PSAS Telemetry Server Documentation

Release L11-dev-6877684

Nathan Bergey, Ashley DeSimone, Bogdan Kovch, Karl Hiner, Emi

October 12, 2016

Contents

Contents

CHAPTER 1

Quickstart

If you just want to see telemetry, point your browser at a running instance of the server (for instance at a launch we will be running a server in mission control)
To install locally follow these instructions:

1.1 Installing

Read about python virualenv's!
Make sure you have python and pip

```
$ sudo apt-get install python2.7 python-pip virtualenvwrapper
```

Note: If this is your first time using python virtual environments, remember to kill your shell and open a new one after installing virtualenvwrapper for the first time (you only have to do this once).

To build the javascript you need coffeescript:

```
$ sudo apt-get install nodejs npm
```

Install globablly, since you might want this for other projects.:

```
$ sudo npm install -g coffee-script
```

Now you can build the js:

```
$ make build
```

For the server, create a python environment to run in:

```
$ mkvirtualenv psas-telemetry
```

Install python dependencies:

```
(psas-telemetry)$ pip install -r requirements.txt
```

1.2 Running

Start the telemetry server. If you changed some scripts, don't forget to rerun *make build*.:

```
(psas-telemetry)$ ./telemetry.py
```

1.3 Usage

Once the rocket is sending data and the backend server is running, simply navigate to http://localhost:8080 to start seeing data.

CHAPTER 2

Portland State Aerospace Society PSAS is a student aerospace engineering project at Portland State University. They are building ultra-low-cost, open source rockets that feature some of the most sophisticated amateur rocket avionics systems out there today.

One major challenge is how to display various data in real time [1]. So we've created a *real time telemetry viewer*.

[1] Or at least human reaction time suitable real time. Tens of milliseconds is okay.

CHAPTER 3

Overview

Incoming data is centered around the idea of *Feeds*. The code consists of a python sever that listens to feeds and sends data to a front end web app.

This way telemetry can be viewed from any device with a modern browser (phones, tablets, laptops, etc.).

Fig. 3.1: The telemetry server is the glue between raw data feeds and end users viewing the data

The front end is designed to be as configurable as possible, with different views stored in yaml files. This allows different users view only the data they are interested in and in a way that suits their device.

Footnotes: