



---

# Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. doi: 10.18637/jss.v000.i00

---

## positional-iss

Akhil Sadam | as97822

Aerospace Department, University of Texas at Austin

---

### Abstract

An containerized Flask webserver designed for querying ISS sightings and positions on February 13, 2022. Midterm project for COE332. R is used to produce documentation.

*Keywords:* positional-iss, Docker, Flask, Python3, R.

---

### 0.1. Implementation / Files

### 0.2. Input Data

- The application queries data from the National Aeronautics and Space Administration (NASA) public website, in particular ISS positional information via the **Public Distribution file** and regional sighting data for the Midwest via the **XMLsightingData\_citiesUSA05** file.

Example input data is available at the above links.

## 1. REST API:

### ENDPOINT: /

- Description: Get homepage HTML
- Parameters:
  - N/A
- Responses:
  - A 200 response will : Return homepage HTML
- Example: `curl -X GET http://0.0.0.0:5026/ -H "accept: application/json"`

### ENDPOINT: /api/doc

- Description: Get API HTML
- Parameters:
  - N/A
- Responses:

- A 200 response will : Return API HTML
- Example: `curl -X GET http://0.0.0.0:5026/api/doc -H "accept: application/json"`

**ENDPOINT: /api/save**

- Description: Get API as rendered string
- Parameters:
  - N/A
- Responses:
  - A 200 response will : Return rendered API as string
- Example: `curl -X GET http://0.0.0.0:5026/api/save -H "accept: application/json"`

**ENDPOINT: /country**

- Description: Get all possible countries.
- Parameters:
  - N/A
- Responses:
  - A 200 response will : Return a list of countries.
- Example: `curl -X GET http://0.0.0.0:5026/country -H "accept: application/json"`  
yields:

```
[
  "United_States"
]
```

**ENDPOINT: /country/{country}**

- Description: Get data for a single country.
- Parameters:
  - **country** : Value (name) of country to be queried. An example would be :  
`United_States`
- Responses:
  - A 200 response will : Return all matching (queried country) sightings as json.
- Example:  
`curl -X GET http://0.0.0.0:5026/country/United_States -H "accept: application/json"`  
yields:

```
[
  {
    "city": "Olathe",
    "country": "United_States",
    "duration_minutes": "6",
    "enters": "10 above SSW",
    "exits": "10 above ENE",
    "max_elevation": "28",
    "region": "Kansas",
    "sighting_date": "Thu Feb 17/06:13 AM",
    "spacecraft": "ISS",
    "utc_date": "Feb 17, 2022",
    "utc_offset": "-6.0",
    "utc_time": "12:13"
  },
  ....
]
```

```

"country": "United_States",
  "duration_minutes": "3",
  "enters": "19 above NNW",
  "exits": "10 above NNE",
  "max_elevation": "19",
  "region": "Massachusetts",
  "sighting_date": "Sat Feb 26/04:56 AM",
  "spacecraft": "ISS",
  "utc_date": "Feb 26, 2022",
  "utc_offset": "-5.0",
  "utc_time": "09:56"
}
]

```

### ENDPOINT: /country/{country}/region

- Description: Get data for all regions of a certain country.
- Parameters:
  - **country** : Value (name) of country to be queried. An example would be :  
United\_States
- Responses:
  - A 200 response will : Return all matching regions for the queried country as json.
- Example:
 

```
curl -X GET http://0.0.0.0:5026/country/United_States/region -H "accept: application/json"
```

 yields:

```

[
  "Kansas",
  "Kentucky",
  "Louisiana",
  "Maine",
  "Mariana_Islands",
  "Maryland",
  "Massachusetts"
]

```

### ENDPOINT: /country/{country}/region/{region}

- Description: Get all data for a specific region of a certain country.
- Parameters:
  - **country** : Value (name) of country to be queried. An example would be :  
United\_States
  - **region** : Value (name) of region to be queried. An example would be : Kansas
- Responses:
  - A 200 response will : Return all matching results for the queried region as json.
- Example:
 

```
curl -X GET http://0.0.0.0:5026/country/United_States/region/Kansas -H "accept: application/json"
```

 yields:

```

[
  {
    "city": "Olathe",
    "country": "United_States",
    "duration_minutes": "6",

```

```

"max_elevation": "28",
  "region": "Kansas",
  "sighting_date": "Thu Feb 17/06:13 AM",
  "spacecraft": "ISS",
  "utc_date": "Feb 17, 2022",
  "utc_offset": "-6.0",
  "utc_time": "12:13"
},
....
{
  "city": "Yates_Center",
  "country": "United_States",
  "duration_minutes": "1",
  "enters": "12 above N",
  "exits": "10 above N",
  "max_elevation": "12",
  "region": "Kansas",
  "sighting_date": "Sat Feb 26/05:29 AM",
  "spacecraft": "ISS",
  "utc_date": "Feb 26, 2022",
  "utc_offset": "-6.0",
  "utc_time": "11:29"
}
]

```

### **ENDPOINT: /country/{country}/region/{region}/city**

- Description: Get all cities for a specific region of a certain country.
- Parameters:
  - **country** : Value (name) of country to be queried. An example would be : **United\_States**
  - **region** : Value (name) of region to be queried. An example would be : **Kansas**
- Responses:
  - A 200 response will : Return all matching cities for the queried region and country as json.
- Example:
 

```
curl -X GET http://0.0.0.0:5026/country/United_States/region/Kansas/city -H "accept: applicat
yields:
```

```

[
  "Olathe",
  "Osborne",
  "Oskaloosa",
  "Oswego",
  "Ottawa",
  "Paola",
  "Phillipsburg",
  "Pittsburg",
  "Pratt",
  "Russell",
  "Saint_Francis",
  "Saint_John",
  "Salina",
  "Scott_City",
  ....

```

```

"Tallgrass_Prairie_National_Preserve",
  "Topeka",
  "Tribune",
  "Troy",
  "Ulysses",
  "WaKeeny",
  "Washington",
  "Wellington",
  "Westmoreland",
  "Wichita",
  "Winfield",
  "Yates_Center"
]

```

### ENDPOINT: /country/{country}/region/{region}/city/{city}

- Description: Get all information for a specific city of a region of a certain country.
- Parameters:
  - country : Value (name) of country to be queried. An example would be : United\_States
  - region : Value (name) of region to be queried. An example would be : Kansas
  - city : Value (name) of city to be queried. An example would be : Wichita
- Responses:
  - A 200 response will : Return all information for the queried city as json.
- Example:
 

```
curl -X GET http://0.0.0.0:5026/country/United_States/region/Kansas/city/Wichita -H "accept: yields:"
```

```

[
  {
    "city": "Wichita",
    "country": "United_States",
    "duration_minutes": "6",
    "enters": "10 above S",
    "exits": "10 above ENE",
    "max_elevation": "25",
    "region": "Kansas",
    "sighting_date": "Thu Feb 17/06:12 AM",
    "spacecraft": "ISS",
    "utc_date": "Feb 17, 2022",
    "utc_offset": "-6.0",
    "utc_time": "12:12"
  },
  ....
  {
    "city": "Wichita",
    "country": "United_States",
    "duration_minutes": "1",
    "enters": "12 above N",
    "exits": "10 above N",
    "max_elevation": "12",
    "region": "Kansas",
    "sighting_date": "Sat Feb 26/05:29 AM",
    "spacecraft": "ISS",
    "utc_date": "Feb 26, 2022",
    "utc_offset": "-6.0",

```

```
"utc_time": "11:29"
  }
]
```

### ENDPOINT: /data

- Description: Updates the list of data dictionaries.
- Parameters:
  - N/A
- Responses:
  - A 201 response will : Updated data dictionary list.
- Example: `curl -X POST http://0.0.0.0:5026/data -H "accept: application/json"`  
yields:

```
"Data updated."
```

### ENDPOINT: /epoch

- Description: Get all possible epochs.
- Parameters:
  - N/A
- Responses:
  - A 200 response will : Return a list of epochs.
- Example: `curl -X GET http://0.0.0.0:5026/epoch -H "accept: application/json"`  
yields:

```
[
  "2022-042T12:00:00.000Z",
  "2022-042T12:04:00.000Z",
  "2022-042T12:08:00.000Z",
  "2022-042T12:12:00.000Z",
  "2022-042T12:16:00.000Z",
  "2022-042T12:20:00.000Z",
  "2022-042T12:24:00.000Z",
  "2022-042T12:28:00.000Z",
  "2022-042T12:32:00.000Z",
  "2022-042T12:36:00.000Z",
  "2022-042T12:40:00.000Z",
  "2022-042T12:44:00.000Z",
  "2022-042T12:48:00.000Z",
  "2022-042T12:52:00.000Z",
  ....
  "2022-057T11:08:56.869Z",
  "2022-057T11:12:56.869Z",
  "2022-057T11:16:56.869Z",
  "2022-057T11:20:56.869Z",
  "2022-057T11:24:56.869Z",
  "2022-057T11:28:56.869Z",
  "2022-057T11:32:56.869Z",
  "2022-057T11:36:56.869Z",
  "2022-057T11:40:56.869Z",
  "2022-057T11:44:56.869Z",
  "2022-057T11:48:56.869Z",
  "2022-057T11:52:56.869Z",
```

```
"2022-057T11:56:56.869Z",
  "2022-057T12:00:00.000Z"
]
```

### ENDPOINT: /epoch/{name}

- Description: Get data for a single epoch.
- Parameters:
  - **name** : Value of epoch to be queried. An example would be : 2022-042T12:04:00.000Z
- Responses:
  - A 200 response will : Return epoch information for first matching epoch as json.
- Example:
 

```
curl -X GET http://0.0.0.0:5026/epoch/2022-042T12:04:00.000Z -H "accept: application/json"
```

 yields:

```
{
  "EPOCH": "2022-042T12:04:00.000Z",
  "X": {
    "#text": "-4483.2181885642003",
    "@units": "km"
  },
  "X_DOT": {
    "#text": "2.63479158884966",
    "@units": "km/s"
  },
  "Y": {
    "#text": "-4839.4374260438099",
    "@units": "km"
  },
  "Y_DOT": {
    "#text": "-4.3774148889971602",
    "@units": "km/s"
  },
  "Z": {
    "#text": "-1653.1850590663901",
    "@units": "km"
  },
  "Z_DOT": {
    "#text": "5.7014974180323597",
    "@units": "km/s"
  }
}
```

### ENDPOINT: /pdf

- Description: Get writeup HTML
- Parameters:
  - N/A

- Responses:

A 200 response will : Return writeup HTML <http://www.jstatsoft.org/>  
 Example: 

```
curl -X GET http://0.0.0.0:5026/pdf -H "accept: application/json"
```

<http://www.jstatsoft.org/>

MMMMMM YYYY, Volume VV, Issue II

doi: 10.18637/jss.v000.i00

Submitted: yyyy-mm-dd

Accepted: yyyy-mm-dd