Objects and JSON

SoftUni Team

Technical Trainers

Software University

http://softuni.bg

Table of Contents

Have a Question?

SoftUni Foundation

sli.do

#tech-fund

Objects

Definition, Properties and Methods

What are Objects ?

- Collection of related data or functionality
- Consists of several variables and functions called **properties** and **methods**
- In JavaScript, at run time you can add and remove properties of any object

Object name

Property name

```
let obj = { name:'Peter', age: 20 }
console.log(obj.name) //Peter
```

Property value

Object Definition

- We can create an object with an **object literal**, using the following syntax:

```
let person = {name:'Peter', age: 20, hairColor: 'black'}
```

- We can define empty object and add the properties later

```
let person = {}
 person.name ='Peter'
 person["lastName"] = 'Parker'
 person.age = 20
 person.hairColor = 'black'
```

You can access and set properties using both ways

Object Methods

SoftUni Foundation

- Functions within a JavaScript object are called **methods**
- We can define methods using several syntaxes:

```
let person = {
  sayHello : function() {
     console.log('Hi, guys')
  }
}
```

```
let person = {
  sayHello() {
     console.log('Hi, guys')
  }
}
```

- We can add a method to an already defined object

```
let person = { name:'Peter', age: 20 }
person.sayHello = () => console.log('Hi, guys')
```

The Object Methods

- Methods:
  - Object.**entries()** – returns array of all properties and their values of an object
  - Object.**keys()** – returns array with all the properties
  - Object.**values()** – returns array with all the values of the properties

```
Object.entries(cat) //[['name', 'Tom'], ['age', 5]]
```

```
Object.keys(cat) //['name', 'age']
```

```
Object.values(cat) //['Tom', 5]
```

Iterate through Keys

SoftUni Foundation

- Use **for-in** to loop through the keys:

in loops through the properties

```
let obj = { name:'Peter', age:'18', grade:'5.50' }
for (let prop in obj) {
    console.log(`${prop}: ${obj[prop]}`);
}
```

Returns the value of the property

Problem: Person Info

- Create a person object that has first name, last name and age. Print the entries of a given object.

Input	Output
Peter Pan 20	firstName: Peter lastName: Pan age: 20

Input	Output
Jack Sparrow unknown	firstName: Jack lastName: Sparrow age: unknown

Solution: Person Info

- Create a Person object
- Set properties first name, last name and age
- Get the entries. Loop through them and print them

```
//TODO: create the object and set the properties
let entries = Object.entries(person);
for (let [ key, value ] of entries) {
    console.log(`${key}: ${value}`);
}
```

Check your solution here: https://judge.softuni.bg/Contests/1323

Problem: City

- Create a City object which will hold area, population, country and postcode. Loop through all the keys and print them with their values

Input	Output
Sofia 492 1238438 Bulgaria 1000	name -> Sofia area -> 492 population -> 1238438 country -> Bulgaria postCode -> 1000

Solution: City

- Create a City object
- Set the properties
- Loop through the entries and print them

```
//TODO: create the city object and set the properties
for (let prop in city) {
    //TODO: print in appropriate format
}
```

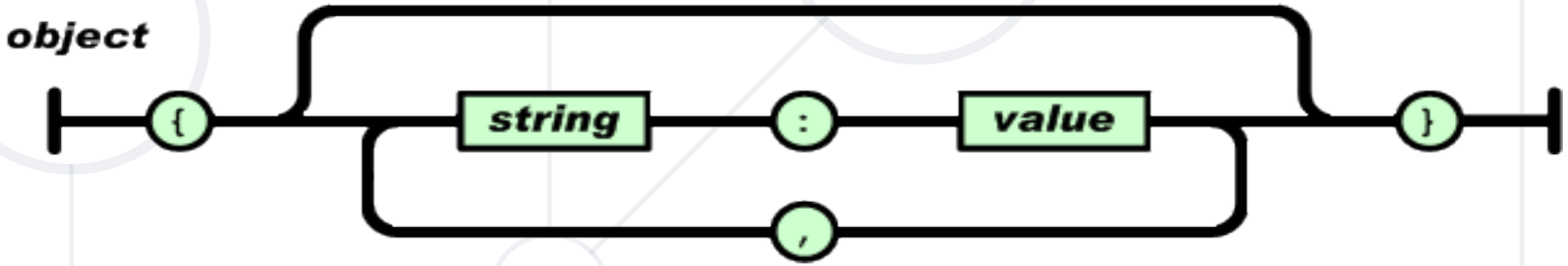
Check your solution here: https://judge.softuni.bg/Contests/1323

JSON

JavaScript Object Notation

What is JSON

- **JSON** stands for **J**ava**S**cript **O**bject **N**otation
- **Open-standard** file format that uses text to transmit data objects
- JSON is **language independent**
- JSON is "**self-describing**" and easy to understand



JSON Usage

- Exchange data between **browser** and **server**
- JSON is a **lightweight** format compared to XML
- JavaScript has built in functions to **parse JSON** so it's easy to use
- JSON uses **human-readable** text to transmit data

JSON Example

SoftUni Foundation

Brackets define a JSON

Keys and values separated by :

Keys are in double quotes

It is possible to have object in object

In JSON we can have arrays

```
{
  "name": "Peter",
  "age": 25,
  "grades": {
    "Math": [2.50, 3.50],
    "Chemestry": [4.50]
  }
}
```

JSON Methods

- We can convert JavaScript object into JSON string using JSON.**stringify**(object) method

```
let text = JSON.stringify(obj)
```

- We can convert JSON string into JavaScript object using JSON.**parse**(text) method

```
let obj = JSON.parse(text)
```

Problem: Convert to Object

- Write a function that receives a string in **JSON** format and converts it to object
- Print the entries of the object

Input
`'{"name": "George", "age": 40, "town": "Sofia"}'`
Output

`name: George`
`age: 40`
`town: Sofia`

Tips: Convert to Object

- Use JSON.**parse**( ) method to parse JSON string to an object
- Use Object.entries( ) method to get object's properties names and values
- Loop through the entries and print them

```
function solve(json){
    //TODO: use the tips to write the function
}
```

Solution: Convert to Object

```
function solve(json) {
    let person= JSON.parse(json);

    let entries = Object.entries(person);

    for (let [key, value] of entries) {
        console.log(`${key}: ${value}`);
    }

}
```

Check your solution here: https://judge.softuni.bg/Contests/1323

Problem: Convert to JSON

- Write a function that receives first name, last name, hair color and sets them to an object.
- Convert the object to **JSON string** and print it.

Input
`'George', 'Jones', 'Brown'`
Output
`{"firstName": "George", "lastName": "Jones", "hairColor": "Brown"}`

Tips: Convert to JSON

- Create an object with the given input
- Use JSON.**stringify**( ) method to parse object to JSON string
- Keep in mind that the property name in the JSON string will be **exactly the same** as the property name in the object

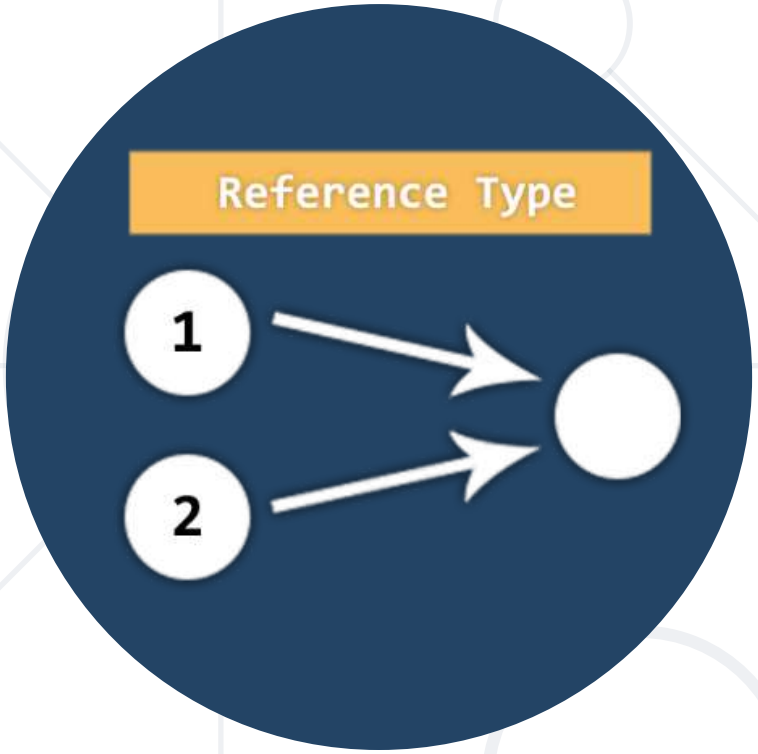
```
function solve(name, lastName, hairColor){
    //TODO: use the tips and write the code
}
```

Solution: Convert to JSON

```
function solve(firstName, lastName, hairColor){
    let person = {
        firstName,
        lastName,
        hairColor
    }

    console.log(JSON.stringify(person));
}
```

Check your solution here: https://judge.softuni.bg/Contests/1323



Value vs. Reference Types

Memory Stack and Heap

Value vs. Reference Types

pass by reference

cup =

fillCup( )

pass by value

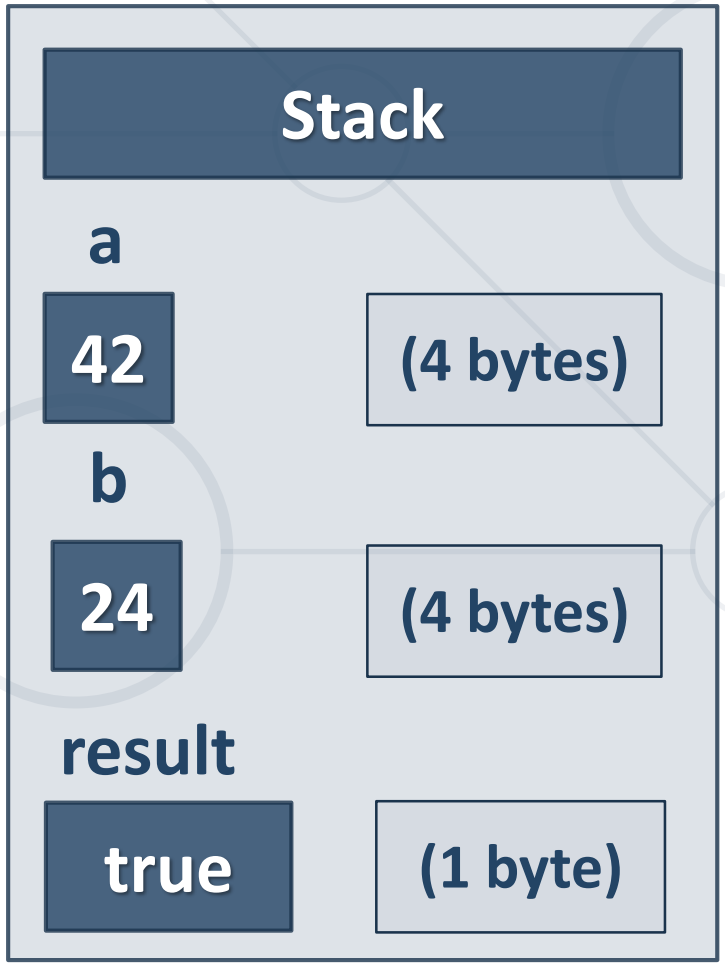
cup =

fillCup( )

Value Types

- **Value type** variables hold directly their value
- Each variable has its own **copy** of the **value**

```
let a = 42;
let b = 24;
let result = true;
```



Value Types Example

```
let a = 4 // a = 4
let b = a // b = 4
console.log(a + b)// 8
b = doubleNumber(b)// b = 10
console.log(a + b) // 14
```

Each variable has its own value

- When we pass variables as parameters we are passing copy of their values

You must return the new value

```
let doubleNumber = (num) => return num *= 2
```

What is Reference Type ?

- **Object** is not the only reference type in JavaScript
  - Arrays are also regular objects
- Variables assigned with non-primitive values are given a **reference** to the memory address with that value
- Many variables can **reference** the **same** object
  - Operations on any variable modify the same data

Reference Type Example

```
function test() {
    let person = { name: 'Peter' };
    let obj = person;
    console.log(person.name);
    changeName('object', obj);
    console.log(person.name);
}

function changeName(name, obj){
    obj.name = name;
}
```

Reference is passed to the function

The function is accessing the object directly, there is no need of return.

Reference vs Value Example: Value Types

```
function solve(){
  let num = 5;
  increment(num, 15);
  console.log(num);
}

function increment(num, value){
  num += value;
}
```

num = 5

Reference vs Value Example: Reference Types

```
function solve(){
  let nums = [ 5 ];
  increment(nums, 15);
  console.log(nums[0]);
}

function increment(nums, value){
  nums[0] += value;
}
```

num = 20

Class

Classes

Object Models

What are classes

- Extensible program-code-template for creating **objects**
- Provides **initial values** for the state of an object
- An object created by the class pattern is called an **instance** of that class
- A class has a **constructor** - subroutine called to create an object. It prepares the new object for use

Class Declaration

SoftUni Foundation

- Declaring a class

To declare a class we use the **class** keyword with the name of the class.

```
class Student {
  constructor(name) {
    this.name = name
  }
}
```

The **constructor** is a special method for creating and initializing an object

Class Example

SoftUni Foundation

- Creating a class:

```
class Student {
  constructor(name, grade) {
    this.name = name
    this.grade = grade
  }
}
```

this keyword is used to set a property of the objects to a given value

- Creating an instance of the class:

```
let student = new Student('Peter', 5.50)
```

Functions in a Class

SoftUni Foundation

- Classes can also have functions as property, called methods:

```
class Dog {
  constructor() {
    this.speak = () => {
      console.log('Woof')
    }
  }
}

let dog = new Dog()
dog.speak() //Woof
```

We access the method as a regular property

Problem: Cat

- Write a function that receives **array of strings** in the following format: '{cat name} {age}'
- Create a Cat class that receives the **name** and the **age** parsed from the input
- It should also have a function named "**meow**" that will print "{cat name}, age {age} says Meow" on the console
- For each of the strings provided you must create a cat object.

Input	Output
['Mellow 2','Tom 5']	Mellow, age 2 says Meow Tom, age 5 says Meow

Tips: Cat

- Create a Cat class
- Set properties name and age
- Set property 'meow' to be a function that prints the result
- Parse the input data
- Create all objects using class constructor and the parsed input data and store them in an array
- Loop through the array using for…of cycle and invoke .meow() method

Solution: Cat

```
function solve(arr){
    //TODO: create the Cat class
     let cats = [];
     for(let i = 0; i < arr.length; i++){
        let catData = arr[i].split(' ');
        let [name, age] = [catData[0], catData[1]];
         cats.push(new Cat(name, age));
      }
     //TODO iterate through cats[] and invoke .meow()
            using for…of loop
}
```

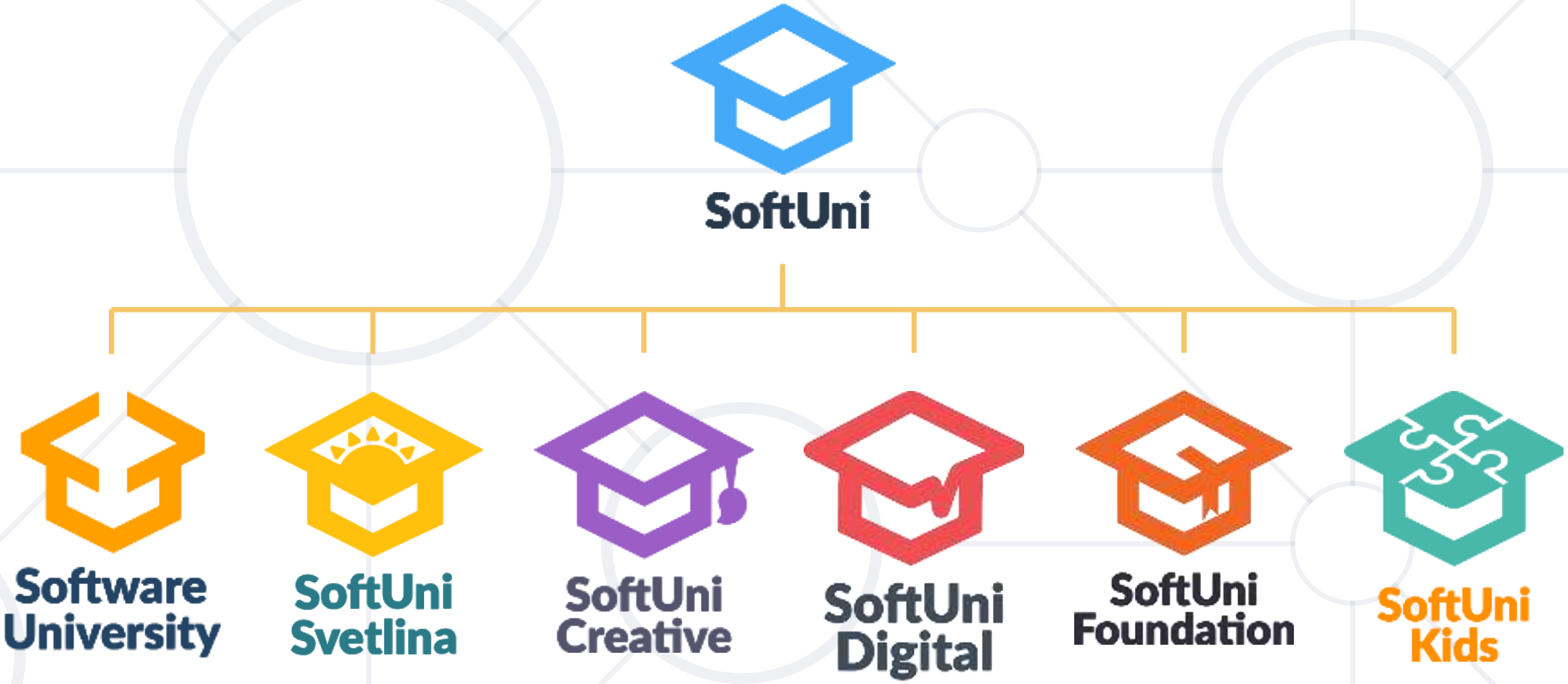
Check your solution here: https://judge.softuni.bg/Contests/1323

Live Exercises

Summary

- Objects hold **key-value pairs**
  - Access key and value **by index** in loops
  - Access value with **[**'key name'**]**
  - Access value with obj.key
- Use Object **Methods** such as:
  - **Object.keys**
  - **Object.values**
- **Parse** and **stringify** objects in **JSON** formats

Questions?


SoftUni
Software University
SoftUni Svetlina
SoftUni Creative
SoftUni Digital
SoftUni Foundation
SoftUni Kids

SoftUni Diamond Partners

SoftUni Foundation

XSsoftware

SBTech
we know sports

telenor

SoftwareGroup
doing it right

NETPEAK

SmartIT

Postbank
Решения за твоето утре

SUPER HOSTING .BG

INDEAVR
Serving the high achievers

INFRAGISTICS

LIEBHERR

æternity

codexio

SoftUni Organizational Partners

SoftUni Foundation

ИНФОРМАЦИОННО ОБСЛУЖВАНЕ

OneBit SOFTWARE

WORLD OF MYTHS

Lukanet.com

codeXio

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education and Employment Opportunities
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license