Objects and Associative Arrays

Objects, JSON, Associative Arrays, Maps and Sets

SoftUni Team

Technical Trainers

Software University

http://softuni.bg

Table of Content

Have a Question?

SoftUni Foundation

sli.do

#JSCORE

Objects in JS

Objects, Properties and JSON

Objects in JS

- Objects in JavaScript hold **key-value pairs**:

```
let obj = { name : "SoftUni", age : 3 };
console.log(obj); // Object {name: "SoftUni", age: 3}
obj['site'] = "https://softuni.bg";
console.log(obj); // Object {name: "SoftUni", age: 3,
site: " https://softuni.bg" }
delete obj.name; // Delete a property
obj.site = undefined; // Delete a property value
console.log(obj); // Object {age: 3, site: undefined}
```

Object Keys and Values

```
let course = { name: 'JS Core', hall: 'Open Source' };
let keys = Object.keys(course);
console.log(keys);  // [ 'name', 'hall' ]
if (course.hasOwnProperty('name'))
console.log(course.name);  // JS Core
```

```
let values = Object.values(course);
console.log(values); // [ 'JS Core', 'Open Source' ]
if (values.includes('JS Core'))
console.log("Found 'JS Core' value");
```

Object Freeze and Seal

```
let cat = { name: 'Tom', age: 5 };
Object.freeze(cat);
cat.age = 10;            // Error in strict mode
cat.gender = 'male';     // Error in strict mode
console.log(cat);        // { name: 'Tom', age: 5 }
```

```
cat = { name: 'Tom', age: 5 };
Object.seal(cat);
cat.age = 10;            // OK
delete cat.age;          // Error in strict mode
console.log(cat);        // { name: 'Tom', age: 10 }
```

Objects and JSON

- JavaScript **objects** can be stored as text in **JSON** format
  - JSON == JavaScript Object Notation == text object format

```
let obj = { name : "SoftUni", age : 3 };
let str = JSON.stringify(obj);
console.log(str); // {"name":"SoftUni","age":3}
```

```
let str = "{\"name\":\"Nakov\",\"age\":24}";
let obj = JSON.parse(str);
console.log(obj); // Object {name: "Nakov", age: 24}
```

Problem: Towns to JSON

- Read an **array of strings**, holding towns with GPS coordinates
  - Parse each string to **JS object** (see the below format)
  - Print the output array of objects as **JSON string**

```
| Town | Latitude | Longitude |
| Sofia | 42.696552 | 23.32601 |
| Beijing | 39.913818 | 116.363625 |
```

⇩

```
[{"Town":"Sofia","Latitude":42.696552,"Longitude":23.32601},
{"Town":"Beijing","Latitude":39.913818,"Longitude":116.363625}]
```

Solution: Towns to JSON

SoftUni Foundation

```
function parseTownsToJSON(towns) {
    let townsArr = [];
    for (let town of towns.slice(1)) {
        let [empty, townName, lat, lng] =
            town.split(/\s*\|\s*/);
        let townObj = { Town: townName, Latitude:
            Number(lat), Longitude: Number(lng) };
        townsArr.push(townObj);
    }
    return JSON.stringify(townsArr);
}
```

```
parseTownsToJSON(['|Town|Lat|Lng|','|Sofia |42|23|'])
```

Check your solution here: https://judge.softuni.bg/Contests/315

Nested Objects in JS

```
let polygon = {
  about: { name: "triangle", color: "red" },
  corners: [{x:2, y:6}, {x:3, y:1}, {x:-2, y:2}]
};

console.log(JSON.stringify(polygon)); // {"about":
{"name":"triangle","color":"red"},"corners":[{"x":2,"y":6},
{"x":3,"y":1},{"x":-2,"y":2}]}

console.log(polygon.about.color); // red

polygon.about.location = {x:4, y:-7};
```

Problem: Score to HTML

- Read a **JSON string**, holding array of objects: **{name, score}**
- Print the objects as **HTML table** like shown below

```
[{"name":"Pesho & Kiro","score":479},{"name":"Gosho,
Maria & Viki","score":205}]
```

```
<table>
  <tr><th>name</th><th>score</th></tr>
  <tr><td>Pesho &amp; Kiro</td><td>479</td></tr>
  <tr><td>Gosho, Maria &amp; Viki</td><td>205</td></tr>
</table>
```

Solution: Score to HTML

```
function scoreToHTMLTable(scoreJSON) {
  let html = "<table>\n";
  html += "  <tr><th>name</th><th>score</th>\n";
  let arr = JSON.parse(scoreJSON);
  for (let obj of arr)
    html += `  <tr><td>${htmlEscape(obj['name'])}` +
      `</td><td>${obj['score']}</td></tr>\n`;
  return html + "</table>";
  function htmlEscape(text) { // TODO … }
}
```

```
scoreToHTMLTable([{"name":"Pesho","score":70}])
```

Check your solution here: https://judge.softuni.bg/Contests/315

Iterating Over Object Values

```
let laptop = { RAM: '8GB', CPU: 'i7 2.20 GHz' };
```

```
for (let key in laptop) {
  console.log(key);               // RAM, CPU
  console.log(laptop[key]);  // 8GB, i7 2.20 GHz
}
```

```
for (let value of laptop) {
  // TypeError: laptop is not iterable
}
```

Problem: From JSON to HTML Table

- Read a **JSON string**, holding array of JS objects (key / value pairs)
  - Print the objects as **HTML table** like shown below

```
[{"Name":"Tomatoes & Chips","Price":2.35},{"Name":"J&B
Chocolate","Price":0.96}]
```

```
<table>
  <tr><th>Name</th><th>Price</th></tr>
  <tr><td>Tomatoes &amp; Chips</td><td>2.35</td></tr>
  <tr><td>J&amp;B Chocolate</td><td>0.96</td></tr>
</table>
```

Solution: From JSON to HTML Table

```
function JSONToHTMLTable(json) {
  let html = "<table>\n";
  let arr = JSON.parse(json);
  html += "  <tr>";
  for (let key of Object.keys(arr[0]))
    html += `<th>${htmlEscape(key)}</th>`;
    html += "</tr>\n";
  for (let obj of arr) {
  // TODO: print obj values in <tr><td>…</td></tr>
  return html + "</table>";
  function htmlEscape(text) { // TODO … }
}
```

```
JSONToHTMLTable(['[{"X":5,"Y":7},{"X":2,"Y":4}]'])
```

Check your solution here: https://judge.softuni.bg/Contests/315

Associative Arrays

Objects as Associative Arrays in JS

Associative Arrays (Maps, Dictionaries)

- **Associative arrays** (**maps** / **dictionaries**) == arrays indexed by keys
  - Not by numbers 0, 1, 2, …
- Hold a set of pairs **{key -> value}**, just like JS object

Traditional array

key	0	1	2	3	4
value	8	-3	12	408	33

Associative array (dictionary)

key	value
John Smith	+1-555-8976
Lisa Smith	+1-555-1234
Sam Doe	+1-555-5030

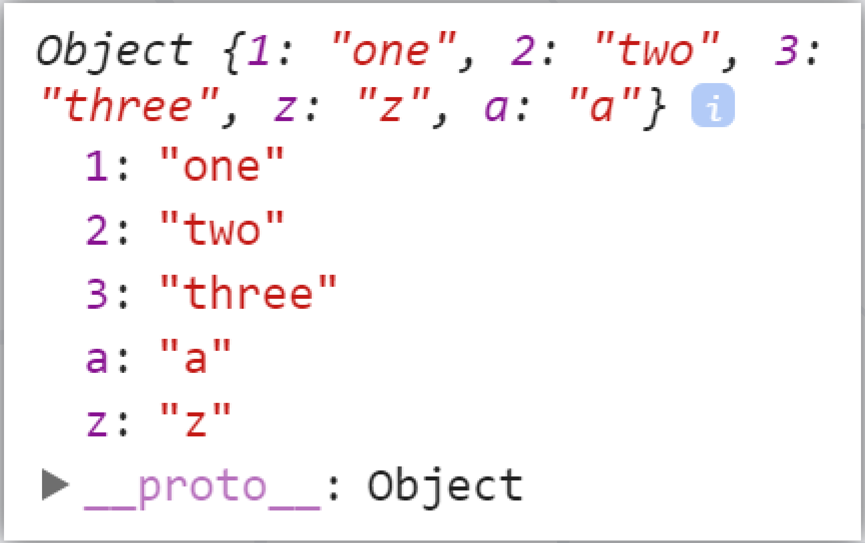
Phonebook - Associative array Example

```
let phonebook = { };
phonebook["John Smith"] = "+1-555-8976";   // Add
phonebook["Lisa Smith"] = "+1-555-1234";
phonebook["Sam Doe"] = "+1-555-5030";
phonebook["Nakov"] = "+359-899-555-592";
phonebook["Nakov"] = "+359-2-981-9819";   // Replace

delete phonebook["John Smith"];             // Delete

console.log(Object.keys(phonebook).length); // 3

for (let key in phonebook) {                // Print
  console.log(`${key} -> ${phonebook[key]}`);
}
```

The Order of Keys in JS Object

- The **order of keys** in JS objects in unspecified!

```
let obj = {
  "1": 'one',
  "3": 'three',
  "2": 'two',
  "z": 'z',
  "a": 'a'
};
console.log(Object.keys(obj)); // ["1", "2", "3", "z", "a"]
console.log(obj); // Object {1: "one", 2: "two", 3: "three",
z: "z", a: "a"}
```



```
Object {1: "one", 2: "two", 3:
"three", z: "z", a: "a"}
  1: "one"
  2: "two"
  3: "three"
  a: "a"
  z: "z"
▶ __proto__: Object
```

Problem: Sum by Town

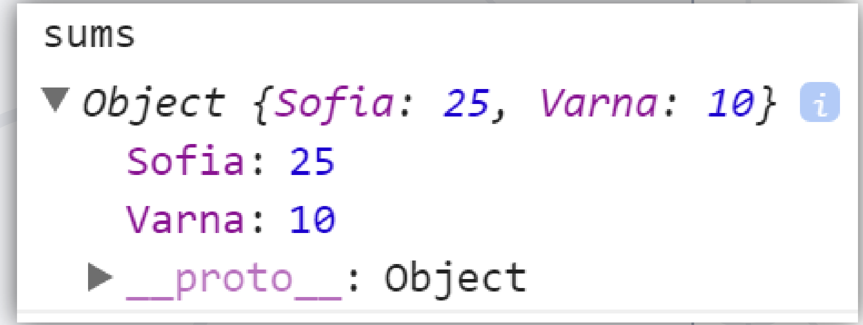
- Read **towns** and **incomes** (like shown below) and print a **JSON object** holding **the total** income for each **town** (see below)
  - Print the towns in their **natural order** as object properties

```
Sofia
20
Varna
3
Sofia
5
Varna
4
```

```
{"Sofia":"25","Varna":"7"}
```

Solution: Sum of Towns

```
function sumOfTowns(arr) {
  let sums = {};
  for (let i=0; i<arr.length; i+=2) {
    let [town, income] = [arr[i], Number(arr[i+1])];
    if (sums[town] == undefined){
      sums[town] = income;
    } else{
      sums[town] += income;
    }
  }
  return JSON.stringify(sums);
}
```



```
sums
▼ Object {Sofia: 25, Varna: 10}
    Sofia: 25
    Varna: 10
  ▶ __proto__: Object
```

```
sumOfTowns(['Sofia','20', 'Varna','10', 'Sofia','5'])
```

Check your solution here: https://judge.softuni.bg/Contests/315

Problem: Count Word in a Text

- Write a JS function to **count** the **words** in a text (case sensitive)
  - Words are sequence of **letters**, **digits** and **_**
  - The **input** text comes as **array of strings**
  - Return the **output** as **JSON string**

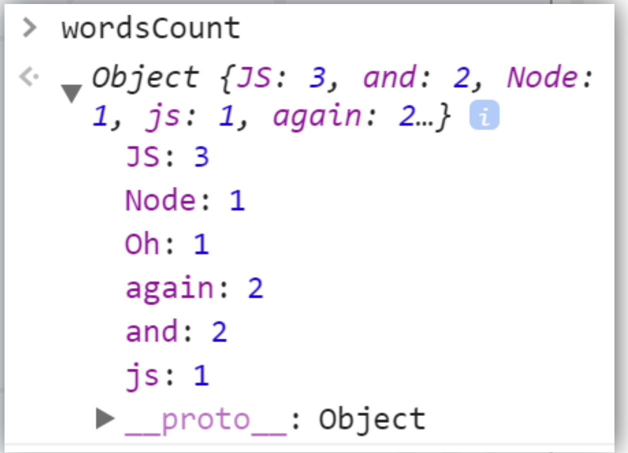
```
JS devs use Node.js for server-side JS.
-- JS for devs
```

```
{"JS":3,"devs":2,"use":1,"Node":1,"js":1,"for":2,
"server":1,"side":1}
```

Solution: Count Words in a Text

```
function countWords(inputLines) {
  let text = inputLines.join('\n');
  let words = text.split(/[^A-Za-z0-9_]+/)
    .filter(w => w != '');
  let wordsCount = {};
  for (let w of words){
    wordsCount[w] ? wordsCount[w]++ :
    wordsCount[w] = 1;
  }
  return JSON.stringify(wordsCount);
}
```

```
countWords(['JS and Node.js', 'JS again and again', 'Oh, JS?'])
```



```
> wordsCount
<· ▼Object {JS: 3, and: 2, Node: 1, js: 1, again: 2…}
      JS: 3
      Node: 1
      Oh: 1
      again: 2
      and: 2
      js: 1
    ▶ __proto__: Object
```

Check your solution here: https://judge.softuni.bg/Contests/315

Live Exercises

Practice: JS Objects & JSON

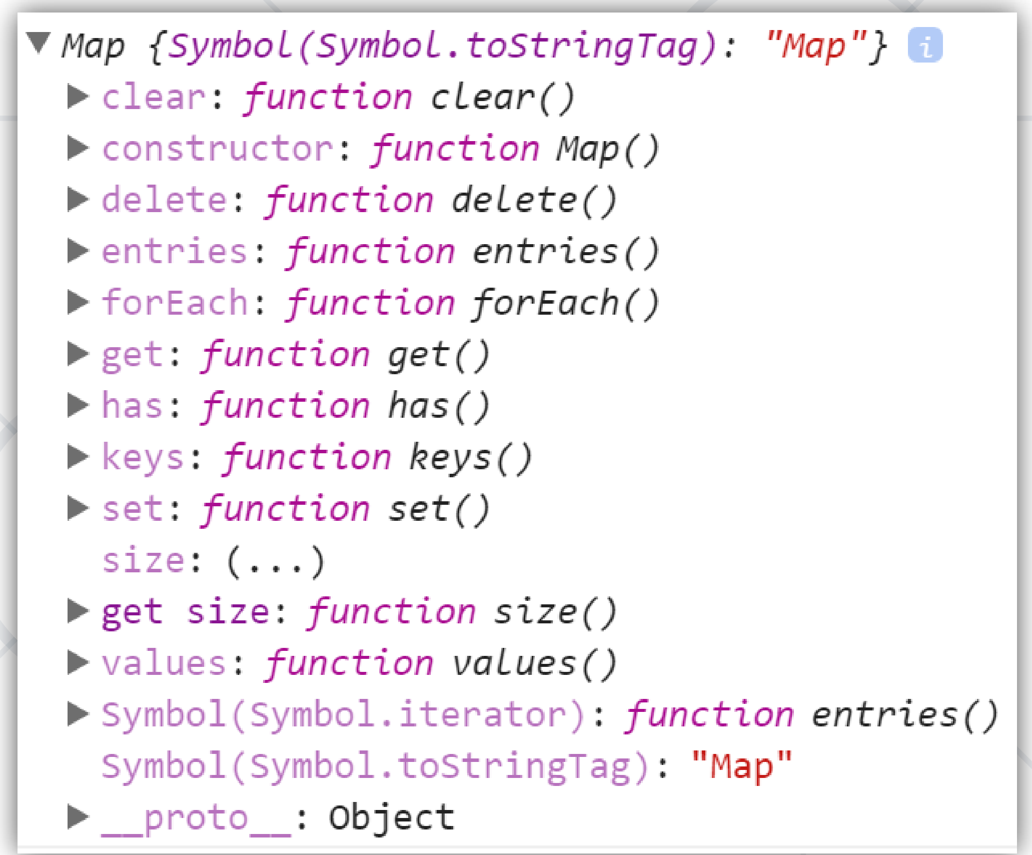
The Map Class in JS

Key / Value Map

The Map Class in JS

- The **Map** class holds **{ key -> value }** map
- Better functionality than plain JS object

```
let score = new Map();
score.set("Peter", 130);
score.set("Maria", 85);
for (let [k, v] of score){
  console.log(k + ' -> ' + v);
}
```



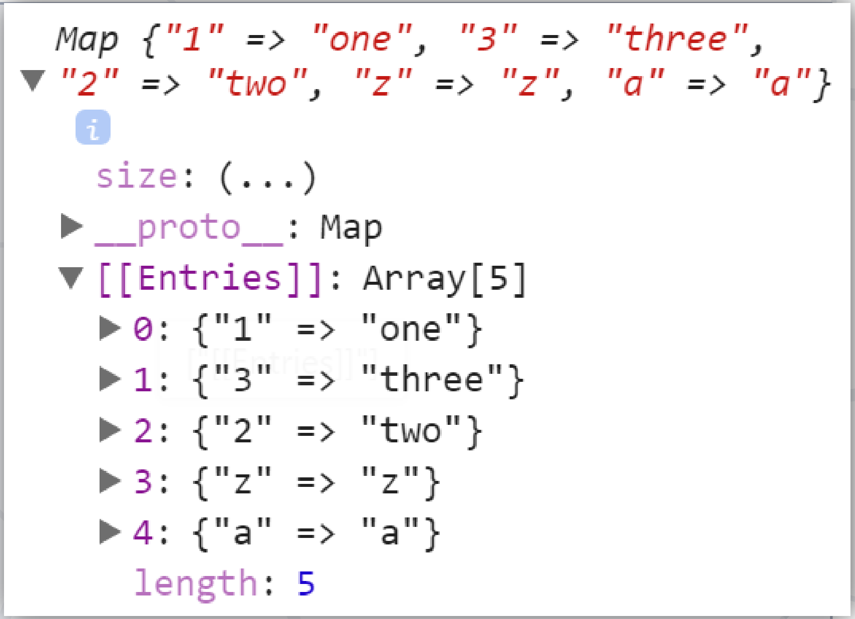
```
▼ Map {Symbol(Symbol.toStringTag): "Map"}
  ▶ clear: function clear()
  ▶ constructor: function Map()
  ▶ delete: function delete()
  ▶ entries: function entries()
  ▶ forEach: function forEach()
  ▶ get: function get()
  ▶ has: function has()
  ▶ keys: function keys()
  ▶ set: function set()
    size: (...)
  ▶ get size: function size()
  ▶ values: function values()
  ▶ Symbol(Symbol.iterator): function entries()
    Symbol(Symbol.toStringTag): "Map"
  ▶ __proto__: Object
```

Phonebook - Map Example

```
let phonebook = new Map();
phonebook.set("John Smith", "+1-555-8976"); // Add
phonebook.set("Lisa Smith","+1-555-1234");
phonebook.set("Sam Doe", "+1-555-5030");
phonebook.set("Nakov", "+359-899-555-592");
phonebook.set("Nakov", "+359-2-981-9819"); // Replace

phonebook.delete("John Smith"); // Delete
console.log(phonebook.size); // 3

for (let [key, value] of phonebook){ // Print
  console.log(`${key} -> ${value}`);
}
```

Maps Preserve the Insertion Order of Keys

```
let map = new Map([
["1", 'one'],
["3", 'three'],
["2", 'two'],
["z", 'z'],
["a", 'a']
]);

console.log(map);
// Map {"1" => "one", "3" => "three", "2" => "two",
"z" => "z", "a" => "a"}
console.log(Array.from(map.keys()));
// ["1", "3", "2", "z", "a"]
```



```
  Map {"1" => "one", "3" => "three",
▼ "2" => "two", "z" => "z", "a" => "a"}
    size: (...)
  ▶ __proto__: Map
  ▼ [[Entries]]: Array[5]
    ▶ 0: {"1" => "one"}
    ▶ 1: {"3" => "three"}
    ▶ 2: {"2" => "two"}
    ▶ 3: {"z" => "z"}
    ▶ 4: {"a" => "a"}
      length: 5
```

Problem: Count Words in a Text (with Map)

- Write a JS function to **count** the **words** in a text (case sensitive)
  - Words are sequence of **letters**, **digits** and **_**
  - The **input** comes as **array of strings**
  - Order alphabetically the **output** words

```
JS devs use Node.js for
server-side JS.
JS devs use JS.
-- JS for devs --
```

```
'devs' -> 3 times
'for' -> 2 times
'js' -> 6 times
'node' -> 1 times
'server' -> 1 times
'side' -> 1 times
'use' -> 2 times
```

Solution: Count Words in a Text (with Map)

```
function countWords(inputLines) {
  let words = inputLines.join('\n').toLowerCase()
    .split(/[^A-Za-z0-9_]+/).filter(w => w != '');
  let wordsCount = new Map();
  for (let w of words)
    wordsCount.has(w) ? wordsCount.set(w,
      wordsCount.get(w)+1) : wordsCount.set(w, 1);
  let allWords = Array.from(wordsCount.keys()).sort();
  allWords.forEach(w =>
    console.log(`'${w}' -> ${wordsCount.get(w)} times`));
}
```

```
countWords(['JS and Node.js', 'JS again and again', 'Oh, JS?'])
```

Check your solution here: https://judge.softuni.bg/Contests/315

Problem: Population in Towns

- Read **towns** and **populations** (like shown below) and print a **the towns** ant their **total population** for each town (see below)
  - Print the towns in the order of their first appearance

```
Varna <-> 40000
Sofia <-> 1200000
Plovdiv <-> 20000
Sofia <-> 100000
Varna <-> 420000
Plovdiv <-> 400000
Plovdiv <-> 50000
```

→

```
Varna : 460000
Sofia : 1300000
Plovdiv : 470000
```

Solution: Population in Towns

```
function populationInTowns(dataRows) {
  let total = new Map();
  for (let dataRow of dataRows) {
    let [town, population] = dataRow.split(/\s*<->\s*/)
    population = Number(population);
    if (total.has(town))
      total.set(town, total.get(town) + population);
    else total.set(town, population);
    }
  for (let [town, sum] of total)
    console.log(town + " : " + sum);
}
```

populationInTowns(['B<->20', 'A<->30', 'B<->5'])

Check your solution here: https://judge.softuni.bg/Contests/315

Problem: City Markets

- Read **sales data** in the following format

```
{town} -> {product} -> {amountOfSales}:{priceForOneUnit}
```

- Print for each **town** the **sum** of **incomes** for each **product**
  - **Order** the towns and products as they **first appear**

```
Sofia -> Laptops HP -> 200 : 2000
Sofia -> Raspberry -> 200000 : 1500
Montana -> Oranges -> 200000 : 1
Montana -> Cherries -> 1000 : 0.3
Sofia -> Audi Q7 -> 200 : 100000
```

```
Town - Sofia
$$$Laptops HP : 400000
$$$Raspberry : 300000000
$$$Audi Q7 : 20000000
Town - Montana
$$$Oranges : 200000
$$$Cherries : 4000
```

Solution: City Markets (Nested Maps)

```
function cityMarkets(sales) {
  let townsWithProducts = new Map();
  for (let sale of sales) {
    let [town, product, quantityPrice] = sale.split(/\s*->\s*/);
    let [quantity, price] = quantityPrice.split(/\s*:\s*/);
    if (!townsWithProducts.has(town))
      townsWithProducts.set(town, new Map());
    let income = quantity * price;
    let oldIncome = townsWithProducts.get(town).get(product);
    if (oldIncome) income += oldIncome;
    townsWithProducts.get(town).set(product, income);
  }
  // TODO: print the incomes by towns and products
}
```

Check your solution here: https://judge.softuni.bg/Contests/315

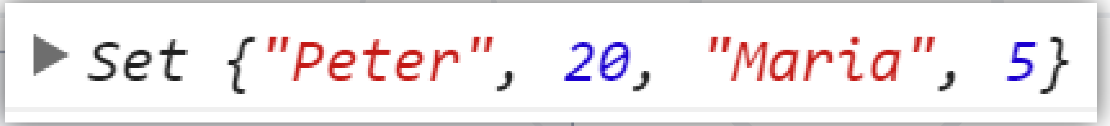
The Set Class in JS

Set of Unique Values of Any Type

The Set Class in JS

- **Sets** in JS are collections of **unique objects**
  - The **insertion order** is **preserved**, with **no duplicates**

```
let names = new Set();
names.add("Peter"); names.add(20);
names.add("Maria"); names.add(5);

console.log(names.has('Peter')); // true

names.add("Maria"); // Duplicates are skipped

names.delete(20); // Delete element if exists

for (let name of names) console.log(name);
```



```
▶ Set {"Peter", 20, "Maria", 5}
```

Problem: Extract Unique Words

- Write a JS function to extract all **unique words** from a text (case insensitive)
  - Words are sequences of **letters**, **digits** and **_**
  - The **input** comes as **array of strings**
  - The **output** should hold the words in their **order of appearance**

```
JS devs use Node.js for
server-side JS.
JS devs use JS.
-- JS for devs --
```

```
js, devs, use, node,
for, server, side
```

Solution: Extract Unique Words

```
function extractWords(inputSentences) {
  let wordPattern = /\b[a-zA-Z0-9_]+\b/g;
  let words = new Set();
  for (let sentence of inputSentences) {
   let matches = sentence.match(wordPattern);
   matches.forEach(x=>words.add(x.toLowerCase()));
  }
  console.log([...words.values()].join(", "));
}
```

```
extractWords(['JS and Node.js', 'JS again and again', 'Oh, JS?'])
```

Check your solution here: https://judge.softuni.bg/Contests/315

Live Exercises

Practice: Using Maps and Sets

Summary

- **Objects** in JS hold key-value pairs

```
let obj = { name : "SoftUni", age : 3 };
obj.age++;
obj[town] = 'Sofia';
delete obj.name;
```

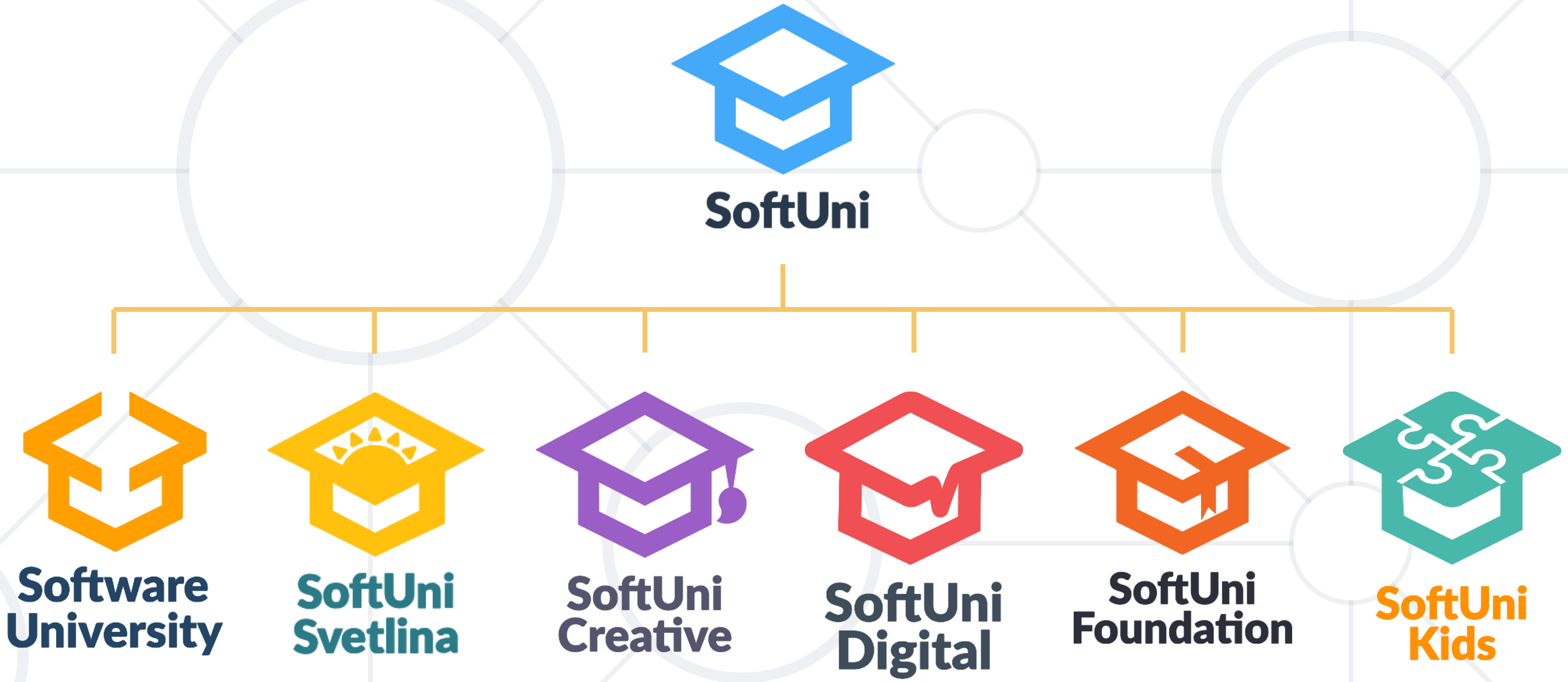
- **Maps** map key to values, preserve key order

```
let map = new Map();
map.set('score', 20);
```

- **Sets** hold unique collection of values

```
let map = new Set(); set.add(5);
```

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