

ES6 top-up

(See archive for code samples)

ES6/2015

- Modularity.
- Destructuring.
- The spread operator.
- Default arguments

Modularity

- Split **application code** into multiple files, termed modules.
- Reusability - make **modules available to other modules**..
- **Pre-ES6 provided module system via separate library;**
- **ES6 modules built into language.**
 - **Two options: Default exports; Named exports (also Mixed exports)**

```
29 // lib.js
30 export default function() {
31   ...
32 }
33 // -----
34 // bar.js
35 import myFunc from './lib';
36 myFunc();
37
```

```
// lib.js (Named exports)
export let myStr = 'important string';
export const pi = 3.14159526;
export function myFunc() {
  ...
}
export class myClass {
  ...
}
// -----
// bar.js
import {myFunc, myStr} from './lib';
myFunc() ;
console.log(myStr)
```

Destructuring

- **Assigning the properties of an array or object to variables using a declarative style rather than an imperative/procedural style..**

Instead of:

```
let nums = [10, 11, 12]  
let v1 = nums[0]  
let v2 = nums[1]  
let v3 = nums[2]
```

Use:

```
let [v1, v2, v3] = nums
```

Instead of:

```
let obj = { alpha:100,  
            beta: 'enterprise'}  
let alpha = obj.alpha  
let beta = obj.beta
```

Use :

```
let {alpha, beta} = obj;
```

- **See 01_*_destructuring.js**

The Spread operator (...)

- **Allows an iterable to expand in places where 0+ arguments are expected.**
 - **Iterable – Arrays, Objects.**
- **`...ArrayRef; ...objectRef`**
- **See `02*_spread.js`**

Default arguments

```
function add(x = 1, y = 2) {  
    return x + y;  
}  
  
console.log(add(5));           // 7  
console.log(add(undefined, 1)); // 2  
console.log(add());            // 3
```

JavaScript Object Notation

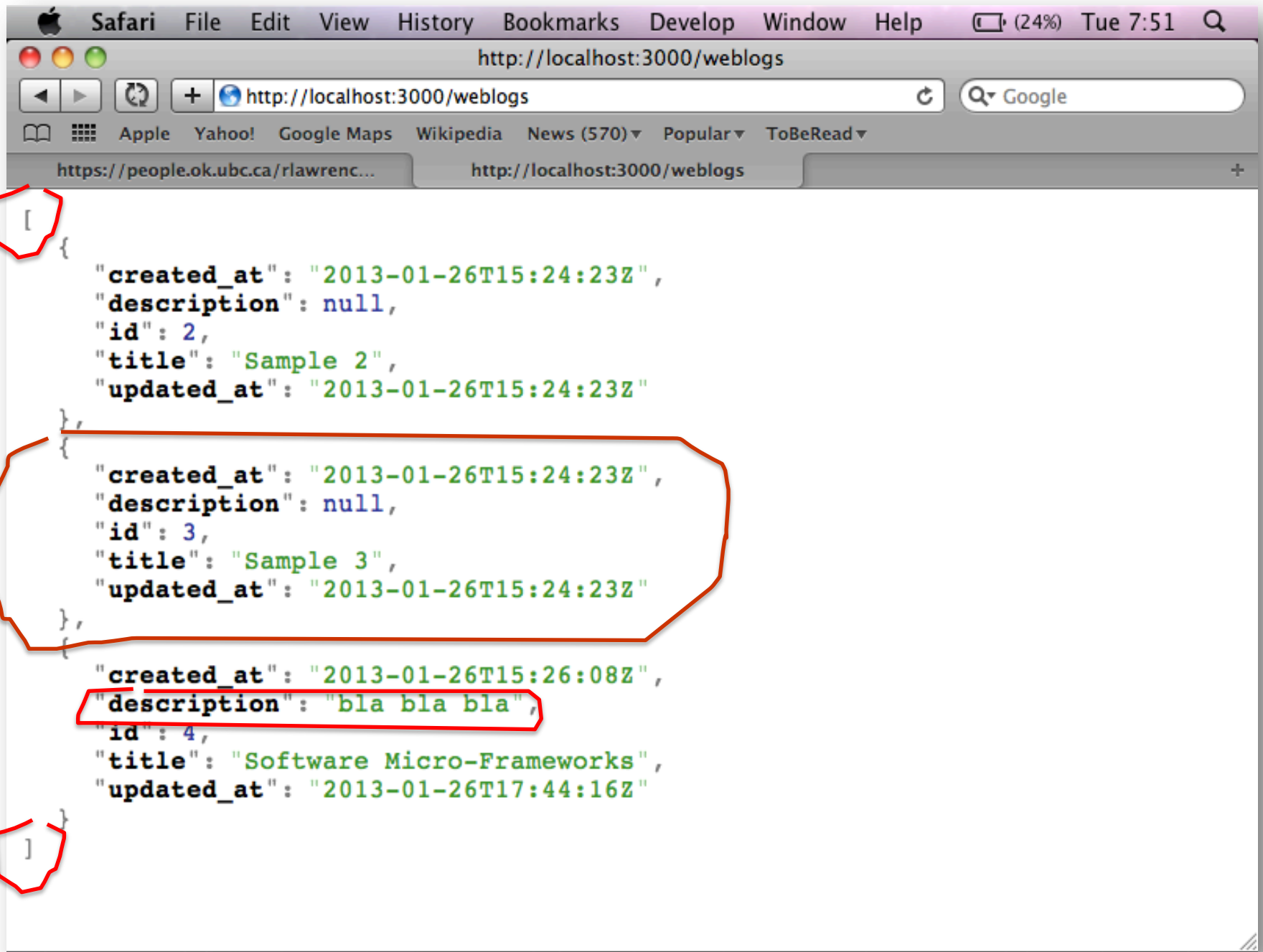
JSON

JSON

- **A standard for serializing data into text form.**
 - **Alternative to XML serialization.**
- **Advantages:**
 - 1. Human-readable (like XML, but easier).**
 - 2. Useful for data interchange between applications (like XML, but less verbose).**
 - 3. Useful for representing and storing semi-structured data.**
 - **Unlike the Relational data model, which is only suited for structured data.**
- **JSON is no longer tied to JavaScript – lots of languages have JSON parsers.**

JSON

- **JSON constructs:**
 - 1. Base Values:**
 - number, strings (double quoted), boolean (true / false), null.
 - 2. Composite values:**
 - a. Objects:** enclosed in { } and consist of set of key-value pairs.
 - Keys must be double-quoted strings ****
 - b. Arrays:** enclosed in [] and are lists of values.
 - Objects and arrays can be nested.



A Property.

```
[ "Books":  
  [  
    { "ISBN":"ISBN-0-13-713526-2",  
      "Price":85,  
      "Edition":3,  
      "Title":"A First Course in Database Systems",  
      "Authors":[ { "First_Name":"Jeffrey", "Last_Name":"Ullman"},  
                   { "First_Name":"Jennifer", "Last_Name":"Widom"} ] }  
    ,  
    { "ISBN":"ISBN-0-13-815504-6",  
      "Price":100,  
      "Remark":"Buy this book bundled with 'A First Course' - a great deal!",  
      "Title":"Database Systems:The Complete Book",  
      "Authors":[ { "First_Name":"Hector", "Last_Name":"Garcia-Molina"},  
                   { "First_Name":"Jeffrey", "Last_Name":"Ullman"},  
                   { "First_Name":"Jennifer", "Last_Name":"Widom"} ] }  
  ],  
  "Magazines":  
  [  
    { "Title":"National Geographic",  
      "Month":"January",  
      "Year":2009 }  
    ,  
    { "Title":"Newsweek",  
      "Month":"February",  
      "Year":2009 }  
  ]  
]
```

Semi-structured

Relational model Vs JSON model

	JSON	Relational
Structure	Nested objects + arrays	Tables
Schema	Variable (and not required)	Fixed
Queries	Limited	SQL, RA
Ordering	Arrays are sorted	No
Systems	Used with programming languages and some NoSQL systems	Many commercial and open source systems

XML Vs JSON.

XML versus JSON

JSON Introduction

	XML	JSON
Verbosity	More	Less
Complexity	More	Less
Validity	DTDs widely used XSDs	JSON Schema not widely used
Prog. Interface	Clunky "Impedance mismatch"	More direct
Querying	XPath - XQuery XSLT -	JSON Path JSON Query

