OBJECT LITERALS

TOPICS IN JAVASCRIPT

Prof. Andrew Sheehan

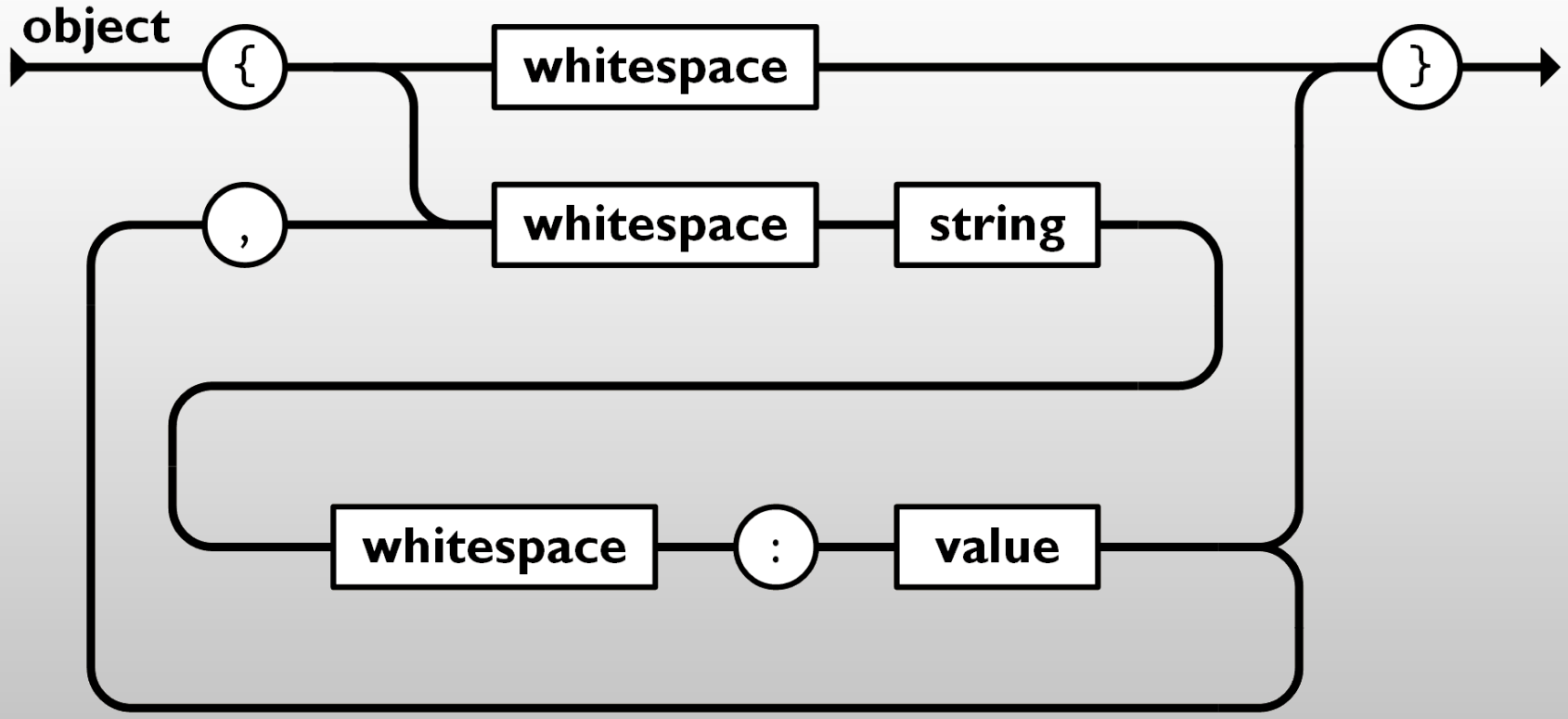
Boston University/MET
Computer Science Department

WHAT ARE THEY

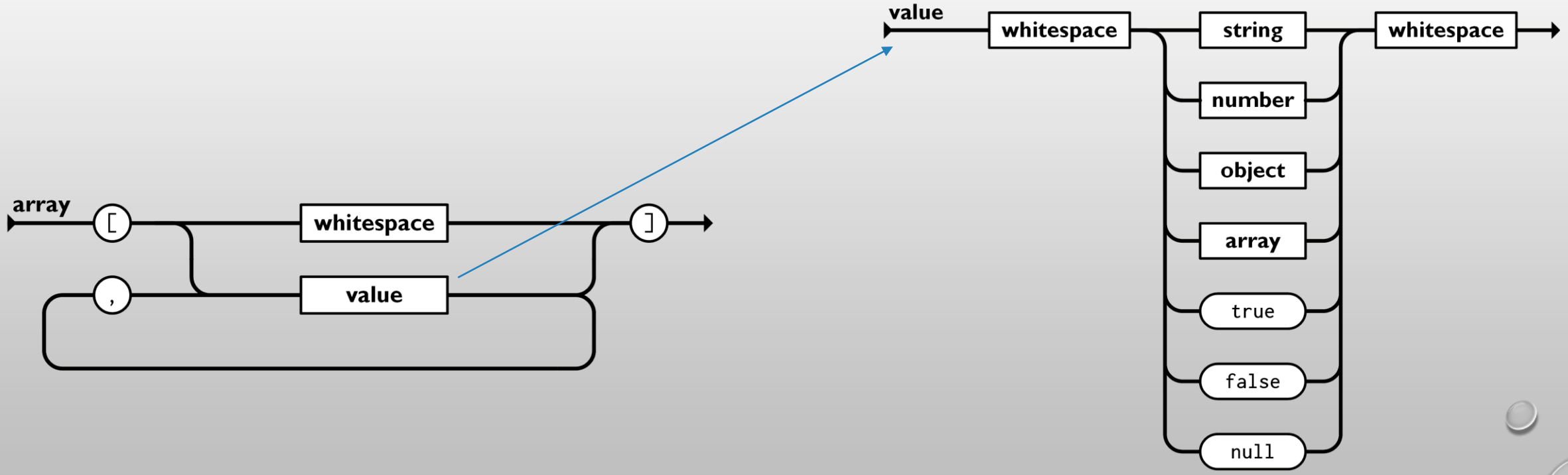
(A) An object literal is a comma-separated list of name:value pairs inside of curly brace.

The values can be arrays, standard data types as well as other objects (Remember: functions are objects…)

STANDARD FORM



ARRAY USE


array
[
whitespace
]
,
value
value
whitespace
string
number
object
array
true
false
null
whitespace

```
let account = {
    customer: "Default (Unknown)",
    id: 0
}
```

EXAMPLE

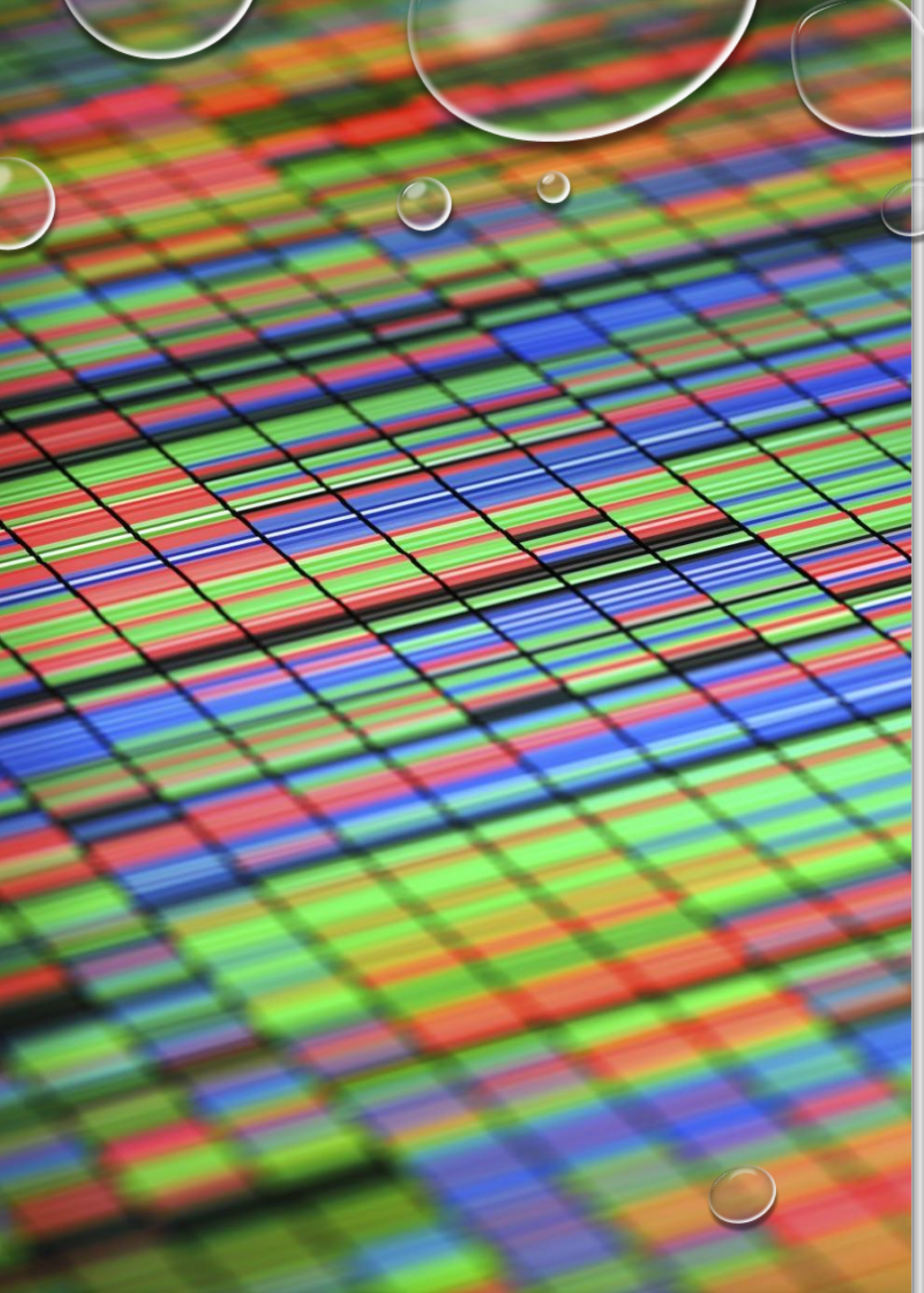
1. Must use surrounding/enclosing brackets.
2. name : standard data type
3. use a comma between pairs


```
let account = {
    client: undefined,
    id: 0
};

let account = { client: undefined, id: 0 };
```

EMPTY LITERAL IS ACCEPTABLE

```
let account = { };
```



KEYS WITH SPACE

```
let account = {
    client: undefined,
    'my key': 0
};
```


KEYS WITH DASHES?
(YES, IT'S FINE)

```
let book = {
    isbn-10 : '01234567890',
    isbn-13 : '0123456789012',
    author : {
        fullname : 'Not Real',
        rating : 5
    }
}
```

EMBEDDED
DOCUMENTS

WITH FUNCTIONS

```
let book = {
    isbn-10 : ‘01234567890’,
    isbn-13 : ‘0123456789012’3,
    author : {
        fullname : ‘Not Real’,
        rating : 5
    },
    getName: () => {
        return this.fullname;
    }
}
```

```
let book = {
    1 : 'one',
    2 : 'two',
    3 : 'three'
}
```

```
let book = {
    '1' : 'one',
    '2' : 'two',
    '3' : 'three'
}
```

NUM8ERS AS KEYS

Numbers can be used as object literal keys, but they will automatically be converted to strings

NEW OBJECT()

VS

{ }

There is no difference

```
let book = {
   isbn-10 : '01234567890',
   isbn-13 : '0123456789012',
   author : {
      fullname : 'Not Real',
      rating : 5
   }
   getName: () => {
        return this.author.fullname;
   }
}
```

```
let book = new Object();

book.isbn-10 = '01234567890';
book.isbn-13 ='0123456789012';
book.author.fullname = 'Not Real';
book.author.rating = 5;
book.getName = () => {
  return this.author.fullname;
}
```

GETTER & SETTER

```
let course = {
    section : ‘A1’,
    get abbreviation() {
      return this.section;
    },
    set abbreviation(abbrev) {
      this.section = abbrev;
    }
}
```

```
console.log(course.abbreviation);
```

LITERAL USE EXAMPLE

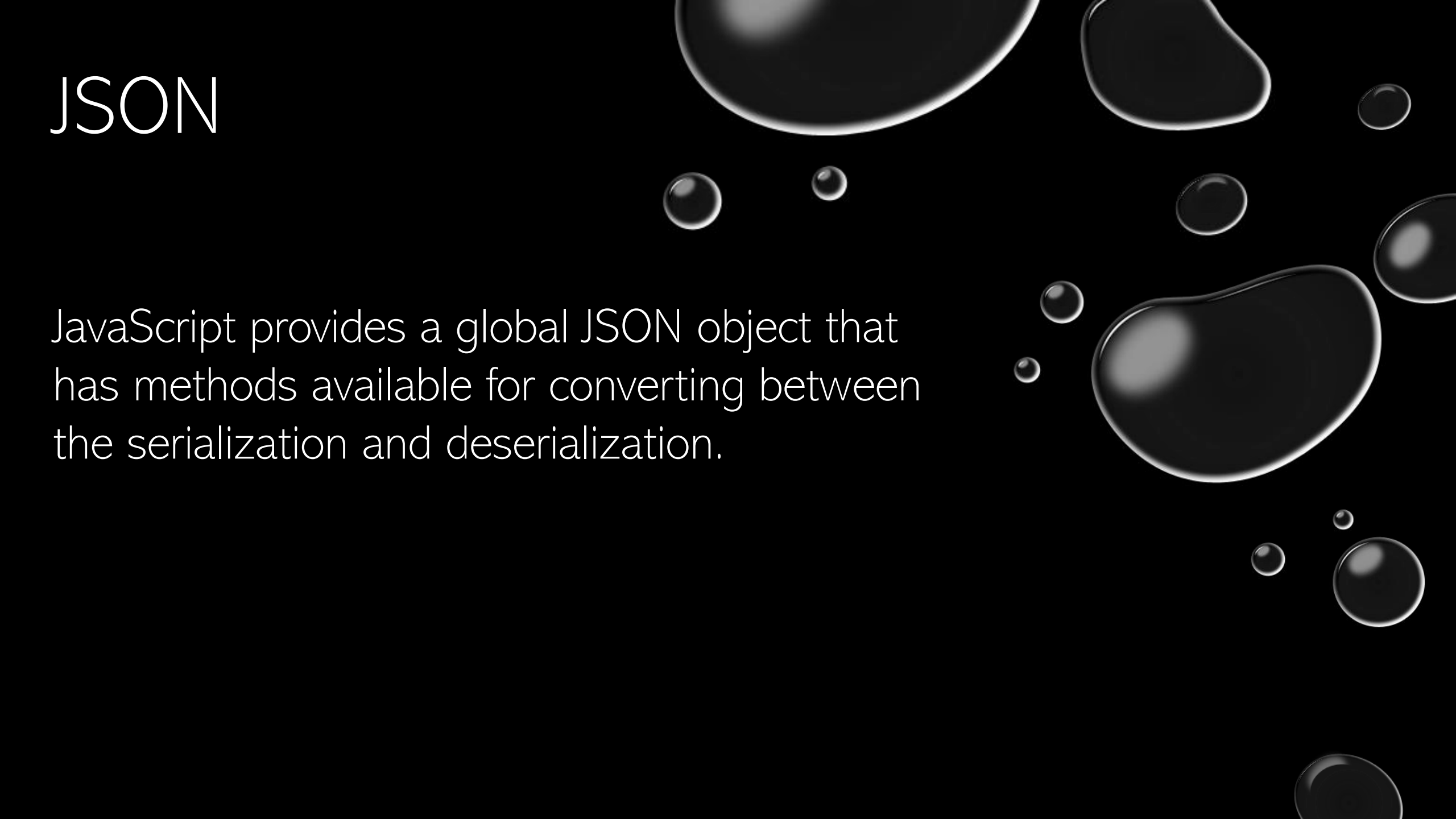
```
function getMonthName (monthNo) {
    let names = {
      "1": "January",
      "2": "February",
      "3": "March",
      "4": "April",
      "5": "May",
      "6": "June",
      "7": "July",
      "8": "August",
      "9": "September",
      "10": "October",
      "11": "November",
      "12": "December"
    };
    return names[monthNo] || "Invalid";
  }

  let monthName = getMonthName(1);
  console.log(monthName);    //January

  monthName = getMonthName(13);
  console.log(monthName);   //Invalid
```

JSON

JavaScript provides a global JSON object that has methods available for converting between the serialization and deserialization.



DESERIALIZATION/ SERIALIZATION

Converting a string to a native object is called deserialization

Converting a native object to a string is called serialization

JSON.PARSE()

JavaScript Demo: JSON.parse()

```
1 const json = '{"result":true, "count":42}';
2 const obj = JSON.parse(json);
3
4 console.log(obj.count);
5 // Expected output: 42
6
7 console.log(obj.result);
8 // Expected output: true
9
```

JSON.STRINGIFY()

JavaScript Demo: JSON.stringify()

```
1  console.log(JSON.stringify({ x: 5, y: 6 }));
2  // Expected output: '{"x":5,"y":6}'
3
4  console.log(JSON.stringify([new Number(3), new String('false'), new Boolean(false)]));
5  // Expected output: '[3,"false",false]'
6
7  console.log(JSON.stringify({ x: [10, undefined, function () {}, Symbol('')] }));
8  // Expected output: '{"x":[10,null,null,null]}'
9
10 console.log(JSON.stringify(new Date(2006, 0, 2, 15, 4, 5)));
11 // Expected output: '"2006-01-02T15:04:05.000Z"'
12
```