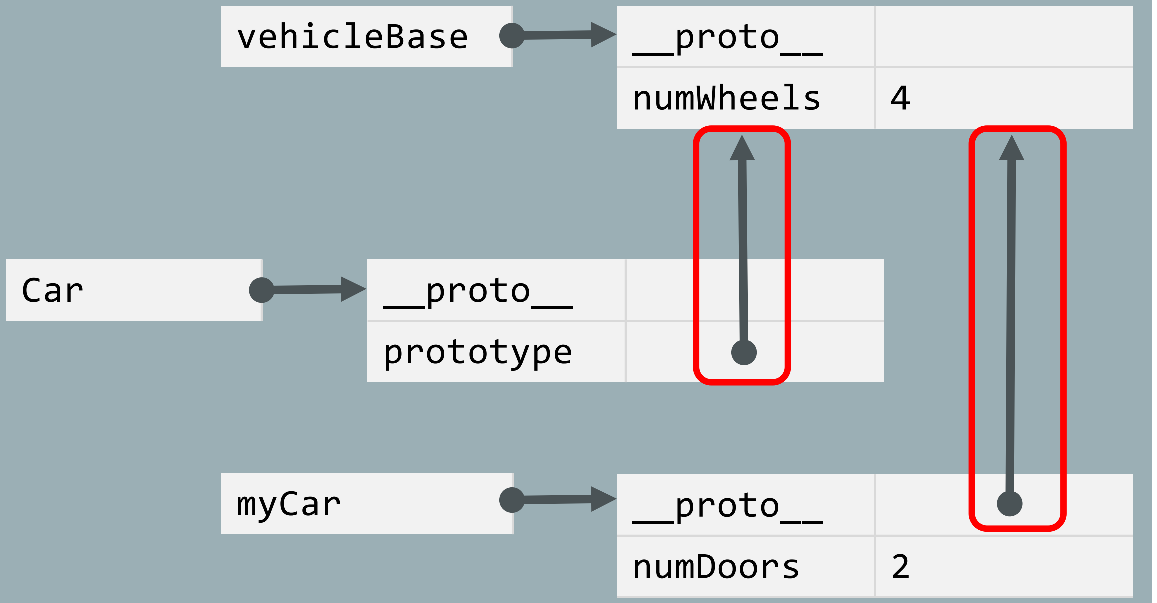
Constructor functions are intended to be invoked using the new operator (Hence the capital letter naming convention)

When a function is invoked using `new`, the following happens

1. A new object is created
2. The new object's `__proto__` is set to the constructor's `prototype` property
3. `this` is set to the new object
4. The function is invoked

CONSTRUCTOR FUNCTION

```
const vehicleBase = {
    numWheels: 4
};

function Car(numDoors) {
    this.numDoors = numDoors;
}

Car.prototype = vehicleBase;

const myCar = new Car(2);
console.log(myCar.numWheels);   // 4
console.log(myCar.numDoors);    // 2
```



ASIDE: A WORD OF WARNING

Bad things can happen if you call a constructor and forget the "new" operator
To defend against this, do one of…

1. Use strict mode
   (this will be undefined a TypeError will be thrown)

```
function Car(numDoors) {
    "use strict";
    this.numDoors = numDoors;
}
```

2. Manually protect against it

```
function Car(numDoors) {
    if (!(this instanceof Car))
        return new Car(numDoors);

    this.numDoors = numDoors;
}
```

PLANETPROTO WORKSHOPPER

Setup

```
npm install -g planetproto
```

Running

To select an exercise:

```
planetproto
```

To verify your solution:

```
planetproto verify mysolution.js
```