// setting my\_object variable to an Object

let my\_object = {

  first\_Name: 'John',

  second\_Name: 'Doe',

  // Giving method to the object

  driveCar() {

    // Accessing the property of the my\_object inside a driveCar method(function belonging to the object)

    console.log(

      my\_object.first\_Name,

      my\_object.second\_Name,

      'is driving a car'

    );

  },

};

// Accessing driveCar method from the my\_object

my\_object.driveCar();

**Using this key word:**

// setting my\_object variable to an Object

let my\_object = {

  first\_Name: 'John',

  second\_Name: 'Doe',

  // Giving method to the object

  driveCar() {

    // Accessing the property of the my\_object inside a driveCar method(function belonging to the object) by using this

    // In the following example this keyword is pointing towards

    // my\_object Object

    console.log(this.first\_Name, ' ', this.second\_Name, 'is driving a car');

  },

};

// Accessing driveCar method from the my\_object

// Calling driveCar method by (),() parenthesis is used to call the method

my\_object.driveCar();

**Creating a function inside a method can console logging this key word:**

// setting my\_object variable to an Object

let my\_object = {

  first\_Name: 'John',

  second\_Name: 'Doe',

  // Giving method to the object

  driveCar() {

    function function\_nested\_method() {

      console.log(this);

    }

    // Calling the function and () to call function

    function\_nested\_method();

    console.log(this.first\_Name, ' ', this.second\_Name, 'is driving a car');

  },

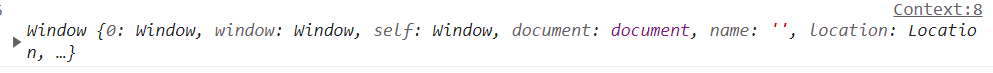
};

// Accessing driveCar method from the my\_object

// Calling driveCar method by (),() parenthesis is used to call the method

my\_object.driveCar();

**Output**



Console logging this key word gives us the Window object.

Window is the root or global object.

Window is a top-level object that contains all of the web browser’s other objects.

// setting my\_object variable to an Object

let my\_object = {

  first\_Name: 'John',

  second\_Name: 'Doe',

  // Giving method to the object

  driveCar() {

    function function\_nested\_method() {

      //  In the following code this key word is pointing towards global object(window object)

      console.log(this);

    }

    // Calling the function and () to call function

    function\_nested\_method();

    // In the following example this keyword is pointing towards

    // my\_object Object

    console.log(this.first\_Name, ' ', this.second\_Name, 'is driving a car');

  },

};

// Accessing driveCar method from the my\_object

// Calling driveCar method by (),() parenthesis is used to call the method

my\_object.driveCar();

This key word is pointing to two different things at two different times.

In following code this keyword is pointing towards my\_object

// In the following example this keyword is pointing towards

    // my\_object Object

    console.log(this.first\_Name, ' ', this.second\_Name, 'is driving a car');

Where as in the following code this keyword is pointing towards window object (global object)

   function function\_nested\_method() {

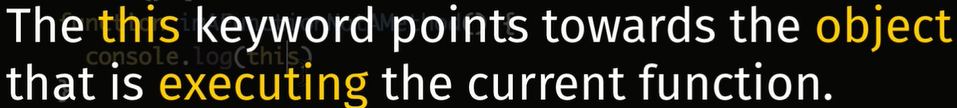
      //  In the following code this key word is pointing towards global object(window object)

      console.log(this);

    }

**How does JavaScript decide to what this key word should be pointing towards?**

**This key word does points towards the object that is executing or calling the current function.**



//my\_object is calling or executing the driveCar function

// Hence this key word points towards the my\_object

my\_object.driveCar();

// In the following example this keyword is pointing towards

    // as diveCar  is being called by my\_object

    console.log(this.first\_Name, ' ', this.second\_Name, 'is driving a car');

Text

Description automatically generated

my\_object is executing driveCar method. Hence this key word will point towards my\_object inside the driveCar method.

 // Calling the function and () to call function

    // When we are calling a regular old function,JavaScript assumes that regular function is being called

    // by the global object(window object), hence this key word inside regular function function\_nested\_method will point towards global object

    function\_nested\_method();

function function\_nested\_method() {

      //  In the following code this key word is pointing towards global object(window object)

      console.log(this);

    }



**Standalone breath function:**

// breath function is a stand alone function and will be called by global object(window object)

// In that global object (window object) there is not first\_Name and there is no second\_Name propertie

// inside the global object

// It is not a method that belongs to an object

function breath() {

  console.log(

    this.first\_Name,

    '',

    this.second\_Name,

    'just inhaled and exhaled'

  );

}

In JavaScript function is an Object, functions have access to a method called call. Inside () parenthesis of the call method we insert whatever we want this keyword point towards.

If we want this keywords point towards specific object then we place that specific object inside () of the call method.

// setting my\_object variable to an Object

let my\_object = {

  first\_Name: 'John',

  second\_Name: 'Doe',

};

// breath function is a stand alone function and will be called by global object(window object)

// In that global object (window object) there is not first\_Name and there is no second\_Name propertie

// inside the global object

// It is not a method that belongs to an object

function breath() {

  console.log(

    this.first\_Name,

    '',

    this.second\_Name,

    'just inhaled and exhaled'

  );

}

// Name of the function.call()

// call method is going to call and execute breath function and allows us to control this key word

// Whatever we place inside () of the call method , call will make this keyword point towards  that

// whatever.

// So if we want "this" keyword point towards my\_object inside breath function , then we pass my\_object

// inside () of the call method

breath.call(my\_object);