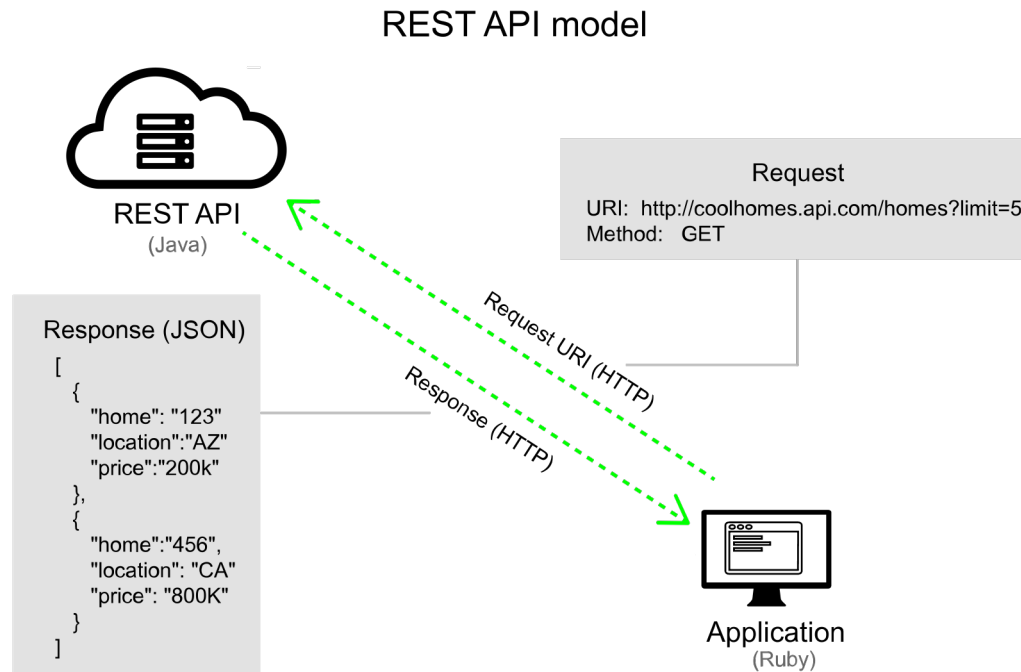


Cityflow – REST API Request



- Http – hvad er det?
 - Chrome dev-tools
- Shell – hvad er det?
 - Intall Git Bash (windows)
 - Curl – hvad er det?

```
http  shell  javascript

curl --request GET \
  --url https://api.cityflow.live/devices \
  --header 'authorization: Bearer {BEARER_TOKEN}'
```

Cityflow – REST API Request

1. Introduction
2. Protocols
3. Data Formats
4. Authentication, Part 1
5. Authentication, Part 2
6. API Design
7. Real-Time Communication
8. Implementation

The screenshot displays a REST client interface with two requests. The first request is a GET request to the login endpoint, and the second is a GET request to a script endpoint.

Request 1: GET /users/login

Navn: ☐ v1?sku=101gqxfpJUH9E&access_token=pk.eyJ1Ijoi... ☒ login ☐ login ☐ getUserDefaultLocation ☐ types ☐ devices ☐ organizations ☐ latest ☐ pina

43 anmodninger | 1.2 MB blev overført | 6.1 MB ressource

Headere: **Anmodningsheaderne**

- :authority: api.cityflow.live
- :method: OPTIONS
- :path: /users/login
- :scheme: https
- accept: */*
- accept-encoding: gzip, deflate, br
- accept-language: da-DK, da; q=0.9, en-US; q=0.8, en; q=0.7, de; q=0.6
- access-control-request-headers: authorization, content-type
- access-control-request-method: POST

Request 2: GET /runEnv.sh?start=ronne&destsel=ronne&date=&afg=..

Navn: ☒ runEnv.sh?start=ronne&destsel=ronne&date=&afg=.. ☐ Logger.js ☐ jquery.min.js ☐ SmartBomb.js ☐ ReferrerMonitor.js ☐ CoreAPI.js ☐ API.js ☐ ChromeAPI.js ☐ ComponentFactory.js

153 anmodninger | 892 kB blev overført | 1.1 MB ressource

Headere: **Parametre for forespørgselsstreng**

- start: ronne
- destsel: ronne
- date:
- afg: Kurt
- dest: Verner
- hobby: ridning
- gender: F
- mus: mus

API Intro

What An API Is and Why It's Valuable

Unfortunately, the characteristics that make websites optimal for humans make them difficult for computers to use.

The solution is an API. An API is the tool that makes a website's data digestible for a computer. Through it, a computer can view and edit data, just like a person can by loading pages and submitting forms

How An API Is Used

One side we have already talked about:

the server. This is the side that actually provides the API.

The other side is the "client." This is a separate program that knows what data is available through the API and can manipulate it, typically at the request of a user.

Protocols

Knowing the rules

Computers an etiquette as polite people, though it goes by the term "protocol." A computer protocol is an accepted set of **rules** that govern how two computers can speak to each other.

The Protocol of the Web

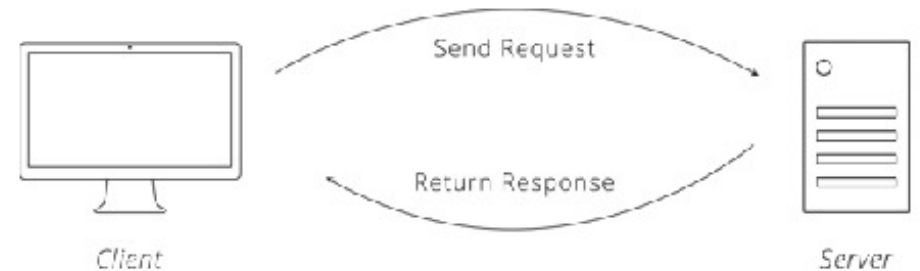
On the web, the main protocol is the Hyper-Text Transfer Protocol, better known by its acronym, HTTP.

HTTP Requests

Communication in HTTP centers around a concept called the Request-Response Cycle.

A valid request

- 1 URL (Uniform Resource Locator) [1](#)
- 2 Method
- 3 List of Headers
- 4 Body

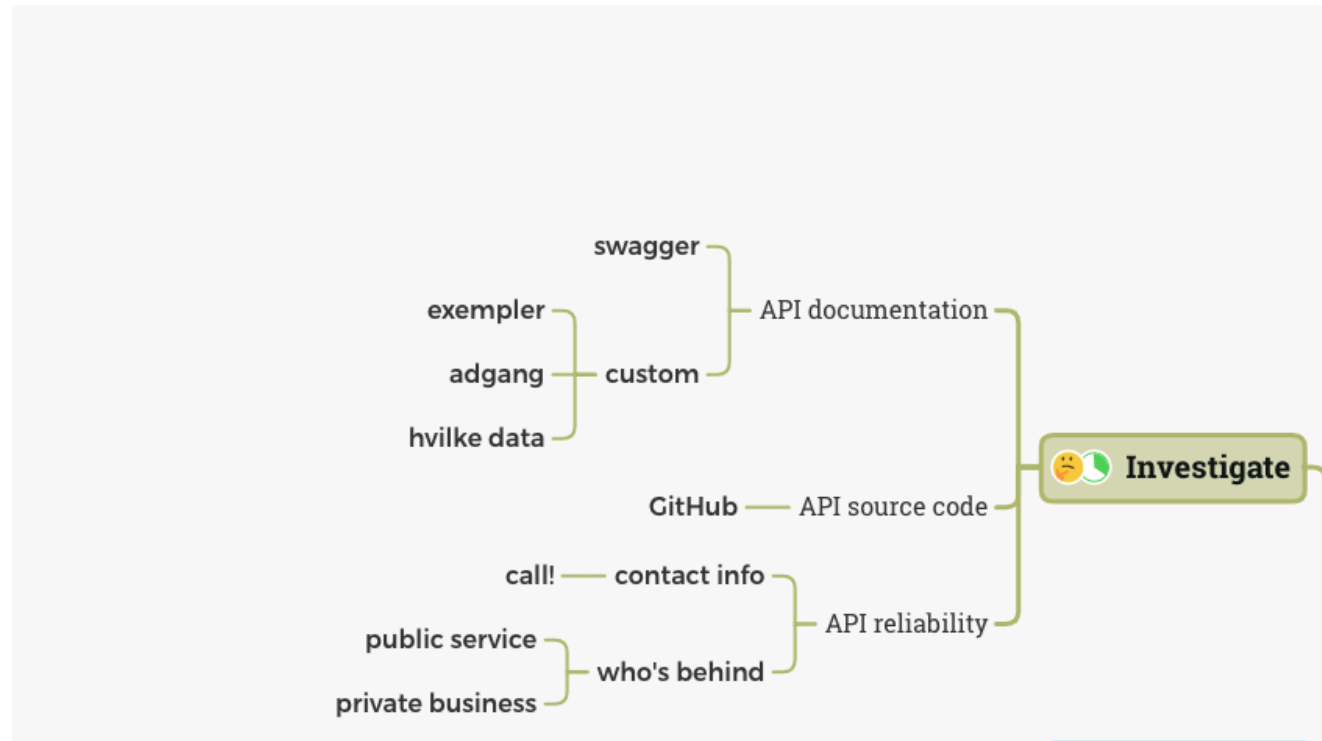


HTTP Response

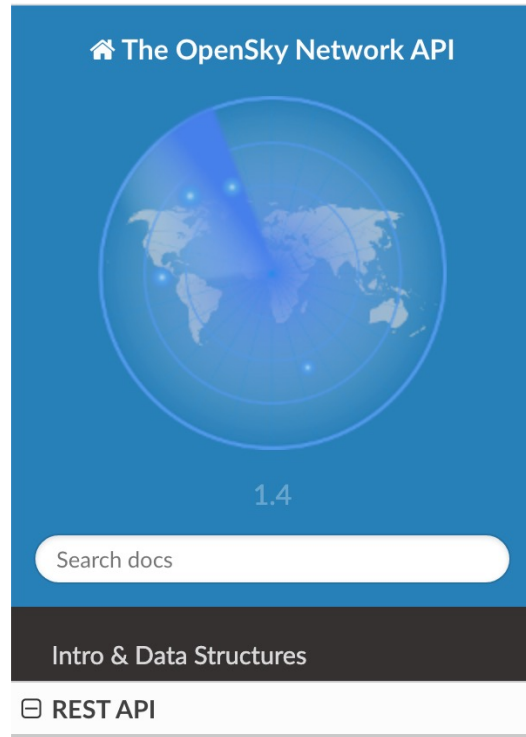
After the server receives a request from the client, it attempts to fulfill the request and send the client back a response. HTTP responses have a very similar structure to requests.

Pause

Data-retrieval: Public API -> Investigate



Recap OpenSky API



- 1) registrer dig på tjenesten
- 2) beskriv api'ets endpoints
 - 1) Overordnet
 - 2) Detaljer

Request endpoints

Intro & Data Structures

☐ REST API

- ⊕ All State Vectors
- ⊕ Own State Vectors
- ⊕ Flights in Time Interval
- ⊕ Flights by Aircraft
- ⊕ Arrivals by Airport
- ⊕ Departures by Airport
- ⊕ Track by Aircraft

GET /states/all

GET /flights/all

GET /flights/aircraft

GET /flights/arrival

ØVELSE m. curl i terminalen

Retrieve all states as an anonymous user:

```
$ curl -s "https://opensky-network.org/api/states/all" | python -m json.tool
```

Request:

<https://opensky-network.org/api/states/all?lamin=45.8389&lomin=5.9962&lamax=47.8229&lomax=10.5226>

Example query with bounding box covering **Switzerland**: <https://opensky-network.org/api/states/all?lamin=45.8389&lomin=5.9962&lamax=47.8229&lomax=10.5226>

```
[  
  51110b,  
  SAS646 ,  
  Estonia,  
  1664437945,  
  1664437945,  
  9.9992,  
  53.629,  
  null,  
  true,  
  4.89,  
  239.06,  
  null,  
  null,  
  null,  
  null,  
  false,  
  0
```

1	callsign	string	Callsign of the vehicle (8 chars). Can be null if no callsign has been received.
---	----------	--------	----------------------------------------------------------------------------------

Hvad er det?

Response

Øvelse:

Hvornår fejrer man 1234567890 dag?

Hvad skete der på denne dag:

2017-07-14 04:40:00 CEST

Task: Find all SAS-flights

```
[ 51110b,  
  SAS646 ,  
  Estonia,  
  1664437945,  
  1664437945,  
  9.9992,  
  53.629,  
  null,  
  true,  
  4.89,  
  239.06,  
  null,  
  null,  
  null,  
  null,  
  false,  
  0  
],
```

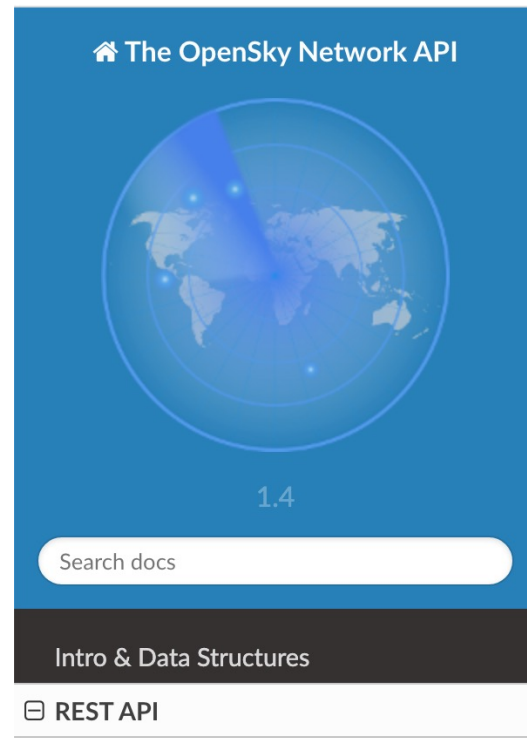
Response

			Callsign of the vehicle (8 chars). Can be null if no callsign has been received.
1	callsign	string	

```
18 # get rows where string matches substring  
19 retcount <- str_detect(statedf$V2,"SAS")  
20 sasdf <- statedf[retcount,]  
21 sum(retcount)
```

```
> sum(retcount)  
[1] 68
```

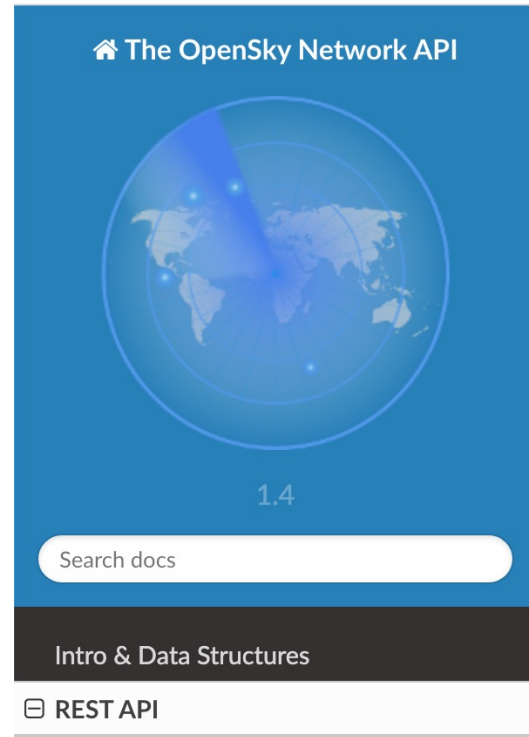
Data-retrieval CASE: OpenSky API



- 1) Hvor mange SAS fly er der i luften lige nu?
- 2) Hvor mange flyver over Point Reyes i løbet af en time?
 - 1) i løbet af et minut
 - 2) i løbet af 3 døgn
 - 1) lav et script som gemmer din dataframe i en fil
 - 2) kør det i konsollen/terminal
- 3) Find jeres eget område og sæt det op i et script (til fredag)
- 4) Lav en liste over fly over DK
 - 1) Konstruér en algoritme som spotter "cirkende" fly

Data-retrieval CASE: OpenSky API

liste over Point Reyes



ROPER33
United States - US Air Force (USAF) **flightradar24** LIVE AIR TRAFFIC

© CJMoeser

N/A **N/A**

ACTUAL **21:36** ESTIMATED -

AIRCRAFT TYPE (T38)
Northrop T-38A Talon

REGISTRATION **64-13285** COUNTRY OF REG.

SERIAL NUMBER (MSN) AGE **N/A**

Recent 64-13285 flights

CALIBRATED ALTITUDE **3 000 ft** VERTICAL SPEED



Point P

Fra api-c

Example que

[network.org/](https://opennetwork.org/)

<https://opennetwork.org/>
&lamin=5.9
10.5226

Finde bbox

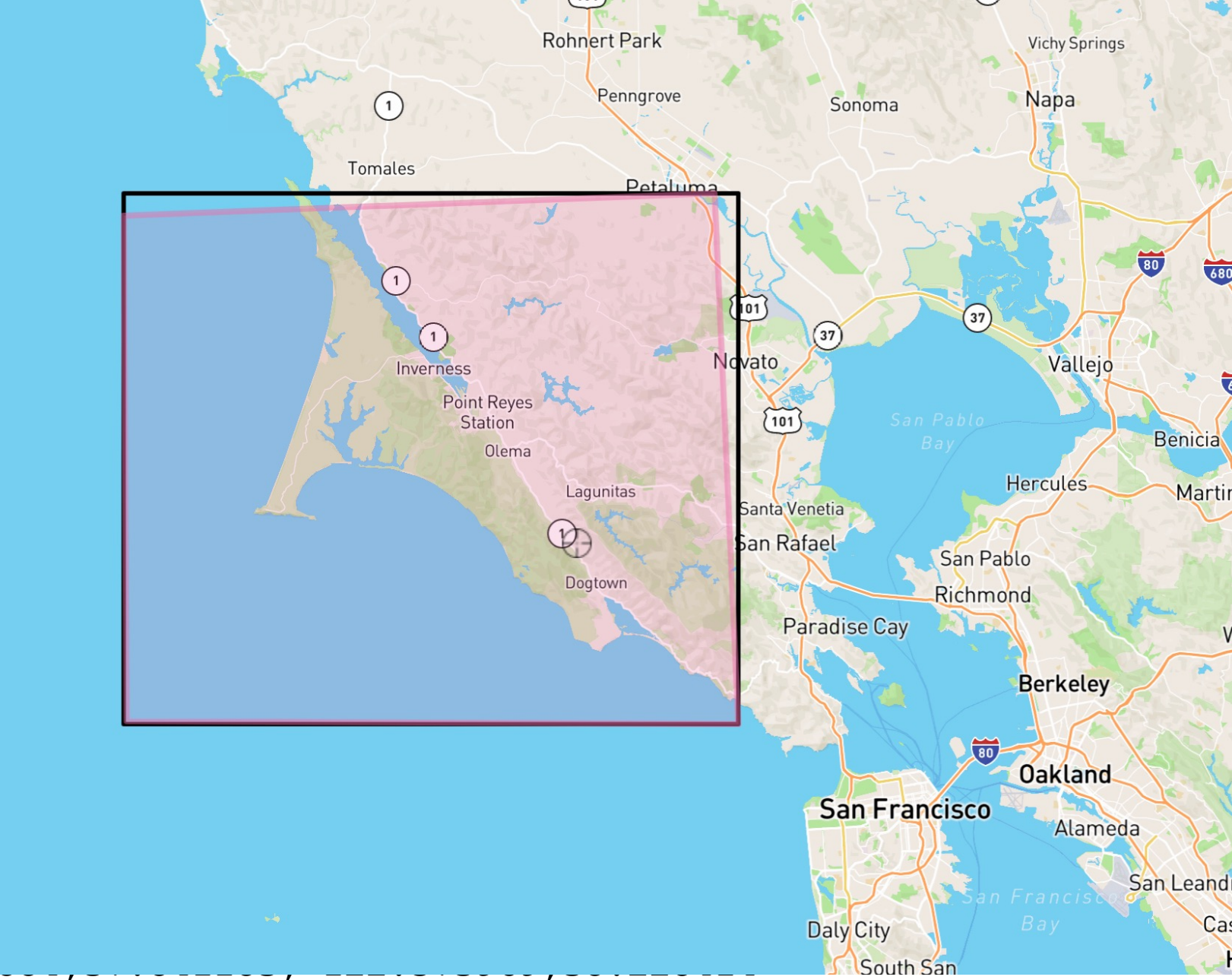
lamin=

lamax=

lomin=

lomap=

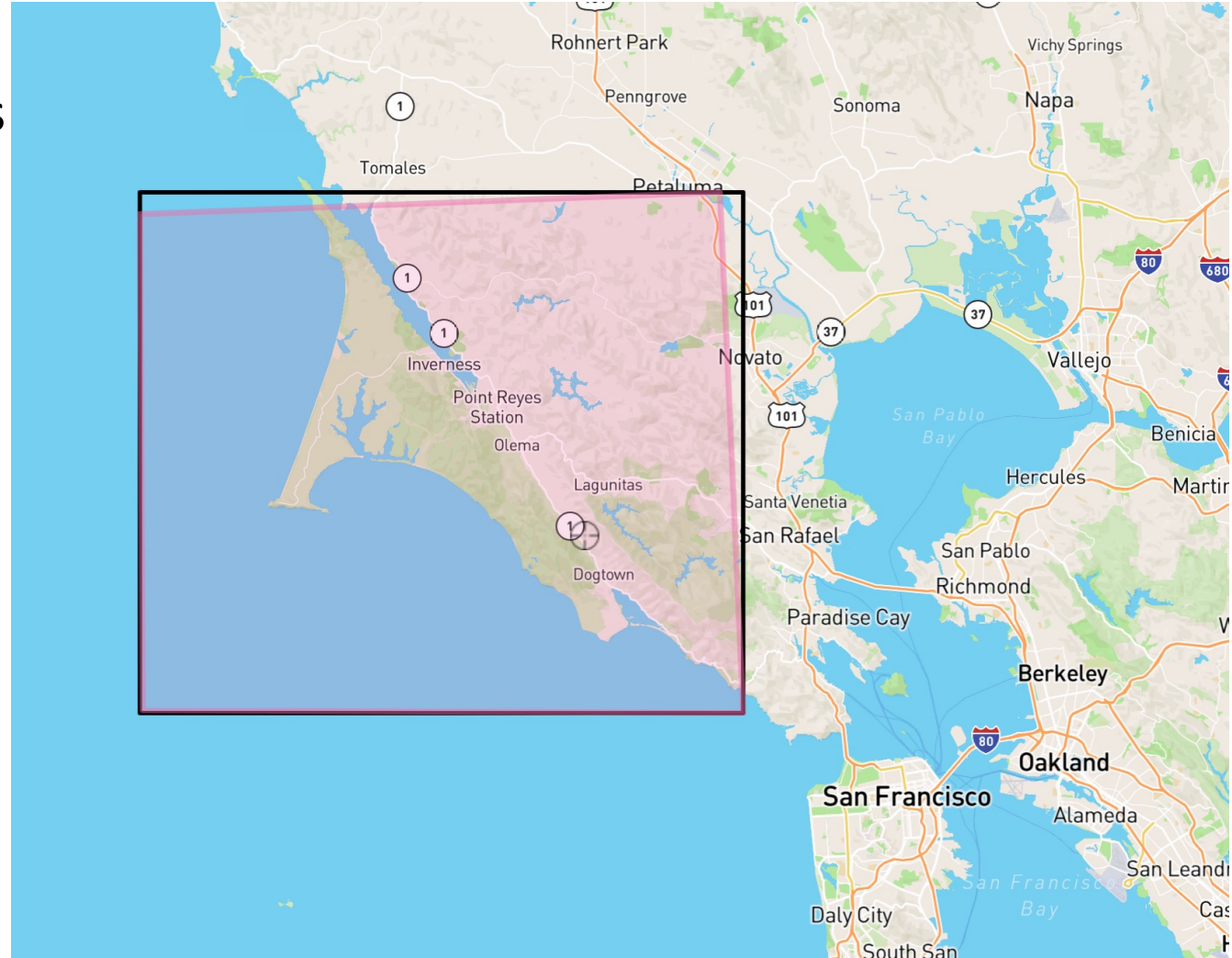
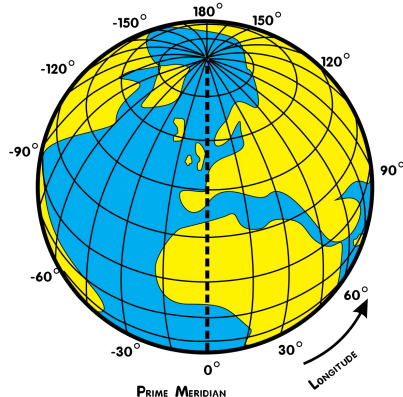
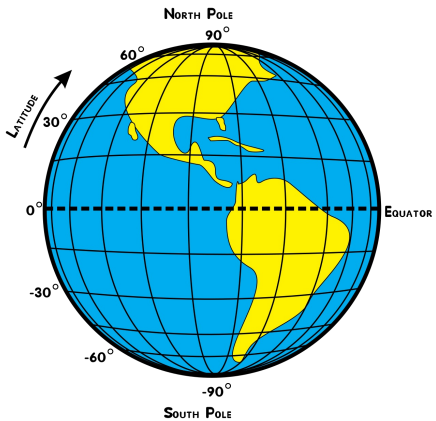
-123.143



Point Reyes flight-data-collection

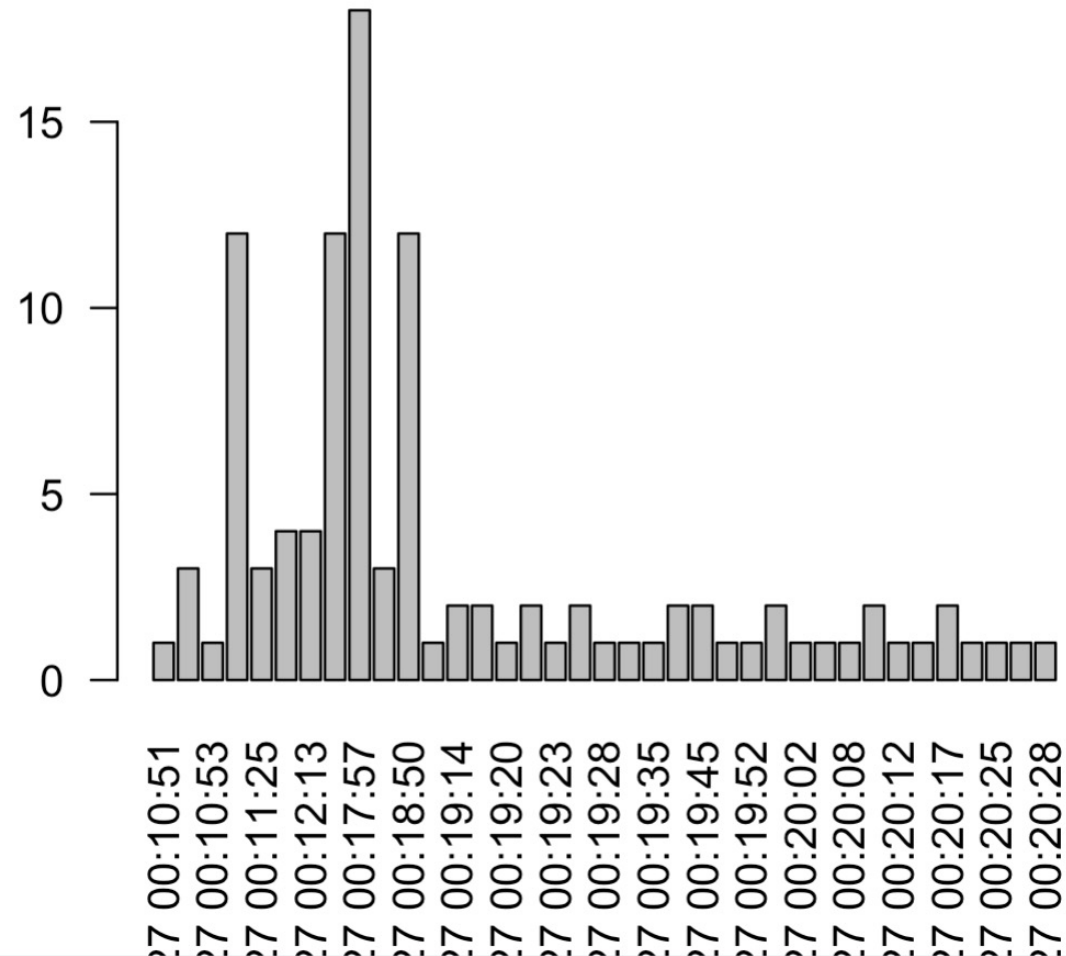
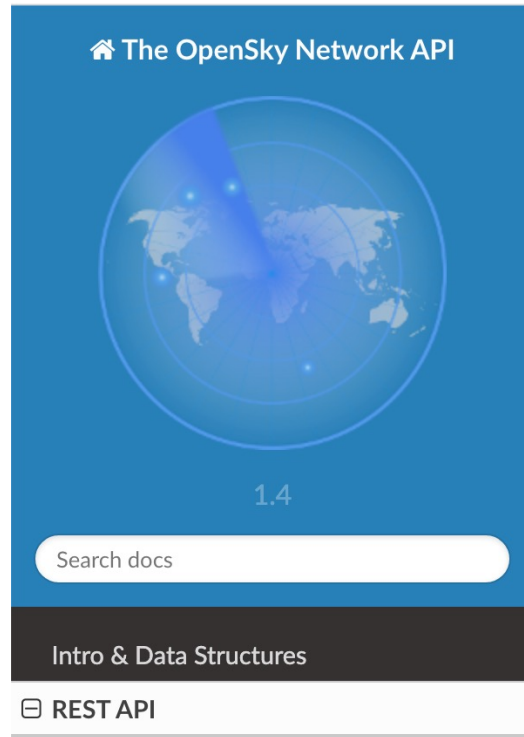
Finde bbox omkring Point Reyes

lamin= 37.841183
lamax=38.228414
lomin= -123.143884
lomap= -122.573969



Data-retrieval CASE: OpenSky API

liste over Point Reyes



ØVELSE:

kør snippet

<https://gist.github.com/cphwulf/a661c2ea17493615cd82a2b663c43704>

og find vektoren med colnames

Pause

Data-retrieval CASE: OpenSky API

Liste over fly som krydser eget spor

prflightlist	list [11]	List of length 11
[[1]]	list [38 x 17] (S3: data.frame)	A data.frame with 38 rows and 17 columns
[[2]]	list [38 x 17] (S3: data.frame)	A data.frame with 38 rows and 17 columns
[[3]]	list [39 x 17] (S3: data.frame)	A data.frame with 39 rows and 17 columns
[[4]]	list [39 x 17] (S3: data.frame)	A data.frame with 39 rows and 17 columns
[[5]]	list [38 x 17] (S3: data.frame)	A data.frame with 38 rows and 17 columns

```
totaldf <- do.call('rbind',mydf)
```

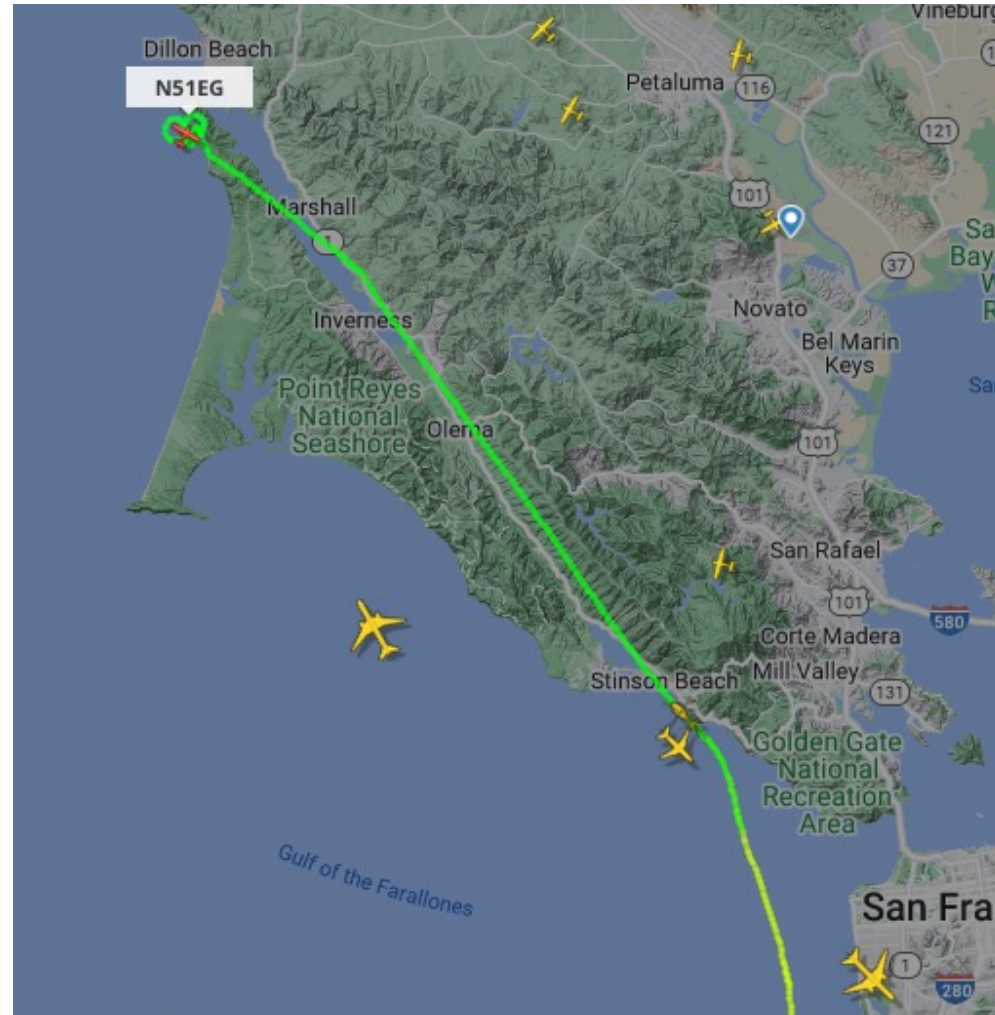
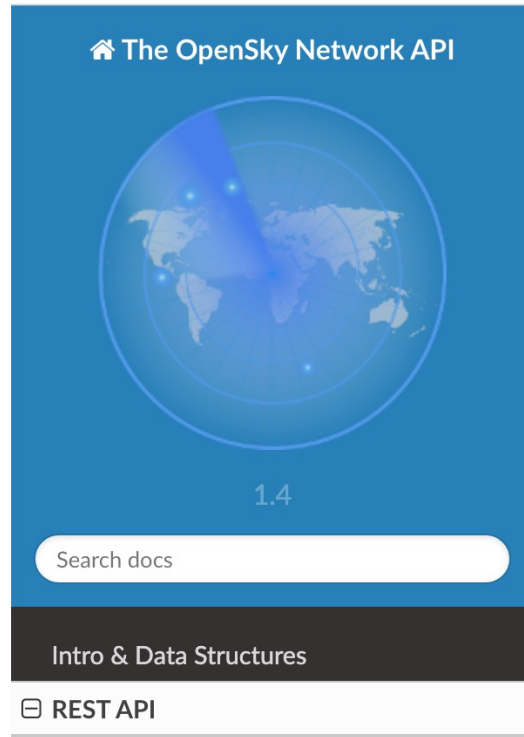
icao24	callsign	time_position
a808ba	N61696	1664482763
a4142f	MMY48	1664482763
a569c4	EJM448	1664482763
aaa590	UAL1728	1664482763
a8a608	N65647	1664482514
aca97c	N915CM	1664482763
a6b572	N531PI	1664482587

```
as.POSIXct(as.numeric(pltdf[1,3]), origin="1970-01-01")
```

	icao24	callsign	time_position	xdate
36	a4ddcd	N412Z	1664482749	2022-09-29 22:19:09
21	a19c4b	N20230	1664482758	2022-09-29 22:19:18
59	a19c4b	N20230	1664482758	2022-09-29 22:19:18
19	a9d015	N731PP	1664482760	2022-09-29 22:19:20
22	a7da84	N605HC	1664482760	2022-09-29 22:19:20

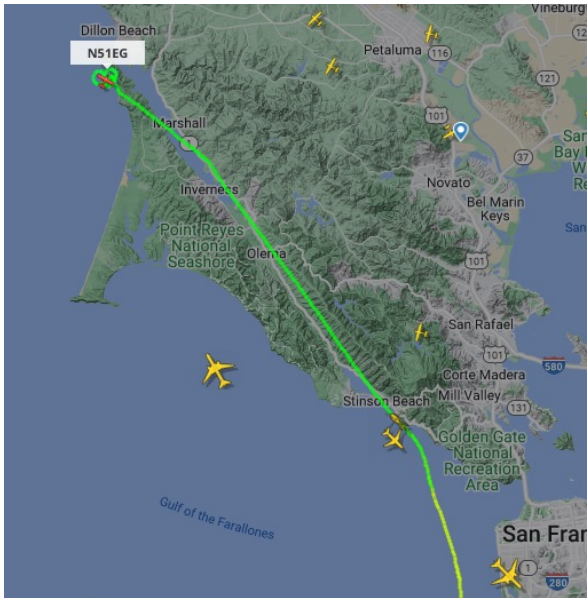
Data-retrieval CASE: OpenSky API

Liste over fly som krydser eget spor



Data-retrieval CASE: OpenSky API

Liste over fly som krydser eget spor

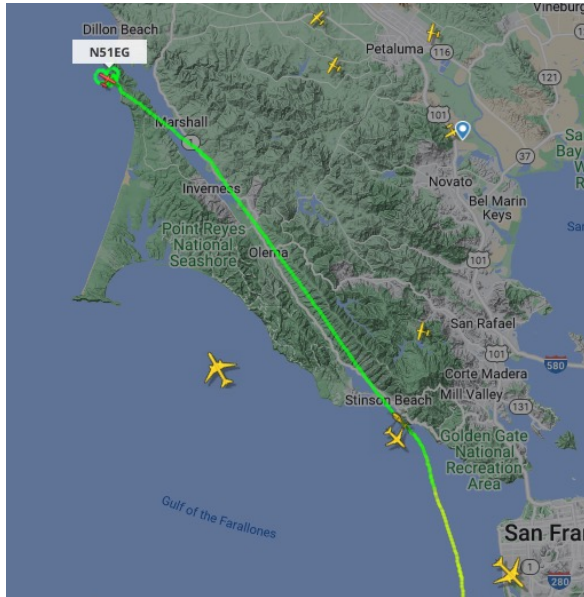








	icao24	callsign	origin_country	time_position	last_contact	longitude	latitude
155	a65f81	N51EG	United States	1664482732	1664482732	-122.9804	38.2029
139	a65f81	N51EG	United States	1664482715	1664482715	-122.9738	38.2145
123	a65f81	N51EG	United States	1664482693	1664482694	-122.9708	38.2024
108	a65f81	N51EG	United States	1664482676	1664482676	-122.9713	38.2064
94	a65f81	N51EG	United States	1664482656	1664482656	-122.9767	38.2068
81	a65f81	N51EG	United States	1664482631	1664482632	-122.9807	38.2071
65	a65f81	N51EG	United States	1664482606	1664482611	-122.9776	38.2028
49	a65f81	N51EG	United States	1664482596	1664482596	-122.9662	38.195
33	a65f81	N51EG	United States	1664482575	1664482575	-122.962	38.1907
18	a65f81	N51EG	United States	1664482551	1664482551	-122.9538	38.1865
4	a65f81	N51EG	United States	1664482530	1664482532	-122.9778	38.2045

	icao24	callsign	origin_country	time_position	last_contact	longitude	latitude
69	484368	KLM109	Kingdom of the Netherlands	1664482612	1664482612	-122.6345	37.863
53	484368	KLM109	Kingdom of the Netherlands	1664482596	1664482596	-122.6549	37.8829
37	484368	KLM109	Kingdom of the Netherlands	1664482576	1664482576	-122.6747	37.9112
22	484368	KLM109	Kingdom of the Netherlands	1664482552	1664482552	-122.6874	37.9491
8	484368	KLM109	Kingdom of the Netherlands	1664482532	1664482533	-122.6962	37.9798

Data-retrieval CASE: OpenSky API

Liste over fly som krydser eget spor



 cphwulf final	
 dkplay.sh	first try
 newsapi.R	first try
 opensky.R	final
 openskylistofplanes.R	final
 openskyprep.R	final

kl 11: Fremlæggelse af jeres bud på
algoritme til at spotte "cirkellende" fly