Web Mapping: Getting started with JSON data

Bailey A. Hanson
Iowa State University, bahanson@iastate.edu

Christopher J. Seeger
Iowa State University, cjsseeger@iastate.edu

Follow this and additional works at: http://lib.dr.iastate.edu/extension_pubs

Part of the Architectural Technology Commons, Landscape Architecture Commons, Other Architecture Commons, and the Urban, Community and Regional Planning Commons

Recommended Citation
http://lib.dr.iastate.edu/extension_pubs/148

Iowa State University Extension and Outreach publications in the Iowa State University Digital Repository are made available for historical purposes only. Users are hereby notified that the content may be inaccurate, out of date, incomplete and/or may not meet the needs and requirements of the user. Users should make their own assessment of the information and whether it is suitable for their intended purpose. For current publications and information from Iowa State University Extension and Outreach, please visit http://www.extension.iastate.edu.
Web Mapping: Getting started with JSON data

Welcome to the Essential Web Mapping and Data Visualization Task Sheet Series. This series supplements the Iowa State University Geospatial Technology Training Program short course series. The task sheets are designed to provide quick, easy instructions for performing specific GIS, mapping and data visualization tasks.

An alternative to XML, JavaScript Object Notation or JSON is a popular data-interchange that is now commonly used by web-mapping tools and web-based APIs. Based on a subset of the JavaScript Programming Language, JSON is an open standard format that has become the data workhorse for transferring data between server and web applications and is used by a variety of programming languages. This tasksheet will provide a quick overview of the JSON syntax.

1. Objects and how they relate to JSON

JavaScript includes several data structures that can be used to store data. Variables can be used to store single text, number or boolean values. Arrays store multiple values in a single variable as a list of values. Arrays can be multidimensional, but more complex data structures will often use objects.

Objects can be thought of as a customized structure that is composed of indices (aka name or index) and values. Whereas arrays are indicated with square brackets [], objects use curly brackets {} to indicate an object. A colon : separates each index and value and a comma separates each index/value pair.

Imagine you are planning an office picnic for employees and their families. You have information about what each person will bring and how many are attending. If stored as an object you could have a single object for each employee.

   a. **Copy Single Objects** code and go to [http://jsfiddle.net](http://jsfiddle.net)
      jsFiddle is a website where you can test out code.

   b. **Paste** the code into the JavaScript pane.

   c. Below the code **add alert(Fred.food)** Press Run. An alert box with the word “chicken” will display.

Dot notation is used to access items contained within an object so **alert(Susan.drink)** would display “soda”. A list of employee objects can become cumbersome. Fortunately, objects can be arrays of objects, objects of arrays, or objects of objects.

   a. **Copy Array of Objects** code to [http://jsfiddle.net](http://jsfiddle.net)

   b. **Add alert(employees[0].drink)** Press Run. An alert box with the word “beer” will display.

JSON works much like the examples presented with one main difference: indices are string values. This is accomplished by surrounding them with double quotation marks " ".

---

**Single Objects**
```
var Fred = {
  food: "chicken",
  drink: "beer",
  attending: 2
};
var Susan = {
  food: "salad",
  drink: "soda",
  attending: 3
};
```

**Array of Objects**
```
var employees = [
  {
    name: "Fred",
    food: "chicken",
    drink: "beer",
    attending: 2
  },
  {
    name: "Susan",
    food: "salad",
    drink: "soda",
    attending: 3
  }
];
```

**JSON**
```
var employees = [
  {
    "name": "Fred",
    "food": "chicken",
    "drink": "beer",
    "attending": 2
  },
  {
    "name": "Susan",
    "food": "salad",
    "drink": "soda",
    "attending": 3
  }
];
```

Arrays are indexed with the first element at location [0]. employees[1].food will return Susan's salad since Susan is at array index 1.
2. Why Use JSON

1. Easy for humans to read and write.
2. Lightweight and easy for machines to generate.
3. Faster and easier to parse with JavaScript than XML.
4. Strict standardized structure works well with AJAX.

3. JSON Validation Tools

While JSON is easy to write, it is also easy to miss a " or a ,, as you code. Several online editors are available to test the JSON and identify the problems.

- JSONLint is a great place to start http://jsonlint.com/
  Just copy and paste your code and click Validate. If your code starts with var = employees make sure you remove it first otherwise you will get an error as shown in the example.

- JSON Formatter and Validator identifies errors and cleans up the tab spacing of the JSON. It also allows you to enter the URL of a JSON file. http://jsonformatter.curiousconcept.com/

- JSON Editor Online http://www.jsoneditoronline.org/ displays data side by side in a clear, editable treeview and in a code editor.

4. Other Types of JSON

Several formalized syntax of JSON objects have been created for storing different types of optimized data. Three of these of interest to web-mapping are:

- **GeoJSON** - Optimized for storing geodata such as points as longitude/latitude coordinates (yes, the order is Long/Lat) and shapes (lines and polygons) and other spatial features.

- **TopoJSON** - Improves upon GeoJSON by processing GIS data with geospatial topology.

- **ESRI JSON** - ESRI uses a syntax that is similar to GeoJSON, but the Geometry Collections are different and have to be converted between the two formats.

More information about each JSON type will be presented in a future tasksheet.

Contact:

Bailey Hanson  bahanson@iastate.edu, 515-520-1436 or Professor Christopher J. Seeger, ASLA, GISP cjseeger@iastate.edu, 515-509-0651 for more information about the Geospatial Technology Program. This task sheet and more are available at www.extension.iastate.edu/communities/gis