

- 1) Create an Input using the provided 'Receive Features on a REST endpoint' input connector template
 - a. I'd recommend configuring the input connector to create a GeoEvent Definition for you:

The screenshot shows the 'ArcGIS GeoEvent Processor Manager' interface. The 'Inputs' tab is selected. The main heading is 'Creating Input - Receive Features on a REST endpoint'. There are 'Save' and 'Cancel' buttons in the top right. The configuration fields are as follows:

- Name*:
- Create GeoEvent Definition*: Yes No
- GeoEvent Definition Name (New)*:

An 'Advanced' section is expanded, showing:

- Expected Date Format:
- Acceptable MIME Types (Server Mode):

- 2) Create an Output using the provided 'Write to a .json file' output connector template
 - a. You'll need to register a system folder with GEP as a Data Store; I registered C:\arcgisserver\GeoEvent_Files as 'GeoEvent_Files' and have my connectors write to an 'output' sub-folder (see screenshot below)

The screenshot shows the 'ArcGIS GeoEvent Processor Manager' interface. The 'Outputs' tab is selected. The main heading is 'file-json-out (Write to a .json file)'. There are 'Save' and 'Cancel' buttons in the top right. The configuration fields are as follows:

- Name*:
- Folder*:
- Filename Prefix:

An 'Advanced' section is expanded, showing:

- Subfolder:
- File Extension:
- File Rollover Method:
- File Rollover Frequency:
- File Cleanup Age (Minutes):
- File Cleanup Method:
- Formatted JSON*: Yes No

- 3) Design and publish the GeoEvent Service which uses the input and output connectors to receive the feature JSON and write the created GeoEvents out to a system file as a *.json text file.



- 4) Obtain some valid feature JSON for posting.
a. An easy way to do this is to query your local ArcGIS for Server sample service ...

ArcGIS REST Services Directory [Login](#) | [Get Token](#)
[Home](#) > [services](#) > [SampleWorldCities \(MapServer\)](#) > [Cities](#) [Help](#) | [API Reference](#)

[JSON](#)

Layer: Cities (ID: 0)

Name: Cities

Display Field: CITY_NAME

Type: Feature Layer

Geometry Type: esriGeometryPoint

Fields:

- OBJECTID (type: esriFieldTypeOID , alias: OBJECTID)
- Shape (type: esriFieldTypeGeometry , alias: Shape)
- CITY_NAME (type: esriFieldTypeString , alias: CITY_NAME , length: 30)
- POP (type: esriFieldTypeInteger , alias: POP)
- POP_RANK (type: esriFieldTypeInteger , alias: POP_RANK)
- POP_CLASS (type: esriFieldTypeString , alias: POP_CLASS , length: 25)
- LABEL_FLAG (type: esriFieldTypeInteger , alias: LABEL_FLAG)

Supported Operations: [Query](#) [Generate Renderer](#) [Return Updates](#)

Note: You can copy the entire block of feature JSON, or you can locate and copy just one of the features:

```
{
  "displayFieldName": "CITY_NAME",
  "fieldAliases": {
    "OBJECTID": "OBJECTID",
    "CITY_NAME": "CITY_NAME",
    "POP": "POP",
    "POP_RANK": "POP_RANK",
    "POP_CLASS": "POP_CLASS",
    "LABEL_FLAG": "LABEL_FLAG"
  },
  "geometryType": "esriGeometryPoint",
  "spatialReference": {
    "wkid": 4326,
    "latestWkid": 4326
  },
  "fields": [
    {
      "name": "OBJECTID",
      "type": "esriFieldTypeOID",
      "alias": "OBJECTID"
    },
    {
      "name": "CITY_NAME",
      "type": "esriFieldTypeString",
      "alias": "CITY_NAME",
      "length": 30
    },
    {
      "name": "POP",
      "type": "esriFieldTypeInteger",
      "alias": "POP"
    },
    {
      "name": "POP_RANK",
      "type": "esriFieldTypeInteger",
      "alias": "POP_RANK"
    },
    {
      "name": "POP_CLASS",
      "type": "esriFieldTypeString",
      "alias": "POP_CLASS",
      "length": 100
    },
    {
      "name": "LABEL_FLAG",
      "type": "esriFieldTypeInteger",
      "alias": "LABEL_FLAG"
    }
  ],
  "features": [
    {
      "attributes": {
        "OBJECTID": 1,
        "CITY_NAME": "Duiaba",
        "POP": 521934,
        "POP_RANK": 3,
        "POP_CLASS": "500,000 to 999,999",
        "LABEL_FLAG": 0
      },
      "geometry": {
        "x": -56.093017578125,
        "y": -15.614990234375
      }
    },
    {
      "attributes": {
        "OBJECTID": 2,
        "CITY_NAME": "Brasilia",
        "POP": 2207718,
        "POP_RANK": 2,
        "POP_CLASS": "1,000,000 to 4,999,999",
        "LABEL_FLAG": 0
      },
      "geometry": {
```

- 5) Download an HTML Poster application for your preferred browser
 - a. <http://code.google.com/p/chrome-poster> is one such app for Chrome
- 6) Configure the poster to identify the MIME type in its header
 - a. 'Content-Type' / 'application/json' (see screenshot below)
- 7) Post the JSON block to the GeoEvent Processor's REST endpoint
 - a. <http://localhost:6180/geoevent/rest/receiver/<name-of-the-input>> (e.g. rest-features-in)

URL:

Headers:

Name: Value:

value '##' to delete

name	value
Content-Type	application/json

Content Body:

```
{
  "displayFieldName": "CITY_NAME",
  "fieldAliases": {
    "OBJECTID": "OBJECTID",
    "CITY_NAME": "CITY_NAME",
    "POP": "POP",
    "POP_RANK": "POP_RANK",
    "POP_CLASS": "POP_CLASS",
    "LABEL_FLAG": "LABEL_FLAG"
  },
  "geometryType": "esriGeometryPoint",
  "spatialReference": {
    "wkid": 4326,
    "latestWkid": 4326
  }
}
```

Response:

status: 200 OK

- Access-Control-Allow-Origin: *
- Date: Fri, 18 Oct 2013 18:48:08 GMT
- Access-Control-Allow-Credentials: true
- Server: Jetty(7.5.4.v20111024)
- Content-Length: 0
- Content-Type: plain/text

8) Locate your output *.json file in your registered system folder.

You should be able to format the JSON as a list by adding an opening '[' and closing ']' set of brackets around the text in the file ... and replacing every '{' with ',' so that the individual JSON blocks are comma separated. Here's what I got, after formatting in JSON Lint, for a query of the World Cities whose OBJECTID is less-or-equal to 10.

```
[
  {
    "CITY_NAME": "Brasilia",
    "POP": 2207718,
    "POP_RANK": 2,
    "POP_CLASS": "1,000,000 to 4,999,999",
    "LABEL_FLAG": 0,
    "shape": {
      "x": -47.897705078125,
      "y": -15.7921142578125,
      "z": 0,
      "spatialReference": {
        "wkid": 4326
      }
    }
  }
]
```

```
},
{
  "CITY_NAME": "Goiania",
  "POP": 1171195,
  "POP_RANK": 2,
  "POP_CLASS": "1,000,000 to 4,999,999",
  "LABEL_FLAG": 0,
  "shape": {
    "x": -49.2550048828125,
    "y": -16.72698974609375,
    "z": 0,
    "spatialReference": {
      "wkid": 4326
    }
  }
},
{
  "CITY_NAME": "Cuiaba",
  "POP": 521934,
  "POP_RANK": 3,
  "POP_CLASS": "500,000 to 999,999",
  "LABEL_FLAG": 0,
  "shape": {
    "x": -56.093017578125,
    "y": -15.614990234375,
    "z": 0,
    "spatialReference": {
      "wkid": 4326
    }
  }
},
{
  "CITY_NAME": "Campo Grande",
  "POP": 729151,
  "POP_RANK": 3,
  "POP_CLASS": "500,000 to 999,999",
  "LABEL_FLAG": 0,
  "shape": {
    "x": -54.61590576171875,
    "y": -20.45098876953125,
    "z": 0,
    "spatialReference": {
      "wkid": 4326
    }
  }
},
{
  "CITY_NAME": "Salto del Guaira",
  "POP": 7385,
  "POP_RANK": 7,
  "POP_CLASS": "Less than 50,000",
  "LABEL_FLAG": 0,
  "shape": {
    "x": -54.2833251953125,
    "y": -24.04998779296875,
    "z": 0,
    "spatialReference": {
      "wkid": 4326
    }
  }
}
```

```
    }
  },
  {
    "CITY_NAME": "Encarnacion",
    "POP": 74983,
    "POP_RANK": 6,
    "POP_CLASS": "50,000 to 99,999",
    "LABEL_FLAG": 0,
    "shape": {
      "x": -55.85101318359375,
      "y": -27.37701416015625,
      "z": 0,
      "spatialReference": {
        "wkid": 4326
      }
    }
  },
  {
    "CITY_NAME": "Posadas",
    "POP": 312060,
    "POP_RANK": 4,
    "POP_CLASS": "250,000 to 499,999",
    "LABEL_FLAG": 0,
    "shape": {
      "x": -55.906005859375,
      "y": -27.39801025390625,
      "z": 0,
      "spatialReference": {
        "wkid": 4326
      }
    }
  },
  {
    "CITY_NAME": "Puerto Maldonado",
    "POP": 37543,
    "POP_RANK": 7,
    "POP_CLASS": "Less than 50,000",
    "LABEL_FLAG": 0,
    "shape": {
      "x": -69.1920166015625,
      "y": -12.60302734375,
      "z": 0,
      "spatialReference": {
        "wkid": 4326
      }
    }
  }
]
```