LESSON 10 DESTRUCTURING ASSIGNMENTS AND JSON

The Nature of Life Is to Grow

Wholeness: JavaScript is a rapidly evolving language. This lesson covers several new data type features that make the language more efficient and powerful. *Science of Consciousness:* The nature of life is to grow and evolve to greater accomplishment and fulfillment.

Main Points

1. Destructuring assignment
2. JSON

Main Point Preview: Destructuring assignments

Destructuring assignment is a new (ES6) and now widely used convenience syntax that allows us to “unpack” arrays or objects into a set of variables.
Science of Consciousness: This feature allows developers to more quickly accomplish common coding tasks. Do less and accomplish more.

Destructuring assignment

- special syntax that allows us to “unpack” arrays or objects into a set of variables

```
let arr = ["Ilya", "Kantor"]

// sets firstName = arr[0] and surname = arr[1]
let [firstName, surname] = arr;
```

- It works great when combined with split or other array-returning methods:

```
let [firstName, surname] = "Ilya Kantor".split(' ‘);
```

- Does not change the array.

Convenience syntax for multiple assignments

- ‘Syntactic sugar’ to replace the following:

```
let firstName = arr[0];
let surname = arr[1];
```

- Does not change the array.

- syntax sugar for calling for..of over the value to the right of = and assigning the values.

Destructuring assignment 2

Unwanted elements of the array can also be thrown away via an extra comma:

```
// second element is not needed
let [firstName, , title] = ["Julius", "Caesar", "Consul", "of the Roman Republic"];
alert( title ); // Consul
```

can use any “assignables” at the left side.

```
let user = {};
[user.name, user.surname] = "Ilya Kantor".split(' ');
alert(user.name); // Ilya
```

Destructuring assignment -- …rest

- The rest '…'
  - can add one more parameter that gets "the rest" using three dots "...":
  - value of rest is array of remaining elements.
  - can use any other variable name in place of rest
    - three dots before it
    - last in the destructuring assignment.

```
let [name1, name2, ...rest] = ["Julius", "Caesar", "Consul", "of the Roman Republic"];
alert(name1); // Julius
alert(name2); // Caesar

// Note that type of `rest` is Array.
alert(rest[0]); // Consul
alert(rest[1]); // of the Roman Republic
alert(rest.length); // 2
```

Destructuring assignment -- missing and default values

- Default values
  - If different number of values in array than variables in assignment, there will be no error.
  - Absent values are considered undefined:
  - Extra values are ignored

```
let [firstName, surname] = [];
alert(firstName); // undefined
alert(surname); // undefined
```

- If we want a “default” value to replace the missing one, we can provide it using =:

```
// default values
let [name = "Guest", surname = "Anonymous"] = ["Julius"];
alert(name);    // Julius (from array)
alert(surname); // Anonymous (default used)
```

Exercise

```
const team = [ "Bob",  "Fred",  "Jim“]
// destructure the team array onto variables with the same names as the elements, but all lower case



console.log("expect Bob", bob );
console.log("expect Jim", jim);
```

Object destructuring

- destructuring assignment also works with objects

```
let options = {
  title: "Menu",
  width: 100,
  height: 200
};
let {title, width, height} = options;
alert(title);  // Menu
alert(width);  // 100
alert(height); // 200
```

- Properties are assigned to the corresponding variables.
  - order does not matter

```
let {height, width, title} = { title: "Menu", height: 200, width: 100 }
```

Destructure property to another name

- to assign a property to a variable with another name, set it using a colon

```
let options = {
  title: "Menu",
  width: 100,
  height: 200
};

// { sourceProperty: targetVariable }
let {width: w, height: h, title} = options;

// width -> w
// height -> h
// title -> title

alert(title);  // Menu
alert(w);      // 100
alert(h);      // 200
```

wrap destructuring expression in parentheses (---)

- use existing variables without let. there's a catch.

```
let title, width, height;

// error in this line
{title, width, height} = {title: "Menu", width: 200, height: 100};
```

- JavaScript assumes a code block instead of destructuring

- To show it's not a code block, wrap the expression in parentheses (---):

```
// okay now
({title, width, height} = {title: "Menu", width: 200, height: 100});
```

Object destructuring – default values and parameters

- For potentially missing properties can set default values using "="

```
let options = {
  title: "Menu"
};
let {width = 100, height = 200, title} = options;
alert(title);  // Menu
alert(width);  // 100
alert(height); // 200
```

- can pass parameters as an object, and the function destructures them into parameters:

```
let options = {
  title: "My menu",
  items: ["Item1", "Item2"]
};
function showMenu({title = "Untitled", width = 200, height = 100, items = []}) {
  // title, items – taken from options,
  // width, height – defaults used
  alert( `${title} ${width} ${height}` ); // My Menu 200 100
  alert( items ); // Item1, Item2
}
showMenu(options);
```

Exercise

```
const team = { point: "Bob", shooting: "Fred", power: "Jim", small: "Al",  center:  "Big Sleep" }

// 1. destructure the team object onto variables with the same names as the properties



console.log("expect Big Sleep", center);
console.log("expect Jim", power);



// 2. destructure the team onto variables:  one (point guard), two (shooting guard), three (small forward), four (power forward) and five (center)



console.log("expect Jim: ", four);
console.log("expect Bob: ", one);
```

destructuring exercises

- Destructuring assignment

- The maximal salary
  - use built-in method Object.entries(obj)
    - returns array of key/value pairs for an object

```
for (let [key, value] of Object.entries(object1))
  {  console.log(`${key}: ${value}`);}
```

Main Point : Destructuring assignments

Destructuring assignment is a new (ES6) and now widely used convenience syntax that allows us to “unpack” arrays or objects into a set of variables.
Science of Consciousness: This feature allows developers to more quickly accomplish common coding tasks. Do less and accomplish more.

Main Point Preview: JSON

JSON is a widely used data exchange format used in every modern programming language. It is based on JavaScript object representations. The main uses are to convert JavaScript objects to strings for transmission over a network and to parse JSON strings sent from servers into JavaScript objects.
Science of Consciousness: JSON is a mechanism for improving communication between remote entities—clients and servers. Coherent and orderly consciousness is a mechanism for improving communication between people.

JSON methods, toJSON

- Often must convert objects into strings
  - send it over a network,
  - output it for logging purposes.

```
let user = {
  name: "John",
  age: 30,
  toString:  function() {
    return `{name: "${this.name}", age: ${this.age}}`;
  }
};
alert(user); // {name: "John", age: 30}
```

- …But new properties are added, old properties are renamed and removed.
  - Updating such toString every time can become a pain.

JSON.stringify

- JSON (JavaScript Object Notation) is a general format to represent values and objects.
  - RFC 4627 standard.
  - Initially for JavaScript, but many other languages have libraries to handle it as well.
  - data exchange when client uses JavaScript and the server uses Ruby/PHP/Java/Whatever.

JSON.stringify to convert objects into JSON.
JSON.parse to convert JSON back into an object

```
let student = {
  name: 'John',
  age: 30,
  isAdmin: false,
  courses: ['html', 'css', 'js'],
  wife: null};

let json = JSON.stringify(student);

{  "name": "John",
  "age": 30,
  "isAdmin": false,
  "courses": ["html", "css", "js"],
  "wife": null}
```

- important differences from the object literal:
  - Strings use double quotes.
  - No single quotes or backticks in JSON.
  - Object property names double-quoted
- JSON is data-only
- some object properties are skipped
  - Function properties (methods).
  - Symbolic properties.
  - Properties that store undefined.

JSON.parse

*

- convert JSON back into an object

```
// stringified array
let numbers = "[0, 1, 2, 3]";
numbers = JSON.parse(numbers);
alert( numbers[1] ); // 1

let user = '{ "name": "John", "age": 35, "isAdmin": false, "friends": [0,1,2,3] }';
user = JSON.parse(user);
alert( user.friends[1] ); // 1
```

typical mistakes in hand-written JSON

```
let json = `{
  name: "John",                     // mistake: property name without quotes
  "surname": 'Smith',               // mistake: single quotes in value (must be double)
  'isAdmin': false                  // mistake: single quotes in key (must be double)
  "birthday": new Date(2000, 2, 3), // mistake: no "new" is allowed, only bare values
  "friends": [0,1,2,3]              // here all fine
}`;
```

Exercise

//create and log to the console a json string from john. Then create a new instance of john, johnClone, using JSON.parse. Is john === johnClone?

```
const john = {
    name: "John",
    surname: "Smith",
    isAdmin: false,
    birthday: {"year": 2000, "month": "February", "day": 3},
    friends: [0,1,2,3]
  };
```

Main Point : JSON

JSON is a widely used data exchange format used in every modern programming language. It is based on JavaScript object representations. The main uses are to convert JavaScripts to strings for transmission over a network and to parse JSON strings sent from servers into JavaScript objects. *Science of Consciousness:* JSON is a mechanism for improving communication between remote entities—clients and servers. Coherent and orderly consciousness is a mechanism for improving communication between people.

CONNECTING THE PARTS OF KNOWLEDGE WITH THE WHOLENESS OF KNOWLEDGE

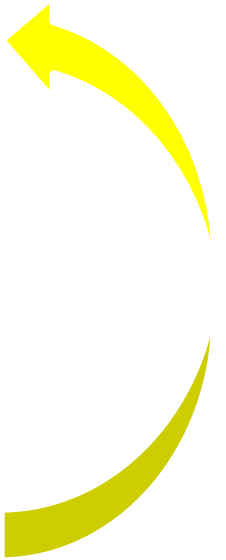
The Nature of Life Is to Grow

1. Destructuring assignments and JSON are new features and data representations in JavaScript.

2. Destructuring provides syntactic convenience and programming efficiency and JSON enables remote transfer of data.

---

3. **Transcendental consciousness**. Is the simplest state of awareness and home of all the laws of nature.
4. **Impulses within the transcendental field:** Thoughts at this level will be maximally life supporting.
5. **Wholeness moving within itself:** In unity consciousness one experiences the goal of all growth and evolution in daily life.



JavaScript Object Notation (JSON)

JSON is a syntax for storing and exchanging data and an efficient alternative to XML

```
{"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]}
```

A name/value pair consists of a field name **(in double quotes)**, followed by a colon, followed by a value.

JSON values can be:

- A number (integer or floating point)
- A string (in double quotes)
- A Boolean (true or false)
- An array (in square brackets)
- An object (in curly braces)
- null

JSON expressions exercise

Given the JSON data at right, what expressions would produce:

- The window's title?
- The image's third coordinate?
- The number of messages?
- The y-offset of the last message?

```
const jsonString = '{
  "window": {
    "title": "Sample Widget",
    "width": 500,
    "height": 500
  },
  "image": {
    "src": "images/logo.png",
    "coords": [250, 150, 350, 400],
    "alignment": "center"
  },
  "messages": [
    {"text": "Save", "offset": [10, 30]}
    {"text": "Help", "offset": [ 0, 50]},
    {"text": "Quit", "offset": [30, 10]},
  ],
  "debug": "true"
}';
const data = JSON.parse(jsonString);
```