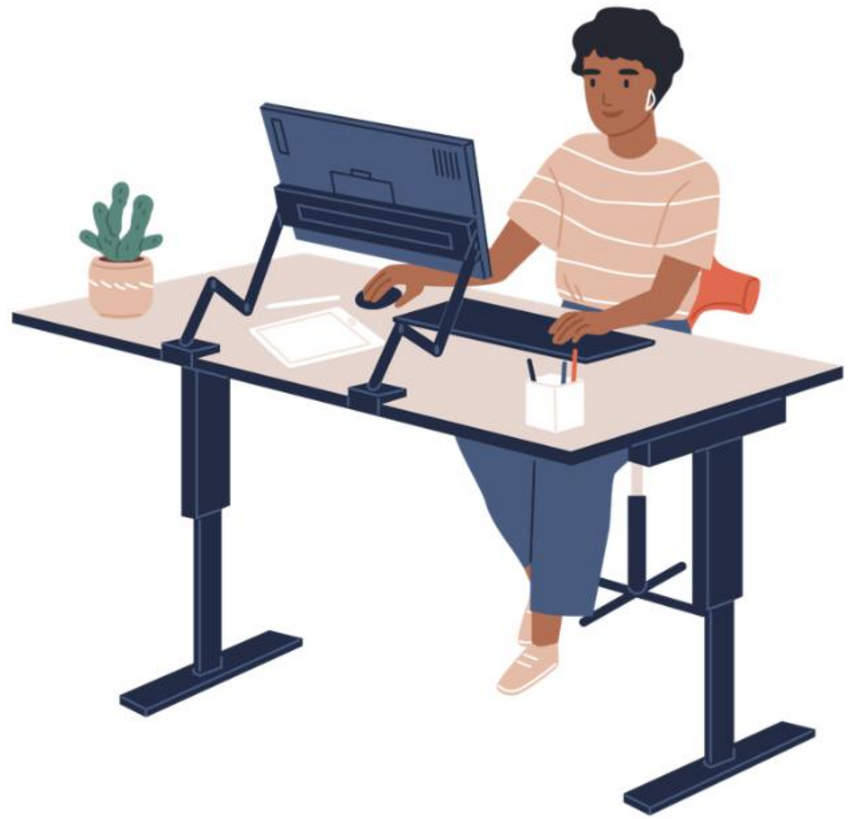
Slide Note

Learning Consolidation

Utilize Arrays to Model Aggregate Data

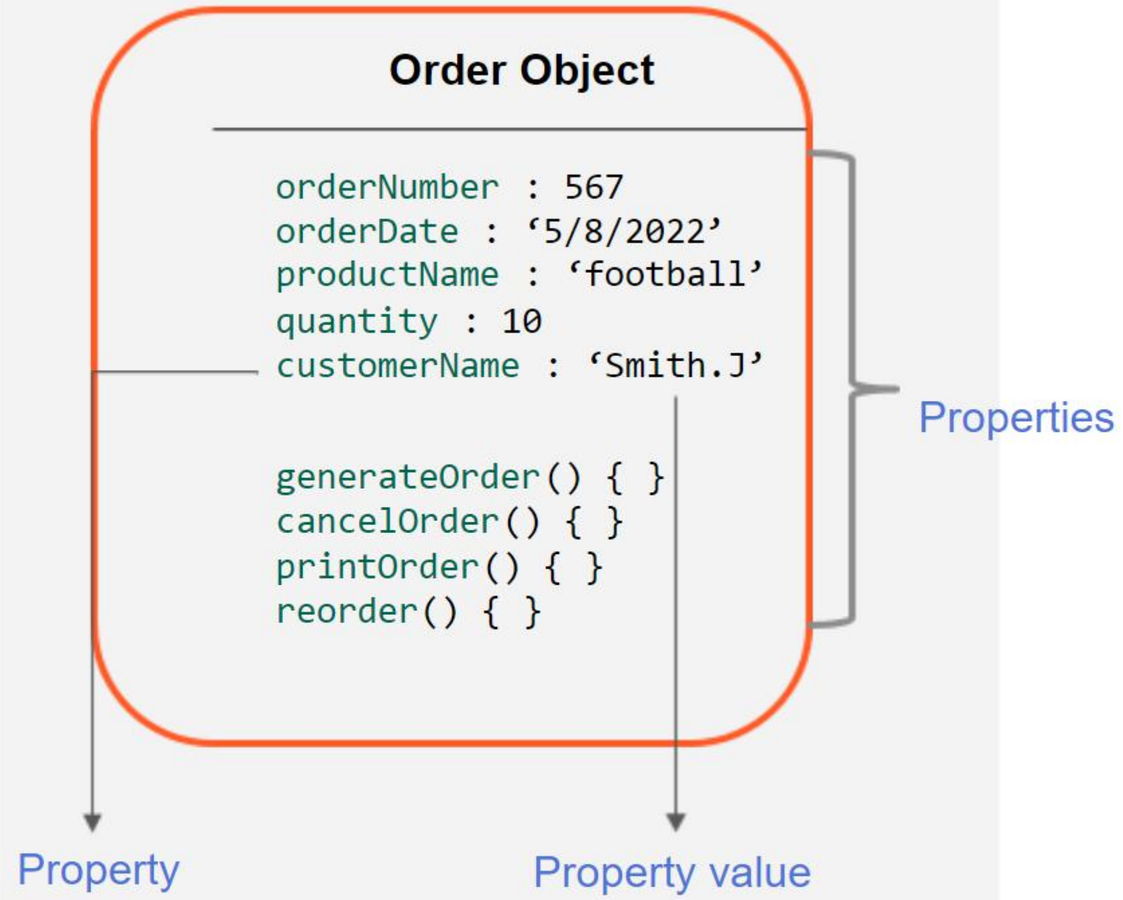


Slide Note

In this sprint, you have learned to:

- Create objects to model data
- Create arrays to model aggregate data
- Explain array structure
- Apply `for` and `for...of` loops to access array elements
- Pass an array as a function parameter

Slide Note



Object

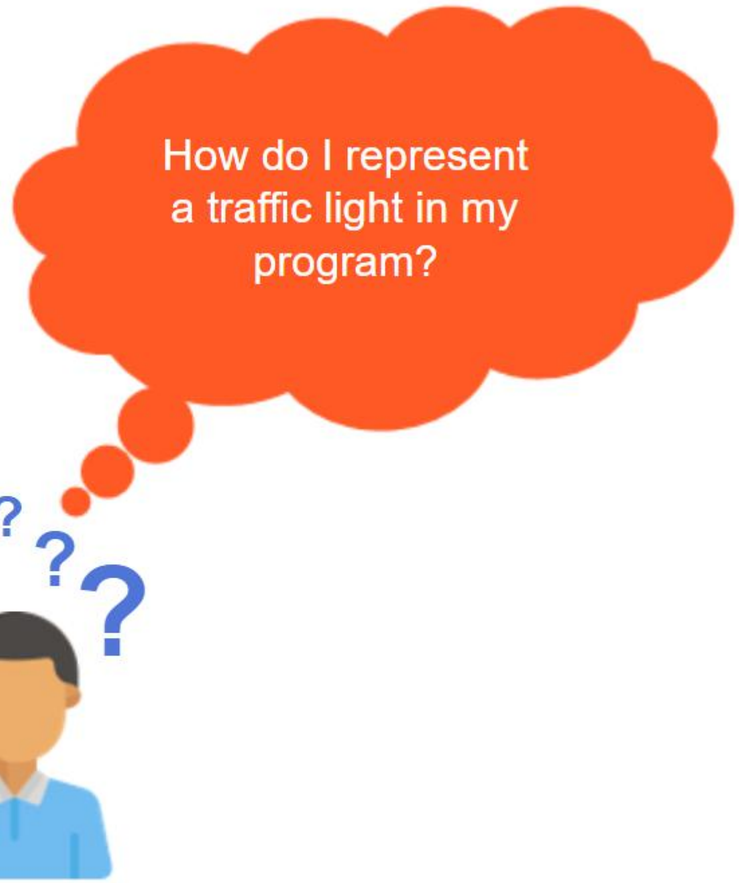
- An object in JavaScript is a collection of properties.
- A property is an association between key and value.
  - Key is the name of the property
  - Value is the data assigned to the key
- The property could be a:
  - Variable that stores attributes' data
  - Function that represents behavior
- Function inside an object is termed **Method.**

Slide Note

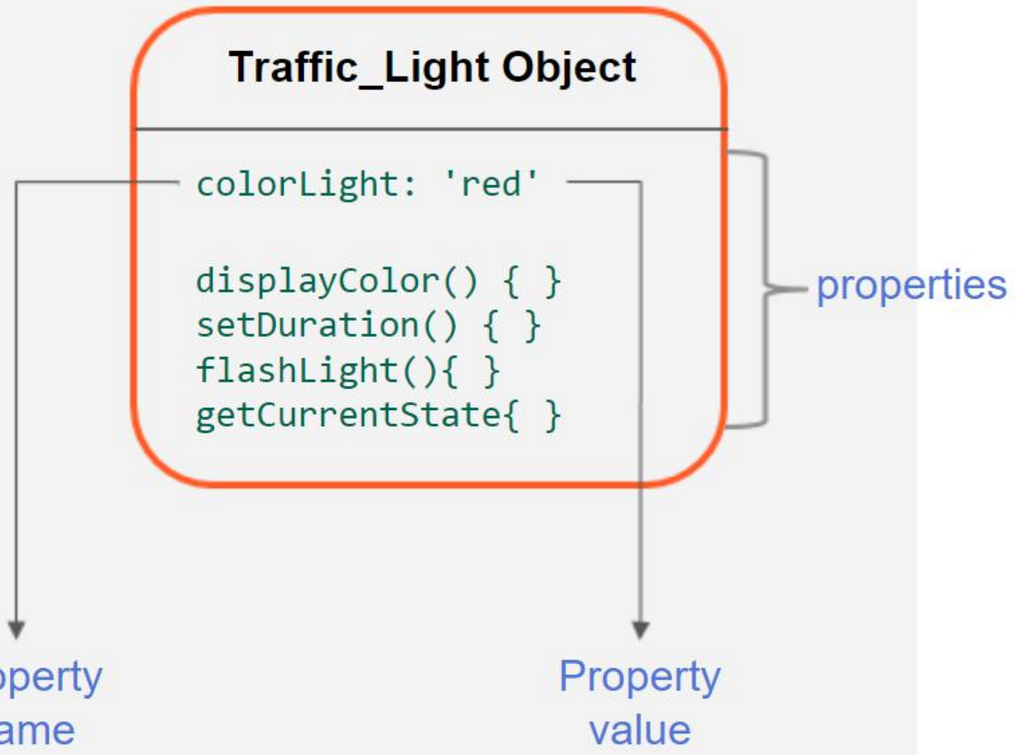
Online order is a real-time entity

Traffic Light

- A traffic light has the following property:
  - The state of the traffic light color (red, orange, or green).
- The behaviors of traffic lights are:
  - Displaying colors one at a time.
  - Flashing an orange light in a lull period.
  - Displaying a color for a certain duration.



Slide Note



Representing a Traffic Light As an Object

- The traffic light object contains a set of attributes and behavioral characteristics.
- A traffic light comprises the below properties:
  - colorLight
  - displayColor()
  - setDuration()
  - flashLight()
  - getCurrentState()
- Such a data structure which has details of different types qualifies to be represented as an **object**.

Slide Note

The other alternative way is to create object using constructor

Constructor is a function that specifies name, properties and methods of object.

Inside function, use keyword *this* to assign values to the object's properties based on the values passed to the function.

Create an instance of the object with new.

In below code:

Employee is name of object type

firstName, lastName, email are its properties

```
function Employee(firstName, lastName, email) {
    this.firstName = firstName;
    this.lastName = lastName;
    this.email = email;
}
const employee = new Employee('Nancy','Davilio','nancy.d@gma
```

Create Objects in JavaScript

- Objects in JavaScript can be created using an initializer.
- An initializer uses object literal notation, i.e. { }.
- Object initializers are expressions.
- Each object initializer results in a new object being created.
- In the code shown:
  - `Student` is an object.
  - `firstName`, `lastName`, `email`, and `fullname()` are properties of a student.

Note: fullName() is a method which combines firstName and lastName.

Property value

Property name

```
const student = {
    firstName: "Tina",
    lastName: "Keith",
    email: "tina.k@gmail.com",
    fullName() {
      return firstName + "." +
lastName
    }
}
```

Slide Note

Access Object Properties

- In an object, the dot operator (.) is used to associate the property with its object:
  - For example, `employee.firstName` provides access to the `firstName` value of the `employee` object.
  - For example, `employee.email` provides access to the `email` value of the employee object.

```
const employee = {
    firstName: "Tina",
    lastName: "Keith",
    email: "tina.k@gmail.com",
}
console.log(`Employee Details`);
console.log(`${employee.firstName}`);
console.log(`${employee.email}`);
}
```

Slide Note

Array and Its Characteristics

- An array is a complex type that enables storing a collection of multiple elements under a single variable name.
- JavaScript arrays are not primitive but are instead array objects.
- JavaScript arrays are resizable and can contain a mix of different data types.
- An index is the position at which the element is stored in the array.
- Array elements are zero-indexed:
  - The first element of an array is at index 0.
  - The second is at index 1, and so on.
  - The last element is at the value of the array's size minus 1.
- In the example below, stu_name is the array name and the values inside the square brackets are the elements of the array.

```
let stu_name = ["Britto", "Charles", "Thomas"];
```

Slide Note

Create an Array

- Arrays can be created in multiple ways.
- The easiest way to create an array is using an array literal.

```
let stu_name = ["Britto", "Charles", "Thomas"];
```

stu_name: Variable name for array

[ ]: Square brackets are the notation for array declaration.

- You can also create an array and then provide elements by initializing with values.

```
let stu_name = [];
stu_name[0] = "Britto";
stu_name[1] = "Charles";
stu_name[2]= "Thomas";
```

Slide Note

Accessing Arrays

- Arrays can be accessed in various ways:
  - Using index
  - Using a simple `for` loop
  - Using a `for…of` loop
- JavaScript provides a built-in property called length, which is used to get the number of elements in an array. This is used while iterating through the array until the last element of the array in a simple `for loop`.
- The loop traverses the array until the last element while using `for…of` loops. There is no need to specify the length of the array explicitly. There is no use of an index or rather a counter inside the `for…of loop`.

Slide Note

Arrays as Parameters and Return Types

- An array can be passed to a method as a parameter just as normal variables can.
- An array can be a return type from a method.
- There is no need to specify the [ ] along with the array name while passing the array as a parameters and while returning the value from a function.

Slide Note

```
const colors = ["Red", "Blue",
"Green "];
const [red, blue, green]
= colors; console.log(red); //Red
console.log(blue); //Blue
console.log(green); //Green
```

Destructuring an Array

- **Destructuring** in JavaScript is a simplified method of extracting multiple properties from an array.
- It takes the structure and deconstructs it down into its own constituent parts through assignments by using a syntax that looks like array literals.
- Without de-structuring, if we want to extract data from an array, we need to use a separate variable to assign every individual element of an array and repeat the process again and again.

Slide Note

Array Spread Operator

- The JavaScript spread operator is denoted by three dots (...).
- It allows us to quickly copy all or part of an existing array or object into another array or object.
- The spread operator is often used in combination with the array de-structuring.

```
const colors1 = ["Violet", "Indigo", "Blue
"];
const colors2
=  ["Green", "Yellow", "Orange", "Red"];
let colors = [...colors1, ...colors2];
//Prints all the rainbow colors
console.log(colors);
//Assign the first and second items
from num array to variables and put the
rest in an array
const num = [1,2,3,4,5,6];
const [one, two,...rest] = num;
```

Slide Note

Self Check

How do you access the accountBalance property of the account object?

```
 let account = { accountNumber: 567001, accountType: 'Savings',
accountBalance: 7000};
```

1. account.accountBalance
2. account.accountBalance()
3. account$accountBalance
4. account$accountBalance()

Slide Note

Self Check: Solution

How do you access the accountBalance property of the account object?

```
 let account = { accountNumber: 567001, accountType: 'Savings',
accountBalance: 7000};
```

1. **account.accountBalance**
2. account.accountBalance()
3. account$accountBalance
4. account$accountBalance()

Slide Note

Self Check

Which of the following statements is/are false about an array?

1. An array can contain duplicate elements.
2. An array uses a zero index to reference the first element.
3. The size of the array is determined using its length property.
4. An array cannot expand automatically when its size is full.

Slide Note

Self Check: Solution

Which of the following statements is/are false about an array?

1. An array can contain duplicate elements.
2. An array uses a zero index to reference the first element.
3. The size of the array is determined using its length property.
4. **An array cannot expand automatically when its size is full.**