CSIT884:
Web Development

JavaScript Objects & JSON

School of Computing and Information Technology
University of Wollongong

Array

An array is an ordered collection of values, where each value is called an element.

```
var arrayName = [item0, item1, ...];
var emptyArray = [];
```

Array

An array is an ordered collection of values, where each value is called an element.

```
var subjectList = ["ISIT206", "MATH121", ”CSIT884"]; . . (1)
subjectList[1] = "CSIT883"; . . . (2)
alert(subjectList[0]); . . . (3)

subjectList[6] = "LAW201"; . . . (4)
// this will create holes in array
```

Array

```
var arrayName = new Array (item0, item1, ...); . . . (1)
var emptyArray = new Array(); . . . (2)
var empElArray = new Array(number_of_elements); . . . (3)
```

```
var subjectList = new array("ISIT206", "MATH121"); . . . (1)

var subjectListEm = new array();. . . (2)

var subjectListEl = new array(3);. . . (3)
```

Array

Length `property` - specifies the number of elements in the array

```
arrayName.length
```

Loop through an array

```
for(var i = 0; i < subjectList.length; i++){
      alert(subjectList[i]);
}                                             . . . (1)
for (subject of subjectList) {
      alert(subject);
}                                  . . . (2)
subjectList.forEach(myFunction);
function myFunction(subject, index) {
      alert(subject + '-' + index);
}                                        . . . (3)
```

Array

The `push()` method adds a new element to the end of an array

```
var square = [];  //empty array
for(var i = 0; i < 10; i++) {
    square.push(i*i);
}
square.push(10, 20);

for(var i = 0; i < square.length; i++) {
    alert(square[i]);
}
```

Array

The **`pop()`** method removes the last element in an array

```
var square = [];  //empty array
for(var i = 0; i < 10; i++) {
    square.push(i*i);
}
var last = square.pop();
alert(last);
for(var i = 0; i < square.length; i++) {
    alert(square[i]);
}
```

JavaScript: Object

Object is defined by a list of `property:value`

```
var objectName = {property1:value1, property2:value2, ...};
```

```
var emptyObject = {};
```

```
var info = {
  name: "John",
  dob: new Date("1996-01-20"),
  year: 2
};
```

```
// two ways:
info.year
info["year"]
```

Object values can be obtained by **two ways**:

```
obj.property
```

```
obj["property"]
```

Object

Change the values of an object

```
var info = {
  name: "John",
  dob: new Date("1996-01-20"),
  year: 2
};
```

```
// two ways:
info.year = 1;
info["year"] = 1;
```

Object

Delete object properties

```
var info = {
  name: "John",
  dob: new Date("1996-01-20"),
  year: 2
};

// two ways:
delete info.year;
delete info["year"];
```

Object

Create an empty object

```
var info = { };


info.firstName = "John";

info["lastName"] = "Lee";


alert(info["firstName"]);

alert(info.lastName);
```

Object

Objects can also have methods

```
var info = {
  name: "John",
  dob: new Date("1996-01-20"),
  year: 2,
  print: function(){
       return this.name + "-" + this.dob;
  }
};
```

```
info.print();
```

Object method can be **accessed** by:

`obj.methodName()`

Array vs Object

```
var arrayName = [item0, item1, ...];

var objectName = {property1:value1, property2:value2, ...};
```

Arrays use numbered index:

```
arrayName[0] = "LOGIC101";
arrayName[1] = "CSCI111";
```

Objects use named index:

```
objectName["firstName"] = "John";

objectName.lastName = "Lee";
```

Array Sorting

```
var subjects = ["ISIT206", "MATH121", "CSIT884"];
subjects.sort();

Now subjects is ["CSIT884", "ISIT206", "MATH121"]. . . (1)
```

```
var numbers = [1, 20, -3, 4];
numbers.sort();

Now numbers is [-3, 1, 20, 4] !!!. . . (2)
```

```
numbers.sort(function (a, b) {  return a - b;  });
Now numbers is [-3, 1, 4, 20]                        . . . (3)
```

Array Sorting

In general:

```
the_array_to_be_sorted.sort(the_sorting_function ...);
```

The sorting function `function (a, b)` must

- return a **positive value** to indicate `a > b`
- return a **negative value** to indicate `a < b`
- return **zero** to indicate `a = b`

```
var numbers = [1, 20, -3, 4];
numbers.sort(function (a, b) {  return a - b;  });
Now numbers is [-3, 1, 4, 20]
```

Array Sorting

```
ninja_results = [
  {name: "John", level: 4, seconds: 85},
  {name: "Peter", level: 2, seconds: 35},
  {name: "Kate", level: 4, seconds: 80},
  {name: "Luke", level: 5, seconds: 120}
];
```

We want to sort the ninja results based on the level first, if two objects (persons) achieved the same level, then we compare the number of seconds.

Array Sorting

```
ninja_results.sort(
  function (p1, p2) {
    if (p1["level"] > p2["level"]){
       return 1;
    }
    if (p1["level"] < p2["level"]){
       return -1;
    }
    //at this point the two objects
    //have the same level
    if (p1["seconds"] < p2["seconds"]){
       return 1;
    }
    if (p1["seconds"] > p2["seconds"]){
       return -1;
    }
    return 0;
  }
);
```

Before sorting

```
ninja_results = [
  {name: "John", level: 4, seconds: 85},
  {name: "Peter", level: 2, seconds: 35},
  {name: "Kate", level: 4, seconds: 80},
  {name: "Luke", level: 5, seconds: 120}
];
```

After sorting

```
ninja_results = [
  {name: "Peter", level: 2, seconds: 35},
  {name: "John", level: 4, seconds: 85},
  {name: "Kate", level: 4, seconds: 80},
  {name: "Luke", level: 5, seconds: 120}
];
```

JavaScript Object Notation (JSON)

- In most web applications, XML and JSON are used to store or transport data
- JSON is "self-describing" and easy to understand

This is an example of a JSON describing a student object:

```
{
  "fullname": "John Smith",
  "studentNumber": "U1234567",
  "age": 20,
  "csMajor": true
}
```

JSON

- Data is in name/value pairs
- Data is separated by commas
- Curly braces hold objects

```
{
  "fullname": "John Smith",
  "studentNumber": "U1234567",
  "age": 20,
  "csMajor": true
}
```

JSON

Square brackets hold arrays

```
[
  {
    "firstName":"John",
    "lastName":"Smith"
  },
  {
    "firstName":"Kate",
    "lastName":"Williams"
  }
]
```

JSON

- Curly braces hold objects
- Square brackets hold arrays

```
{
  "firstName":"John",
  "lastName":"Smith",
  "subjectList":[
    {
      "code":"MATH101",
      "title":"Algebra"
    },
    {
      "code":"CSIT122",
      "title":"Programming"
    }
  ]
}
```

JSON

Translate from Javascript object to JSON string

```
objJSON = JSON.stringify(obj);
```

Translate from JSON string to javascript object

```
obj = JSON.parse(objJSON);
```

JSON

OBJECT

```
{
  fullname: "John Smith",
  studentNumber: "U1234567",
  age: 20,
  csMajor: true
}
```

$$\begin{tikzcd}\text{OBJECT} \arrow[r, bend right, "\text{JSON.stringify}"'] & \text{JSON} \arrow[l, bend right, "\text{JSON.parse}"']\end{tikzcd}$$

JSON

```
{
  "fullname": "John Smith",
  "studentNumber": "U1234567",
  "age": 20,
  "csMajor": true
}
```

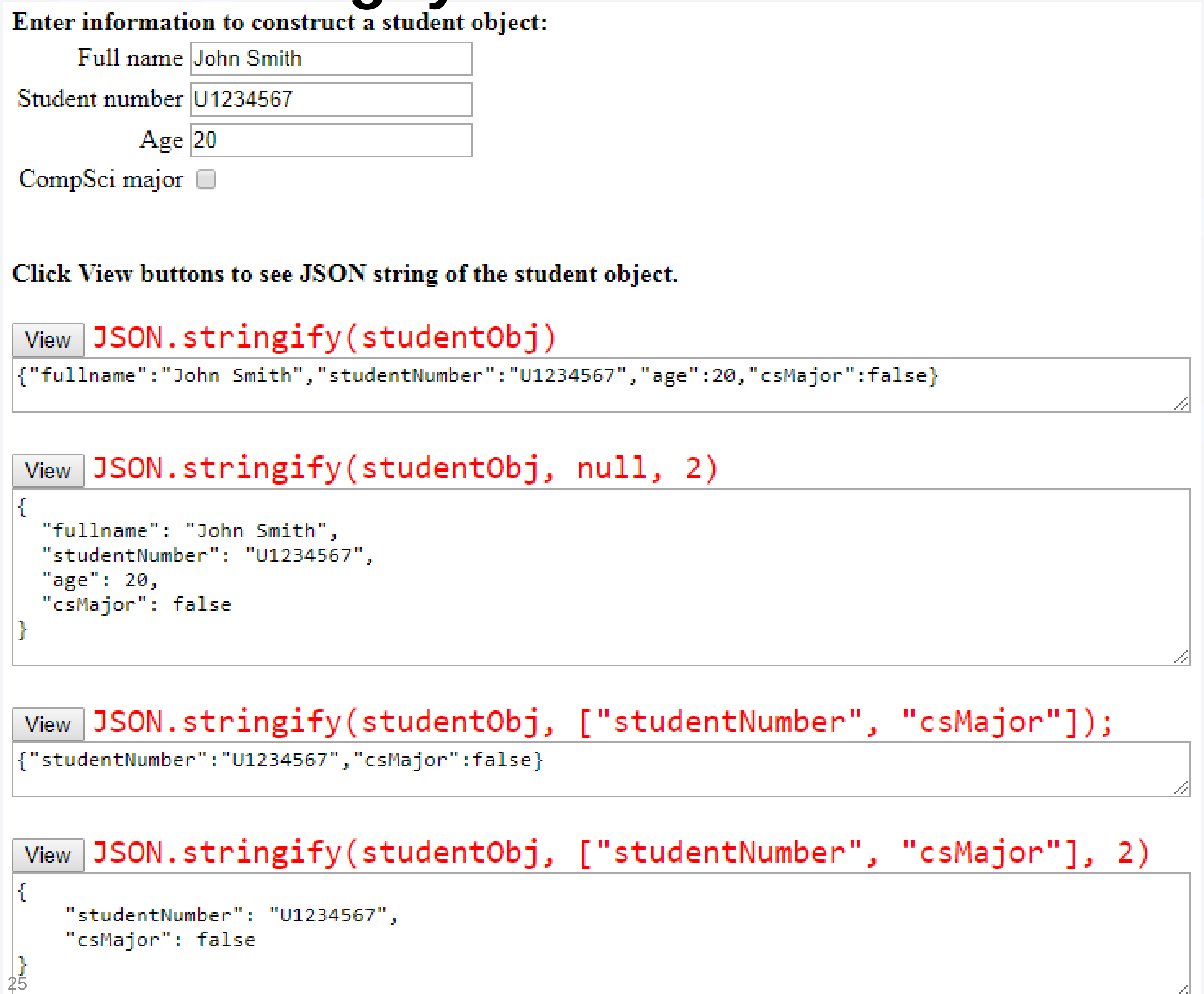
JSON.stringify

The **`JSON.stringify`** method converts a JavaScript value to a JSON string.

```
Syntax:
    JSON.stringify(jsvalue, replacer, space)
```

- `jsvalue:` the javascript value to convert to a JSON string.
- `replacer` (Optional): selecting/filtering which properties of the object to be included in the JSON string. If the `replacer` is null or not provided, all properties of the object are included in the resulting JSON string.
- `space` (Optional): use for indentation, specifying white spaces in the output JSON string for readability purposes.

JSON.stringify function demo



Enter information to construct a student object:

Full name	John Smith
Student number	U1234567
Age	20
CompSci major	☐

Click View buttons to see JSON string of the student object.

View `JSON.stringify(studentObj)`

```
{"fullname":"John Smith","studentNumber":"U1234567","age":20,"csMajor":false}
```

View `JSON.stringify(studentObj, null, 2)`

```
{
  "fullname": "John Smith",
  "studentNumber": "U1234567",
  "age": 20,
  "csMajor": false
}
```

View `JSON.stringify(studentObj, ["studentNumber", "csMajor"]);`

```
{"studentNumber":"U1234567","csMajor":false}
```

View `JSON.stringify(studentObj, ["studentNumber", "csMajor"], 2)`

```
{
    "studentNumber": "U1234567",
    "csMajor": false
}
```

JSON.stringify

```
var studentObj = {
  fullname: "John Smith",
  studentNumber: "U1234567",
  age: 20,
  csMajor: false
};
```

```
JSON.stringify(studentObj)
```

```
{"fullname":"John Smith","studentNumber":"U1234567","age":20,"csMajor":false}
```

output JSON sticks together
make it hard to read

JSON.stringify

```
var studentObj = {
  fullname: "John Smith",
  studentNumber: "U1234567",
  age: 20,
  csMajor: false
};
```

```
JSON.stringify(studentObj, null, 2)
```

using 2 spaces indentation

```
{
  "fullname": "John Smith",
  "studentNumber": "U1234567",
  "age": 20,
  "csMajor": false
}
```

JSON.stringify

```
var studentObj = {
  fullname: "John Smith",
  studentNumber: "U1234567",
  age: 20,
  csMajor: false
};
```

```
JSON.stringify(studentObj, ["studentNumber", "csMajor"])
```

only output the student number and compsci major

```
{"studentNumber":"U1234567","csMajor":false}
```

JSON.stringify

```
var studentObj = {
  fullname: "John Smith",
  studentNumber: "U1234567",
  age: 20,
  csMajor: false
};
```

```
JSON.stringify(studentObj, ["studentNumber", "csMajor"], 2)
```

only output the student number and compsci major, using 2 spaces indentation

```
{
  "studentNumber":"U1234567",
  "csMajor":false
}
```

Example 1: `JSON.stringify`

```
<button onClick="showObjectJSON()">
Click here to see JSON string
</button>
<p id="out"></p>
```

```
function showObjectJSON(){
  //create a student object
  var studentObj = {};
  studentObj.fullname = "John Smith";
  studentObj.studentNumber = "U1234567";
  studentObj.age = 20;
  studentObj.csMajor = true;

  //get JSON string from the javascript object
  var studentJSON = JSON.stringify(studentObj);

  //print the JSON string to the web
  document.getElementById("out").innerHTML = studentJSON;
}
```

Example 2: `JSON.parse`

```
<button onClick="showObject()">
Click here to see object from JSON
</button>
<p id="out"></p>
```

```
function showObject(){
 //JSON string
 var studentJSON = '{"fullname":"John Smith","studentNumber":
"U1234567","age":20,"csMajor":true}';

 //get javascript object from JSON string
 var studentObj = JSON.parse(studentJSON);

 //print the object to the web
 document.getElementById("out").innerHTML = studentObj.fullname;

//what will be the outcome of this line?
//console.log(studentObj);
}
```

Example 3: JSON.stringify

```
function showArrayJSON(){
  var user1 = {};
  user1.firstName = "John";
  user1.lastName = "Smith";

  var user2 = {};
  user2.firstName = "Kate";
  user2.lastName = "Williams";

  //create an array of user objects
  var userList = [user1, user2];

  //get JSON string from the javascript array
  var userListJSON = JSON.stringify(userList);

  //print the JSON string to the web
   document.getElementById("out").innerHTML = userListJSON;
}
```

```
<button onClick="showArrayJSON()">
Click here to see JSON string
</button>
<p id="out"></p>
```

Example 4: JSON.parse

```
function showArray(){
 //JSON string
 var userListJSON = '[{"firstName":"John","lastName":"Smith"},
                  {"firstName":"Kate","lastName":"Williams"}]';

 //get javascript array from JSON string
 var userList = JSON.parse(userListJSON);

 //print the number of objects to the console
 document.getElementById("out").innerHTML = userList.length;

 document.getElementById("out").innerHTML = userList[0].lastName;
}
```

```
<button onClick="showArray()">
Click here to see array from JSON
</button>
<p id="out"></p>
```

Example 5: `JSON.stringify`

```
function showObjectJSON(){
  var studentObj = {}; //create a student object
  studentObj.firstName = "John";
  studentObj.lastName = "Smith";
  studentObj.subjectList = []; //empty array to hold subjects

  var subjectObj1 = {};
  subjectObj1.code = "MATH101";
  subjectObj1.title = "Algebra";
  //add subject into array
  studentObj.subjectList.push(subjectObj1);

  var subjectObj2 = {};
  subjectObj2.code = "CSIT122";
  subjectObj2.title = ”Programming";
  //add subject into array
  studentObj.subjectList.push(subjectObj2);

  //get JSON string from obj and print it on console
  var studentJSON = JSON.stringify(studentObj, null, 2);
  console.log(studentJSON);
}
```

Example 5: `JSON.stringify`

```
{
  "firstName":"John",
  "lastName":"Smith",
  "subjectList":[
    {
      "code":"MATH101",
      "title":"Algebra"
    },
    {
      "code":"CSIT122",
      "title":"C programming"
    }
  ]
}
```