

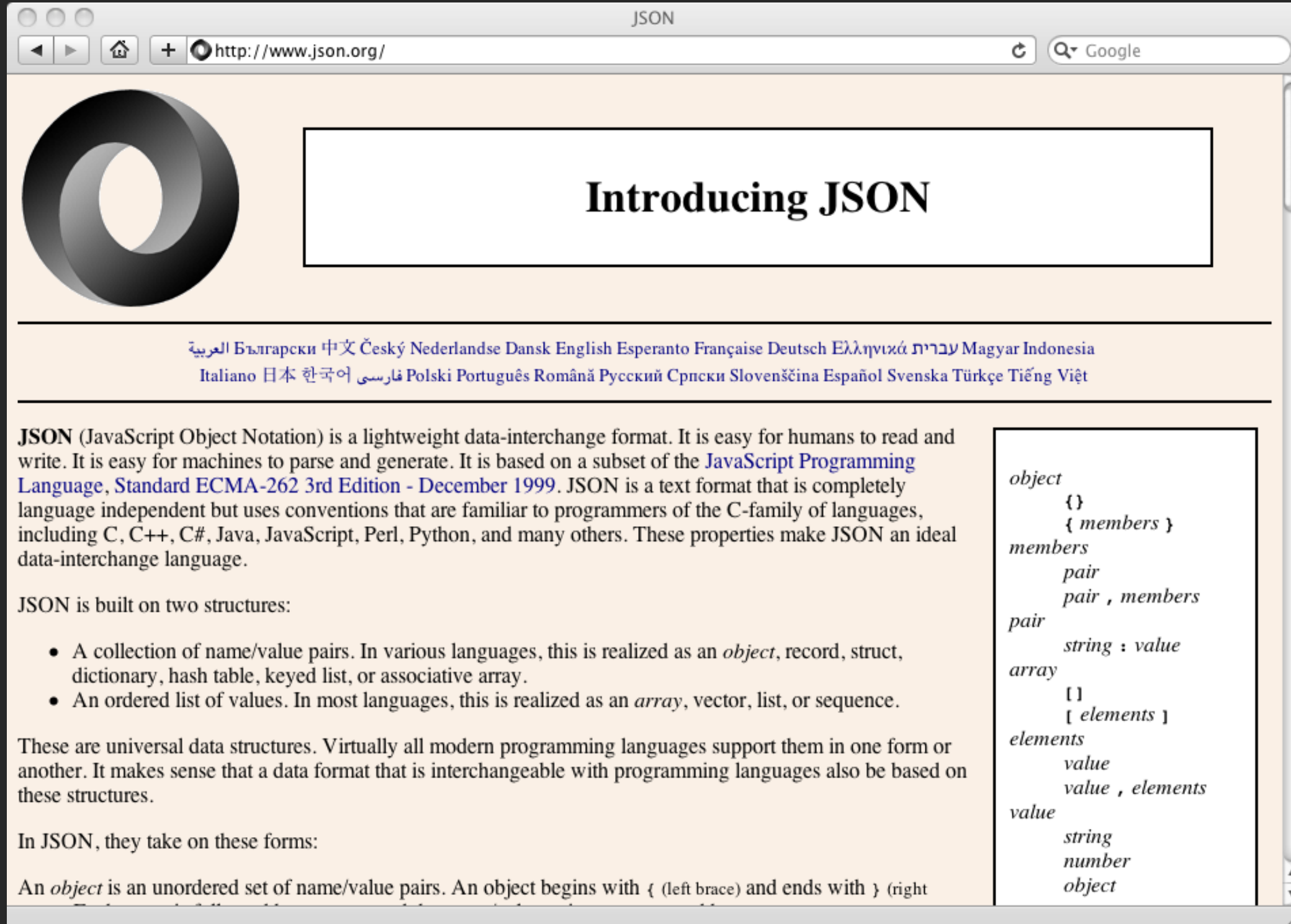
JavaScript Object Notation

- Douglas Crockford - “Discovered” JSON
- Object literal notation in JavaScript



<http://www.youtube.com/watch?v=kc8BAR7SHJI>





The screenshot shows the JSON.org website in a web browser. The browser's address bar displays 'http://www.json.org/'. The page features a large black circular logo on the left and a white box with the title 'Introducing JSON' on the right. Below the title, there is a horizontal list of language links. The main content area on the left describes JSON as a lightweight data-interchange format, while the right side contains a diagram illustrating the hierarchical structure of JSON objects, arrays, pairs, and values.

JSON

http://www.json.org/ Google

Introducing JSON

العربية Български 中文 Český Nederlandse Dansk English Esperanto Française Deutsch Ελληνικά עברית Magyar Indonesia Italiano 日本 한국어 فارسی Polski Português Română Русский Српски Slovenščina Español Svenska Türkçe Tiếng Việt

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the [JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999](#). JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an *object*, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an *array*, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures.

In JSON, they take on these forms:

An *object* is an unordered set of name/value pairs. An object begins with { (left brace) and ends with } (right

object

```
{ }
{ members }
```

members

```
pair
pair , members
```

pair

```
string : value
```

array

```
[ ]
[ elements ]
```

elements

```
value
value , elements
```

value

```
string
number
object
```



```
import json
data = '''{
    "name" : "Chuck",
    "phone" : {
        "type" : "intl",
        "number" : "+1 734 303 4456"
    },
    "email" : {
        "hide" : "yes"
    }
}'''

info = json.loads(data)
print('Name:', info["name"])
print('Hide:', info["email"]["hide"])
```

json1.py

JSON represents data
as nested “lists” and
“dictionaries”

```
import json
input = '''[
    { "id" : "001",
      "x" : "2",
      "name" : "Chuck"
    } ,
    { "id" : "009",
      "x" : "7",
      "name" : "Chuck"
    }
]'''

info = json.loads(input)
print('User count:', len(info))
for item in info:
    print('Name', item['name'])
    print('Id', item['id'])
    print('Attribute', item['x'])
```

json2.py

JSON represents data
as nested “lists” and
“dictionaries”

Service Oriented Approach

http://en.wikipedia.org/wiki/Service-oriented_architecture



Acknowledgements / Contributions



These slides are Copyright 2010- Charles R. Severance (www.dr-chuck.com) of the University of Michigan School of Information and open.umich.edu and made available under a Creative Commons Attribution 4.0 License. Please maintain this last slide in all copies of the document to comply with the attribution requirements of the license. If you make a change, feel free to add your name and organization to the list of contributors on this page as you republish the materials.

Initial Development: Charles Severance, University of Michigan School of Information

... Insert new Contributors here