JSON Exercise 2: Document JSON files

In this exercise, you will create documentation for JSON files that provide weather forecast data. We'll make the JSON a little more realistic than what you had in the last exercise. For this exercise, you'll need a word processor like Microsoft Word, Apple Pages, or Google Drive Document.

Step 1: One day's forecast.

Here's an example JSON response for one day's weather forecast. Note that unlike the files in the previous lecture, there is no "top level" object with one key and one value. There is only one level, so no need for indentation.

```
{
  "date": "2015-09-01",
  "description": "sunny",
  "maxTemp": 22,
  "minTemp": 20,
  "windSpeed": 12,
  "danger": false
}
```

Open up your word processor and create a table with four columns: Element, Description, Type, and Notes. Refer to the previous exercise to figure out what the various elements mean. Also, in the table, be sure to include information from these notes from the developer:

- "description" can have these values: "sunny", "overcast", "partly cloudy", "raining", and "snowing"
- "maxTemp" and "minTemp" are in degrees Celsius
- "windSpeed" is in kilometers per hour

The last thing you need to do is to add one sentence before the table to describe what this does.

Finally, compare your documentation to what I came up with at: http://sdkbridge.com/ud/oneday.pdf

Step 2: Three-day forecast

The next step is to document JSON for a three-day forecast. An example is below:

```
{
  "longitude": 47.60,
  "latitude": 122.33,
  "forecasts": [
    {
      "date": "2015-09-01",
      "description": "sunny",
      "maxTemp": 22,
      "minTemp": 20,
      "windSpeed": 12,
      "danger": false
    },
    {
      "date": "2015-09-02",
      "description": "overcast",
      "maxTemp": 21,
      "minTemp": 17,
      "windSpeed": 15,
      "danger": false
    },
    {
      "date": "2015-09-03",
      "description": "raining",
      "maxTemp": 20,
      "minTemp": 18,
      "windSpeed": 13,
      "danger": false
    }
  ]
}
```

Create a new table to document this JSON response. Use the indentation method, and then copy values from the table in Step 1.

Don't just say "longitude" and "latitude" in the description for longitude and latitude. Your job as an API writer is to figure out why there would be a longitude and latitude in a weather forecast, and then to write a description that makes it clear.

Add a one sentence description before the table. Note that there is nothing about the data structure that says that it holds exactly three days of forecast, because an array can have any number of items. This means that the JSON data can hold any number of days' worth of data. Make sure your description says that.

Finally, compare it to mine:

http://sdkbridge.com/ud/threeday.pdf

Step 3: Meeting Request

Let's imagine we have an API for an online calendar. We send JSON as a request to create a new meeting. The JSON looks like this:

```
{
  "meeting" : {
     "time": "2015-09-01 10:00",
     "duration": 60,
     "description": "2016 Strategic Planning Meeting",
     "location": "Building 23, Room 206",
     "reminder": 15,
     "invitees": ["michael@example.com", "thelma@example.com",
            "david@example.com", "leon@example.com"]
  }
}
```

Notes:

- `time` is GMT.
- `location` is optional. The default is an empty string.
- `reminder` is optional. The default is 10 minutes.
- `invitees` is optional. The default is an empty array.

By the way, an empty string looks like this: `""`, and an empty array looks like this: `[]`. (Not that you have to put that in the documentation.)

Create a documentation table for this request. Add a one sentence description before the table. When you are done, check your answers against mine at: http://sdkbridge.com/ud/meeting.pdf.