Version 1.0 Airport Navigation File Format

Publish Date: (TBD – last edit Jan 27 2019)

Intent

This document describes the file format used to define default navigation routes available on www.ifatc.org.

Technology

The file is formatted using the open source hjson format. hjson is based on the industry standard json format with changes that make it far easier to create and edit for human beings.

Tooling

Any text editor can edit hson files. The people that maintain the specification have a free online editor and validator at https://hjson.org/try.html.

Both Visual Studio Code (https://code.visualstudio.com/) and the Atom editors (http://atom.io) are free and support hjson through free add on packages. I have used both editors and both are very good. I use VS Code as my primary programming editor and it does a great job with hjson.

General Concepts

It is not necessary to fully comprehend this section. The description below and examples should make the format clear. These concepts are reviewed to hopefully make the punctuation in the file format seem less arbitrary.

Conceptually there are things or *objects* (airports, routes, nav points, etc.) described in the file, lists of objects and information about objects (properties).

Objects are enclosed in curly braces (“{“ and “}”) and have one or more properties. In this file format and airport, route, nav points are all objects.

Lists are enclosed with square braces (“[“ and “]”).

Properties have a name (e.g. “runways”) and end with a colon (“:”). Properties can be individual values, another object or a list of values / objects. In this file format when a property has an individual value (vs. an object or list) it is one of two types – text or a number.

Text Values

Text values begin after any spaces between the colon at end of the property and the first non space character. They continue to the end of the line. In the example below the value of of the airport property is “KSAN” (without quotes).

```
airport: KSAN
```

A single text value can also span multiple lines to improve the readability of the file. If they span multiple lines the first line after the property name must be three single apostrophes ('). The next line after the lines containing the text values must also be three single apostrophes. Be sure that the apostrophe is a true apostrophe not a left / right single quote (‘). Using the editors mentioned before makes this easier. The two examples below provide effectively the same data.

Example 1

```
   labels: LEJEN GAMBT SHAMU
```

Example 2

```
   labels:
      '''
      LEJEN GAMBT SHAMU BROWS
      HUULK LAX SXC LEJEN LNTRN XMANS
      '''
```

Numeric Values

In version all numbers are round numbers or integers. They do not include a decimal component. Only one number may be provided and they are not enclosed in quotes.

```
   speed: 230
```

Nav points

Nav points can be any fix, VOR, or airport that is valid in Infinite Flight.

Nav points can also be provided as latitude / longitude coordinates in the same format as for an Infinite Flight flight plan. For example KDEN is located at 39.585990, -104.673464. This would be expressed as 3959N/10467W. YSSY is located at -33.969620, 151.177083 which would be expressed as 3397S/15118E.

Comments

Any line that starts with an octothorpe or pound sign (“#”) is treated as a comment and ignored by the site.

```
    routes: [
        {
            direction: Inbound
            # combination of COMIX TWO and RNAV Z Rwy 27 Approach
            # this line and the one above it are ignored
            name: COMIX TWO
```

Indenting

I show lines indented in the description of the file format. While the computer doesn’t care, it helps significantly with readability for humans. Using the text editors mentioned above makes indenting the content easy.

Format Summary

Each file contains the routes for exactly one airport. The name of the file does not matter although the file extension should be ".hjson".

The file starts and ends with curly braces ("{" and "}") on their own line

Objects and Properties

Name	Type	Contents	How Many
airport	Property	Text	One
Options	Property	Text	Zero or one
labels	Property	Text	Zero or one
routes	List	Route object(s)	One
Route	Object		At least one
direction	Property	Text	One
name	Property	Text	One
segments	List	Text	One
labels	Property	Text	One
points	List	Point object(s)	One
Point	Object		At least one if "points" list is present
name	Property	Text	One
altitude_required_max	Property	Number	For both altitude and speed any combination of min, max and recommended may be provided. At least one speed or altitude value must be provided per Point.
altitude_required_min	Property	Number	
altitude_recommended	Property	Number	
speed_required_max	Property	Number	
speed_required_min	Property	Number	
speed_recommended	Property	Number	

Descriptions and Examples

airport

The ICAO code for the airport the file applies to.

Example

```
  airport: KSAN
```

options

A list of options that modifies how the file parser works. If providing more than one option, they are to be listed on a single line separated by spaces.

no-global-labels

Instructs the parser that the file intentionally contains no top level “labels” list.

no-points

Instructs the parser that the file intentionally contains no top level “points” list.

Example 1

```
   options: no-global-labels
```

Example 2

```
   options: no-global-labels no-points
```

labels

The list of nav points whose labels should be shown first as the zoom level increases. Typically you want this list to be fairly short or the map will quickly become cluttered. The individual nav points are separated with spaces. Note the two examples below provide effectively the same data.

Example 1

```
   labels: LEJEN GAMBT SHAMU BROWS HUULK LAX SXC LEJEN LNTRN XMANS
```

Example 2

```
   labels:
      '''
      LEJEN GAMBT SHAMU BROWS
      HUULK LAX SXC LEJEN LNTRN XMANS
      '''
```

routes

The list of routes starts with a left square bracket (“[“) and ends with a right square bracket (“]”). Each route object starts with a left curly bracket (“{“) and end with a right curly bracket (“}”). Multiple route objects can, and typically will be, listed inside a single list.

Example

```
   routes: [
      {
         # route object contents here
      }
      {
         # second route object contents here
      }
   ]
```

direction (route object)

Whether the route is inbound to the airport (STARs and approaches) or outbound (SIDs). Valid values (without quotes) are: “inbound” and “outbound”.

Example 1

```
direction: inbound
```

Example 2

```
direction: outbound
```

name (route object)

The name displayed on the site for the route. Typically these are the same as the official name for the STAR or SID.

Example 1

```
name: COMIX TWO
```

runways (route object)

The identifiers for the runways that the route applies to. One or more runways can be listed. Runways identifiers that are single digits can be preceded with a single digit but that is not required. Both forms are valid.

Example 1

```
runways: 09 18L 18R
```

Example 2

```
runways: 9 27
```

segments (route object)

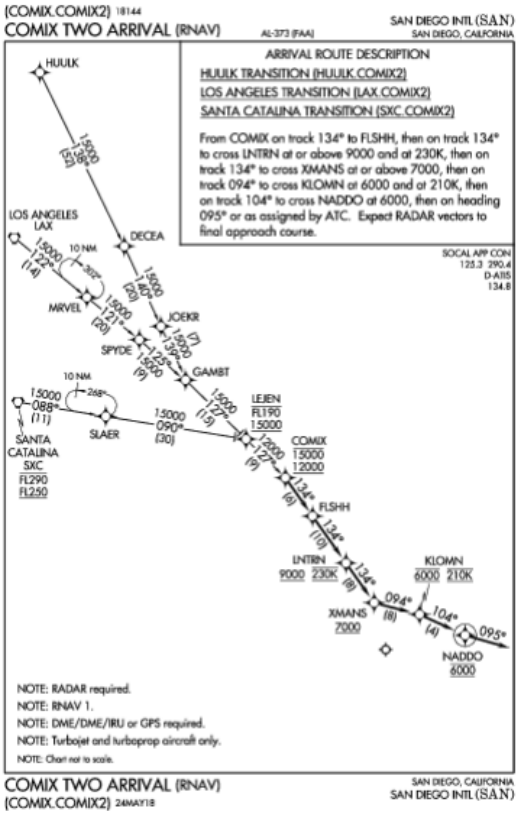
A list of nav points that together comprise a route. You only need to include the unique segments. The is best explained with an example using the COMIX TWO STAR into KSAN.

Aircraft can enter the route at three distinct points: HUULK, LAX and SXC. Aircraft that entered at HUULK and LAX merge at GAMBT. They in turn merge with aircraft that entered at XSC at LEJEN.

In the example below you find each of the waypoints from HUULK to GAMBT and LAX to GAMBT listed. The segment from GAMBT to LEJEN is only listed once.

The third entry point, SXC, is listed until it also ends at LEJEN. From LEJEN forward the remainder of the route is only listed once.

As with the list of labels, segments can be all on one line or can extend over multiple lines by using three single apostrophes ('). Example 1 and Example 2 below provide effectively the same data.



Example 1

```
        segments: [
           HUULK DECEA JOEKR GAMBT
           LAX MRVEL SPYDE GAMBT
           GAMBT LEJEN
           SXC SLAER LEJEN
           LEJEN COMIX FLSHH LNTRN XMANS KLOMN AJADE CATDG CRSNR SAYAE
        ]
```

Example 2

```
        segments: [
           HUULK DECEA JOEKR GAMBT
           LAX MRVEL SPYDE GAMBT
           GAMBT LEJEN
           SXC SLAER LEJEN
           '''
           LEJEN COMIX FLSHH LNTRN XMANS KLOMN AJADE
           CATDG CRSNR SAYAE
           '''
        ]
```

labels (route object)

This list functions the same way as the top level list of labels except they apply only when this route is visible on the map. Here too you want to choose the list of labeled points carefully to

avoid creating too much visual clutter on the map. Typically you want to label entry and exit points, points where segments converge / diverge and significant changes in heading.

Points

A list of nav points for which you are providing a target altitude or speed which will be visible in the tool tip for the point. Either an altitude, speed or both must be provided. You can only list the same nav point once and it must be assigned a single altitude and/or speed.

Example

```
   points: [
      {
         # point object contents here
      }
      {
         # second point object contents here
      }
   ]
```

name (point object)

The name of the nav point.

Example

```
         name: LNTRN
```

altitude (point object)

The altitude restrictions in effect when aircraft arrives at this nav point. A maximum, minimum and/or recommended altitude may be specified. Note that commas are not included.

Example

```
         altitude_required_max: 15000
         altitude_required_min: 12000
         altitude_recommended: 12000
```

speed (point object)

The speed restrictions in effect when aircraft arrives at this nav point. A maximum, minimum and/or recommended airspeed may be specified. Airspeed is specified in knots indicated airspeed.

Example

```
         speed_required_max: 230
         speed_required_min: 210
         speed_recommended: 210
```

Complete example for multiples routes into and out of KSAN

```
{
    airport: KSAN
    labels:
        '''
        LEJEN
        '''
    routes: [
        {
            direction: Inbound
            # combination of COMIX TWO and RNAV Z Rwy 27 Approach
            name: COMIX TWO
            runways: 27
            segments: [
                HUULK DECEA JOEKR GAMBT
                LAX MRVEL SPYDE GAMBT
                GAMBT LEJEN
                SXC SLAER LEJEN
                '''
                LEJEN COMIX FLSHH LNTRN XMANS KLOMN AJADE
                CATDG CRSNR SAYAE
                '''
            ]
            labels: HUULK LAX GAMBT SXC LEJEN LNTRN XMANS
        }
        {
            direction: Inbound
            name: SHAMU ONE
            runways: 9
            segments: [
                LAX EIREE SHAMU
                MZB SHAMU
                SHAMU SARGS
                OCN SARGS
            ]
        }
        {
            direction: Outbound
            name: BORDER SEVEN
            runways: 09 27
            segments: [
                MZB BROWS
                PGY POGGI BROWS
                BROWS JLI
                BROWS IPL
            ]
        }
    ]
    points: [
```

```
        {
            name: DECEA
            altitude_recommended: 15000
        }
        {
            name: LEJEN
            altitude_required_max: 19000
            altitude_required_min: 15000
        }
        {
            name: COMIX
            altitude_required_max: 15000
            altitude_required_min: 12000
            altitude_recommended: 12000
        }
        {
            name: LNTRN
            altitude_required_min: 9000
            speed_required_max: 230
            speed_required_min: 230
        }
        {
            name: XMANS
            altitude_required_min: 7000
        }
        {
            name: NADDO
            altitude_required_max: 6000
            altitude_required_min: 6000
        }
        {
            name: KLOMN
            altitude_required_max: 6000
            altitude_required_min: 6000
            speed_required_max: 210
            speed_required_min: 210
        }
        {
            name: AJADE
            altitude_recommended: 5200
        }
        {
            name: CRSNR
            altitude_recommended: 3700
        }
        {
            name: SAYAE
            altitude_recommended: 3200
        }
        {
```

```
            name: LAX
            altitude_required_max: 27000
        }
        {
            name: SHAMU
            altitude_required_max: 15000
            altitude_required_min: 15000
            speed_required_max: 250
            speed_required_min: 250
        }
    ]
}
```