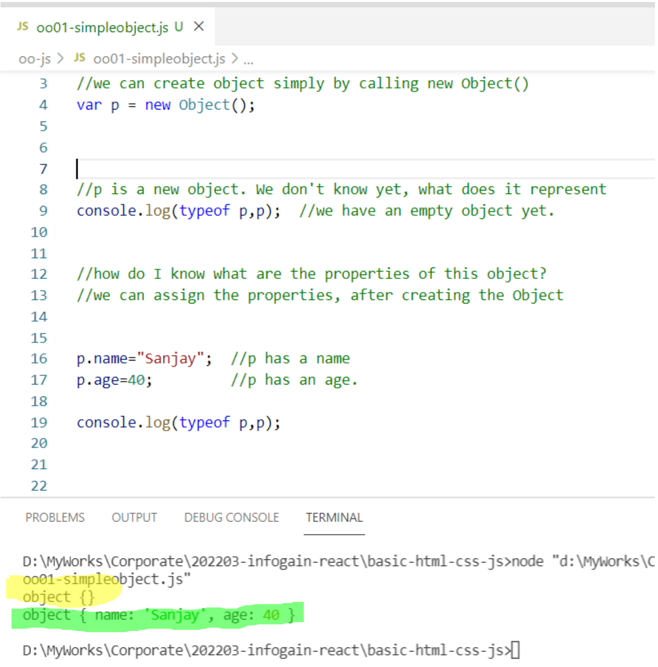
Object Oriented Javascript

Thursday, March 10, 2022 4:40 PM

- Java supports Object Oriented Programming
- We can create Objects without needing to create a class first.



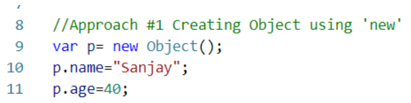
```
//we can create object simply by calling new Object()
var p = new Object();



//p is a new object. We don't know yet, what does it represent
console.log(typeof p,p);  //we have an empty object yet.


//how do I know what are the properties of this object?
//we can assign the properties, after creating the Object


p.name="Sanjay";  //p has a name
p.age=40;         //p has an age.

console.log(typeof p,p);
```

```
D:\MyWorks\Corporate\202203-infogain-react\basic-html-css-js>node "d:\MyWorks\C
oo01-simpleobject.js"
object {}
object { name: 'Sanjay', age: 40 }

D:\MyWorks\Corporate\202203-infogain-react\basic-html-css-js>
```

- In Javascript we can simply create an object by calling
  - new Object()
- An object when created, has no property or method
  - We may not even know the purpose of this object yet
  - The object is created empty
- We can attach properties to this object later dynamically.
  - Object gets dynamic properties.

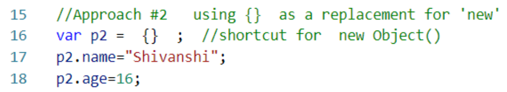
Different ways to create Object and Assign Properties

Approach #1 — Using new Object and dot notation



```
//Approach #1 Creating Object using 'new'
var p= new Object();
p.name="Sanjay";
p.age=40;
```

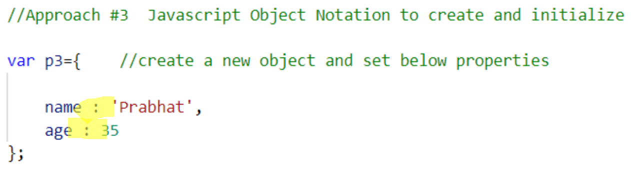
Approach #2 — Using {} as shortcut for new Object



```
//Approach #2   using {}  as a replacement for 'new'
var p2 =  {}  ;  //shortcut for  new Object()
p2.name="Shivanshi";
p2.age=16;
```

Approach #3 — using Javascript Object Notation (JSON)

- We can create and initialize values together

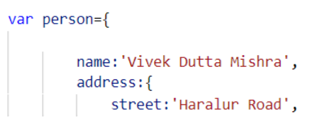


```
//Approach #3  Javascript Object Notation to create and initialize

var p3={    //create a new object and set below properties

    name : 'Prabhat',
    age : 35
};
```

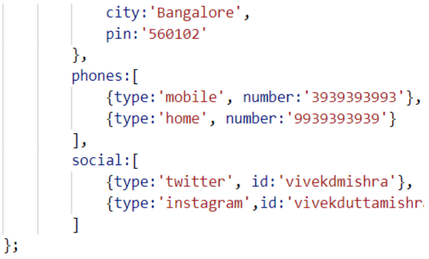
- Note:
  - Property and values are separated by ":" and not '='

JSON is a extremely simple idea with tl points to remember

1. {} —> new Object
2. prop: value —> set value for the property
3. [] —> a new Array

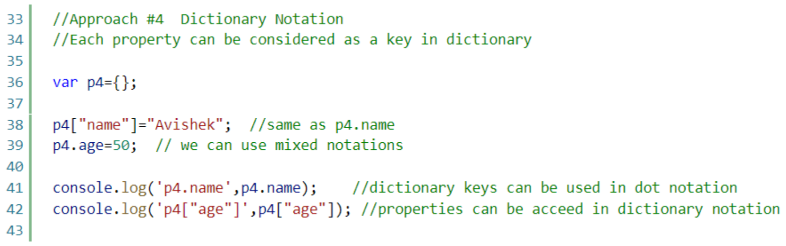


```
var person={

        name:'Vivek Dutta Mishra',
        address:{
            street:'Haralur Road',
```



```
            city:'Bangalore',
            pin:'560102'
        },
        phones:[
            {type:'mobile', number:'3939393993'},
            {type:'home', number:'9939393939'}
        ],
        social:[
            {type:'twitter', id:'vivekdmishra'},
            {type:'instagram',id:'vivekduttamishr
        ]
};
```

Approach #4 Dictionary Notation



```
//Approach #4  Dictionary Notation
//Each property can be considered as a key in dictionary

var p4={};

p4["name"]="Avishek";  //same as p4.name
p4.age=50;  // we can use mixed notations

console.log('p4.name',p4.name);    //dictionary keys can be used in dot notation
console.log('p4["age"]',p4["age"]); //properties can be acceed in dictionary notation

```

- Each Javascript Object is essentially like a dictionary of Key value pair
- Each property can be written in dictionary notation
- The dot notation (.) and dictionary notation are interchangeable
- Both the below idea will be same
  - `x.name ——-> x["name"]`

Mixed Mode

- The approaches we discussed above are not mutually exclusive
- They can be used in combination each other

```
var person = {

    name: 'Sanjay' //JSON Notation
};

person.age = 45;   //dot Notation

person[ "phone" ] = '93939339393';  //Dictionary Notation
```

- When accessing the value you can access any value using either dot notation or dictionary notaion
  - There is no JSON notation to access the value

```
console.log(person [ "name " ] );

console.log( person["age" ]);

console.log( person.phone);
```