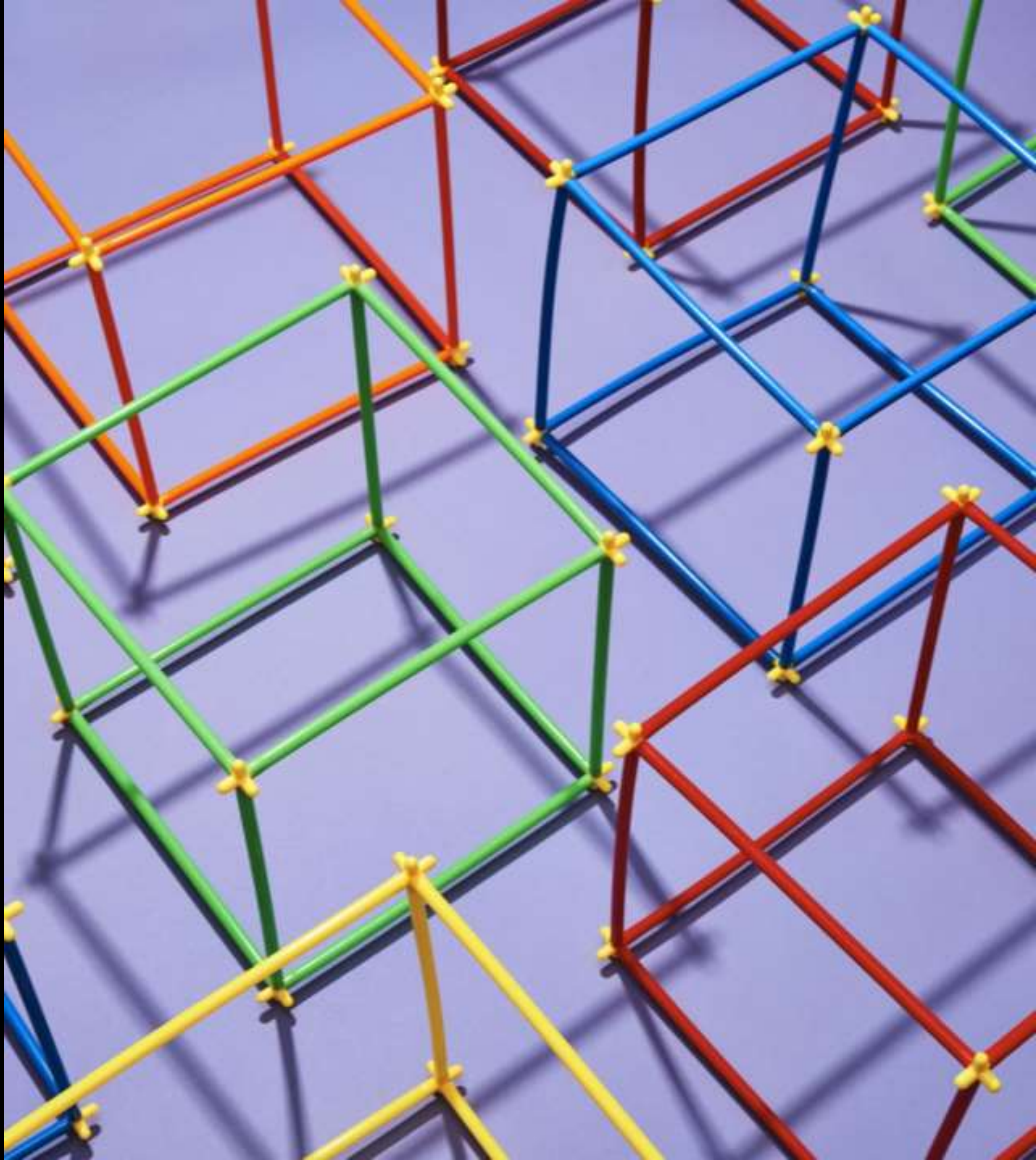
OBJECTS AND THEIR INTERNAL REPRESENTATION IN JAVASCRIPT



What Is an Object?

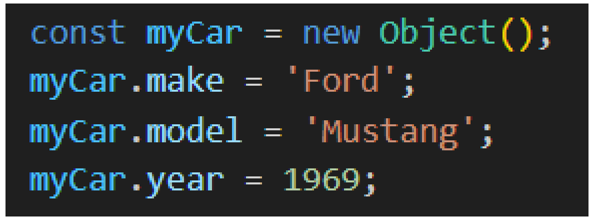
An object in JavaScript can be defined as an unordered collection of related data, represented as “key: value” pairs. These keys are known as properties, and they can be variables or functions. Objects allow us to organize and manage data efficiently.

Properties and Methods

1. Properties:

- A JavaScript object has properties associated with it.
- Think of properties as variables attached to the object.
- Properties define the characteristics of the object.
- You can access object properties using dot notation: objectName.propertyName.

For example:



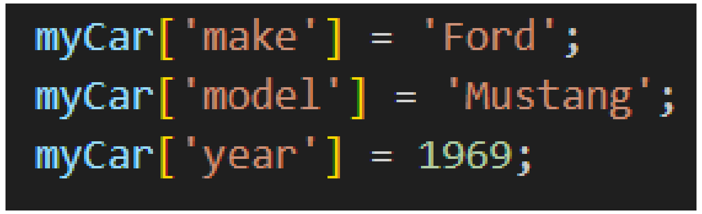
```
const myCar = new Object();
myCar.make = 'Ford';
myCar.model = 'Mustang';
myCar.year = 1969;
```

Here, myCar has properties named make, model, and year.

2. Accessing Properties:

❖ Properties can also be accessed using bracket notation.

For instance:



```
myCar['make'] = 'Ford';
myCar['model'] = 'Mustang';
myCar['year'] = 1969;
```

This notation is useful when property names are dynamically determined or contain special characters.

3. Internal Representation:

❖ Internally, JavaScript engines represent objects using various data structures.
❖ One common representation is the hash table, where keys are hashed to optimize property access.
❖ The hash table allows for quick lookup of properties, making object access efficient.

Conclusion:

JavaScript objects serve as powerful building blocks for creating dynamic and flexible applications. Understanding their internal representation helps developers make informed design choices and optimize performance. So, next time you work with objects, remember that behind the scenes, they’re collections of key-value pairs waiting to bring your code to life!