

Uçak Verilerinin Yakalanması ve Üzerinde Yapılan Olası Çalışmalar

İsim Soyisim Numara
İsim Soyisim Numara

BİTİRME ÇALIŞMASI

KOCAELİ
Mayıs, 2022

Uçak Verilerinin Yakalanması ve Üzerinde Yapılan Olası Çalışmalar

**İsim Soyisim Numara
İsim Soyisim Numara**

BİTİRME ÇALIŞMASI

**Danışman : Doç. Dr.
Üye :
Üye :**

**KOCAELİ
Mayıs, 2022**

İÇİNDEKİLER:

SİMGE LİSTESİ

KISALTMA LİSTESİ

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

ÖZET

ABSTRACT

KAYNAKLAR

SİMGE LİSTESİ:

KISALTMALAR:

| | |
|-------|---|
| CWI | : Centrum Wiskunde & Informatica |
| CNRI | : Corporation for National Research Initiatives |
| PSF | : Python Software Foundation |
| HTTP | : Hypertext Transfer Protocol |
| JSON | : Javascript Object Notation |
| HTML | : The HyperText Markup Language |
| XLT | : Microsoft Excel |
| PHP | : Hypertext Preprocessor |
| SOAP | : Simple Object Access Protocol |
| IOT | : Internet of Things |
| ADS-B | : Automatic Dependent Surveillance-Broadcast |
| IP | : Internet Protocol |
| DNS | : Domain Name System |
| WIFI | : Wireless Fidelity |
| USB | : Universal Serial Bus |
| HDMI | : High-Definition Multimedia Interface |

ŞEKİL LİSTESİ:

ÇİZELGE LİSTESİ:

ÖNSÖZ:

ÖZET:

ABSTRACT:

1. GİRİŞ

Bu bölümde Python programlama dili, OpenSky Rest API konularına değinilecektir.

1.1. PYTHON

Konunun daha iyi anlaşılabilmesi için kullanılan programlama dili olan Python programlama dili ne olduğuna ve tarihine değinelim.

1.1.1 PYTHON NEDİR?

Python yorumlayıcı, nesne tabanlı, üst düzey bir dinamik semantik programlama dilidir. Yüksek level veri yapılarını işleyebilir, dinamik yazma ve dinamik birleştirme yapabilir, bu da hızlı uygulama yazma ve geliştirmeyi kolay hale getirir. Python basit ve kolay öğrenilen bir dil yapısına sahiptir. Python program yapısını ve yeniden kullanım için eklenti ve paketlerini destekler. Python yorumlayıcısı ve kapsamlı standart kütüphaneleri, bütün büyük platformlar için ücretsiz olarak yazılım veya ikilik sistemde mevcuttur ve ücretsiz olarak dağıtılabilir.

Programcılar, sağladığı üretkenlik nedeniyle genellikle Python’a hayran kalırlar. Python’da derleme işlemi olmadığından, düzenleme, test ve hata ayıklama döngüsü hızlıdır. Python programlama dilinde hata ayıklama koladır; bir hata veya hatalı giriş bölünme hatasına sebep olmaz. Bunun yerine, yorumlayıcı bir hata bulduğunda bir istisna oluşturur. Program istisnayı yakalayamadığında, yorumlayıcı bir yığın izi yazdırır. Hata ayıklayıcı, yerel ve genel değişkenlerin incelenmesine, rastgele ifadelerin değerlendirilmesine, kesme işaretlerinin ayarlanmasına, kodda bir seferde bir satır ve adım adım ilerleme gibi seçeneklere izin verir. Hata ayıklayıcısı, Python programlama dili ile yani kendisiyle yazılmıştır. Bir başka özelliği, genellikle bir programda hata ayıklamanın en hızlı yolu, koda birkaç yazdırma (print) fonksiyonu eklemektir. Hızlı düzenleme, test ve hata ayıklama döngüsü bu basit yaklaşımı çok etkili kılar.

Python programlama dilinin temelleri C programlama dili ile atılmıştır. Her ne kadar Python programlama dilinin adı python (python) yılanından geldiği düşünülse de öyle değildir; Guido Van Rossum dilin adını, bir İngiliz komedi grubu olan “Monty Python’s Flying Circus” adlı gösterisinden esinlenerek koymuştur. Buna rağmen Python programlama dilinin pek çok yerde yılan figürü ile temsil edilmesi neredeyse gelenek haline gelmiştir.

1.1.2. PYTHON'UN TARİHİ

Python programlama dili, Guido van Rossum tarafından 1989'lu yıllarda Hollanda'da CWI (Centrum Wiskunde & Informatica: Matematik ve Bilgisayar Bilimleri Merkezi) araştırma merkezinde yapımına başlandı. Python programlama dili başkaları tarafından destek almasına rağmen Guido van Rossum halen Python'un başyazarı olmaya devam etmektedir.

1995'te, Guido Virginia eyaletinin Reston kasabasında bulunan CNRI'da (Corporation for National Research Initiatives: Ulusal Araştırma Girişimleri Kurumu) Python üzerinden çalışmaya devam etti. CNRI'da çalışırken bir kaç Python sürümü yayınladı.

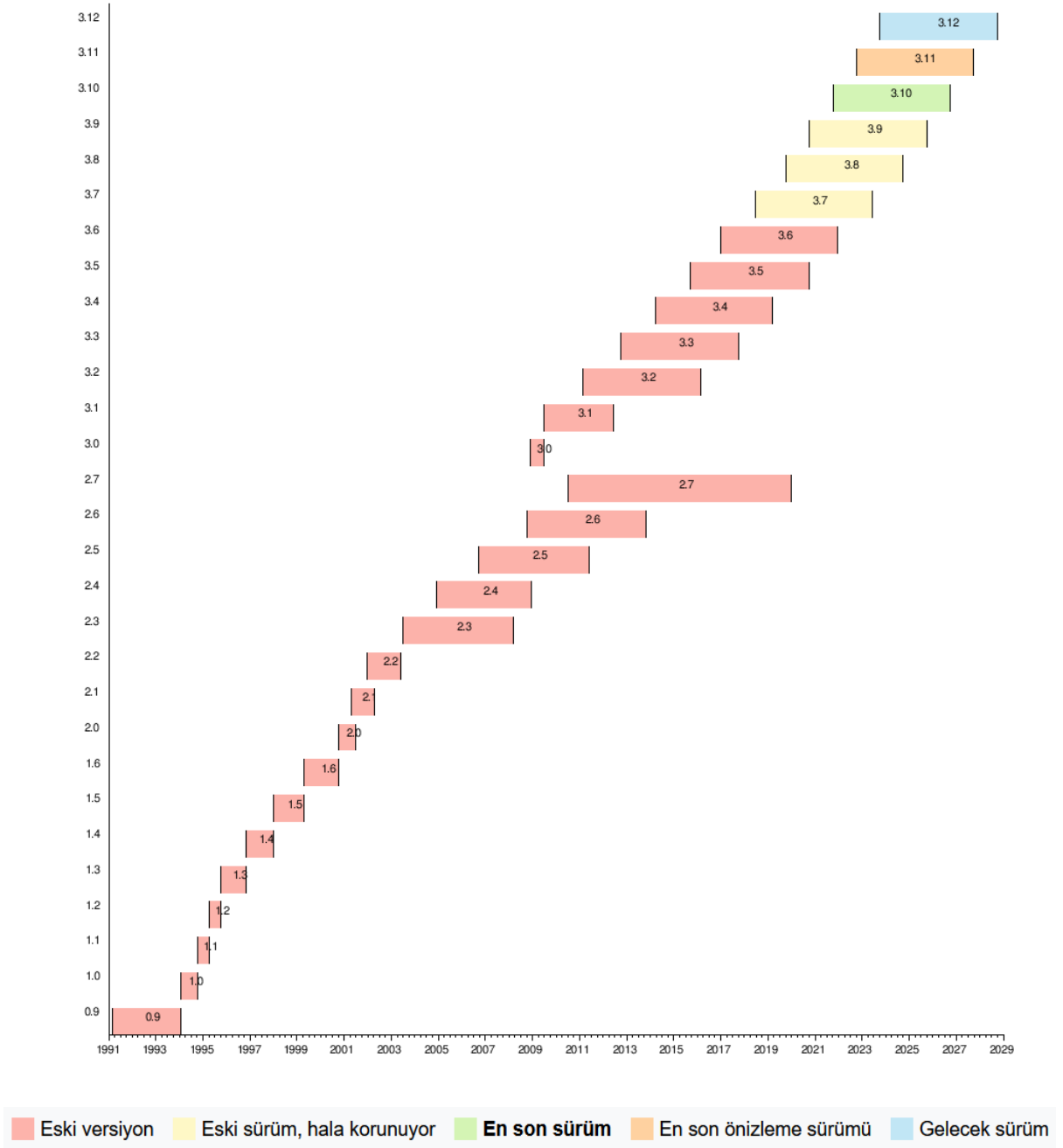
2000 yılının Mayıs ayında, Guido ve Python yazılım geliştirme ekibi BeOpen PythonLabs ekibini oluşturmak için BeOpen.com'a taşındılar. Aynı yılın Ekim ayında, PythonLabs takımı Digital Creations'a (şimdiki isimleri: Zope Corporation) geçiş yaptılar. 2001 yılında, PSF (Python Software Foundation: Python Yazılım Vakfı) oluşturuldu, Python ile ilgili Intellectual Property'e (Fikri Mülkiyet) sahip olabilmek için özellikle oluşturuldu, herhangi bir kar amacı gütmeyen bir kuruluştur. Zope Corporation, PSF'nin sponsor üyesidir.

Bütün Python sürümleri Open Source (Açık Kaynak) üzerinden yayınladı. Tarihsel olarak, çoğu ama hepsi değil Python sürümlerinin bazıları ayrıca GPL (General Public License) uyumlu olarakta yayınladı.

Not: GPL uyumlu olması Python'u GPL altında yayınlamak demek değildir. Bütün Python lisansları, GPL'den farklı olarak, yaptığımız değişiklikleri açık kaynak olmadan da dağıtmanıza izin verir. GPL uyumlu lisanslar, diğerlerinin aksine; Python'un GPL kapsamında yayınlanan diğer yazılımlarıyla birleştirmeyi mümkün kılar.

ÇİZELGE 1 Aşağıdaki tablo Python'un bazı sürüm bilgilerini göstermektedir

| Sürüm | Türetildi | Yıl | Hak Sahibi | GPL Uyumlu? |
|----------------|-----------|-----------------|------------|-------------|
| 0.9.0 - 1.2 | n/a | 1991-1995 | CWI | Evet |
| 1.3 - 1.5.2 | 1.2 | 1995-1999 | CNRI | Evet |
| 1.6 | 1.5.2 | 2000 | CNRI | Hayır |
| 2.0 | 1.6 | 2000 | BeOpen.com | Hayır |
| 1.6.1 | 1.6 | 2001 | CNRI | Hayır |
| 2.1 | 2.0+1.6.1 | 2001 | PSF | Hayır |
| 2.0.1 | 2.0+1.6.1 | 2001 | PSF | Evet |
| 2.1.1 | 2.1+2.0.1 | 2001 | PSF | Evet |
| 2.1.2 | 2.1.1 | 2002 | PSF | Evet |
| 2.1.3 | 2.1.2 | 2002 | PSF | Evet |
| 2.2 ve sonrası | 2.1.1 | 2001 ve sonrası | PSF | Evet |



ŞEKİL 1 (a) Python programlama dili sürümleri ve destek verildiği yıllar gösterildiği görsel
(Wikipedia History of Python 2022)

1.2. REST API

Online ortamda veri oluřturma ve paylařmanın hangi yöntemler ile mümkün kılındığını anlatalım.

1.2.1. API NEDİR?

API, uygulama yazılımı oluřturmak ve entegre etmek için bir dizi tanım ve protokoldür. Bazen bir bilgi saęlayıcı ile bilgi kullanıcısı arasındaki, tüketiciden istenen içerięi (çaęrı) ve üretici tarafından istenen içerięi (cevap) belirleyen bir sözleşme olarak anılır. Örneęin; bir hava durumu hizmeti için API tasarımı, kullanıcının bir posta kodu saęlamasını ve üreticinin, ilki yüksek sıcaklık ve ikincisinin düşük olmak üzere iki parçalı bir yanıtla yanıt vermesini belirtebilir.

Bařka bir deyiřle, bilgi almak veya bir işlevi gerçekleřtirmek için bir bilgisayar veya sistem etkileřim kurmak istiyorsanız, API, isteęi anlayıp yerine getirebilmesi için o sisteme ne istediğini iletmenize yardımcı olur.

Bir API'yi kullanıcılar veya istemciler ile almak istedikleri kaynaklar veya web hizmetleri arasında bir aracı olarak düşünülebilir. Aynı zamanda bir kuruluřun güvenlik, kontrol ve kimlik doęrulamasını sürdürürken kimin neye erişebileceğini belirleyerek kaynakları ve bilgileri paylařmasının bir yoludur.

API'nin bir bařka avantajı da, kaynaęınızın nasıl alındığını veya nereden geldiğini bilmenize gerek olmamasıdır.

1.2.2. REST NEDİR?

REST bir protokol veya standart değil, bir dizi mimari kısıtlamadır. API geliştiricileri REST'i çeşitli şekillerde uygulayabilir.

RESTful API aracılığı ile bir istemci isteği yapıldığında kaynağın durumunun bir temsilini istekte bulunana veya son kısma aktarır. Bu bilgi veya beyan, birkaç şekilde sunulabilir. Örneğin; HTTP: JSON (Javascript Object Notation), HTML, XLT, Python, PHP veya metin belgesi formatı tipinde sunabilir. JSON genel olarak en çok kullanılan dosya tipidir çünkü adına rağmen Javascript programlama dilinden bağımsızdır ve insanlar hemde makineler tarafından okunabilir.

Bir API'nin RESTful olarak kabul edilebilmesi için şu kriterlere uyması gerekir.

- İstemciler sunucular ve kaynaklardan oluşan ve istekleri HTTP aracılığı ile yönetilen bir istemci ve sunucu mimarisi olmalıdır.
- İstemci sunucu arasındaki etkileşimleri kolaylaştıran önbelleğe alınabilir veriler olmalıdır.
- Her tür sunucuyu organize eden katmanlı bir sistem, istenen bilgilerin istemci tarafından görülmeyecek şekilde düzenler halinde alınmasını içeriyor olmalıdır.
- İsteğe bağlı kod, istendiğinde sunucudan istemciye yürütülebilir kod gönderme, istemci işlevselliğini genişletme yeteneğine sahip olmalıdır.

REST API'nin uyması gereken bu özelliklere sahip olmasına rağmen, XML mesajlaşma ve yerleşik güvenlik ve işlem uyumluluğu gibi belirli gereksinimleri olan SOAP (Simple Object Access Protocol) gibi önceden belirlenmiş bir protokole göre kullanımı daha kolay kabul edilir.

Buna karşılık REST, gerektiğinde uygulanabilen artırılmış ölçeklenebilirlik ile REST API'lerini daha hızlı ve hafif hale getiren IOT (Internet of Things: "Nesnelerin İnterneti") ve mobil uygulama geliştirme için mükemmel olan bir dizi yönergedir.

1.2.3. OPENSKY REST API

OpenSky REST API bağlantısı: <https://opensky-network.org/api>

Veriler içerisinde durum vektörleri ve uçuş rotalarını almak için kullanılabilecek çeşitli bilgiler vardır.

1.2.3.1. Tüm Durum Vektörleri

Aşağıdaki API çağrısı, OpenSky’ın herhangi bir durum vektörünü almak için kullanılabilir.

<https://opensky-network.org/api/states/all>

1.2.3.2. Talepte Bulunma

Aşağıdaki talep parametrelerinden istenilenler kullanılarak belirli uçuklar veya saatler için durum bilgisi talep edilebilir.

ÇİZELGE 2 Talep parametreleri

| Nitelik | Veri Tipi | Tanım |
|---------|-------------------|--|
| time | integer (sayısal) | Saniye cinsinden süre durumlarını almak için Unix zaman damgası kullanılır. Yazılmazsa anlık zaman kullanılır. |
| icao24 | string (sözel) | Onaltılık diziyle temsil edilen bir veya daha fazla ICAO24 aktarıcı adresi (örn: abc9f3) kullanır. Birden çok ICAO24 verisini filtrelemek için her adres için bir kez ekleyin. Yazılmazsa tüm uçakların durum vektörleri döndürülür. |

Buna ek olarak WGS84 koordinatlarını ile bir sınırlayıcı bölge oluşturarak sadece o bölgede bulunan uçakların verisini almak mümkündür. Bu amaçla, aşağıdaki parametrelerin hepsini ekleyiniz.

ÇİZELGE 3 Koordinat sisteminde alabileceği değerlerin veri tipleri

| Nitelik | Veri Tipi | Tanım |
|---------|-----------|---|
| lamin | float | Enlem için ondalık derece cinsinden alt sınır. |
| lomin | float | Boylam için ondalık derece cinsinden alt sınır. |
| lamax | float | Enlem için ondalık derece cinsinden üst sınır. |
| lomax | float | Boylam için ondalık derece cinsinden üst sınır. |

<https://opensky-network.org/api/states/all>

Zaman ve uçak için örnek sorgu:

<https://opensky-network.org/api/states/all?time=1458564121&icao24=3c6444>

İsviçre için sınırlayıcı bölge örnek sorgu:

[https://opensky-network.org/api/states/all?](https://opensky-network.org/api/states/all?lamin=45.8389&lomin=5.9962&lamax=47.8229&lomax=10.5226)

[lamin=45.8389&lomin=5.9962&lamax=47.8229&lomax=10.5226](https://opensky-network.org/api/states/all?lamin=45.8389&lomin=5.9962&lamax=47.8229&lomax=10.5226)

1.2.3.3. Gelen Yanıt

Gelen yanıt, aşağıdaki özelliklere sahip bir JSON nesnesidir.

ÇİZELGE 4 Gelen yanıtın veri tipleri

| Nitelik | Veri Tipi | Tanım |
|---------|-------------------|---|
| time | integer (sayısal) | Bu yanıtta durum vektörlerinin ilişkili olduğu zamanı temsil eder. Tüm vektörler [time - 1, time] aralığına sahip bir aracın durumunu temsil eder |
| states | array (dizi) | Durum vektörlerini barındırır. |

States (sınıf) özelliği iki boyutlu bir dizidir. Her satır bir durum vektörünü temsil eder ve aşağıdaki alanları içerir.

ÇİZELGE 5 Gelen veri içerisindeki veriler ve verilerin tipleri

| Dizin Numarası | Nitelik | Veri Tipi | Tanım |
|----------------|----------------|-----------|--|
| 0 | icao24 | string | Özgün ICOA 24 bit adresinde onaltılık gösterimi. |
| 1 | callsign | string | Uçağın çağrı işareti (8 karakter). Çağrı bilgisi alınmadıysa boş bırakılır. |
| 2 | origin_country | string | ICAO 24 bit adresinden anlaşılan ülke adı. |
| 3 | time_position | int | Son konum güncellemesi için, Unix zaman damgası saniye cinsinden. Son 15 saniye içinde OpenSky tarafından herhangi bir pozisyon bilgisi alınmadıysa boş bırakılır. |

| | | | |
|----|-----------------|---------|---|
| 4 | last_contact | int | Genel olarak son güncelleme için Unix zaman damgası saniye cinsinden. Bu alan, aktarıcından alınan herhangi yeni, geçerli mesaj için güncellenir. |
| 5 | longitude | float | Ondalık derece cinsinden WGS-84 boylam bilgisi. Boş olabilir. |
| 6 | latitude | float | Ondalık derece cinsinden WGS-84 enlem bilgisi. Boş olabilir. |
| 7 | baro_altitude | float | Metre cinsinden barometrik yükseklik. Bol olabilir. |
| 8 | on_ground | boolean | Konumun bir yüzey konumu raporundan alınıp alınmadığını Boolean tipinde gösterir. |
| 9 | velocity | float | Yere göre hız m/s olarak. Boş olabilir. |
| 10 | true_track | float | Kuzeyden saat yönünde (kuzey=0°) gerçek yönü. Boş olabilir. |
| 11 | vertical_rate | float | m/s cinsinden dikey hız. Pozitif bir değer uçağın tırmandığını, negatif bir değer ise alçaldığını gösterir. Boş olabilir. |
| 12 | sensors | int[] | Bu durum vektörü katkıda bulunan alıcıların kimlikleri. İstekte sensör için filtreleme kullanılmadıysa boştur. |
| 13 | geo_altitude | float | Metre cinsinden geometrik yükseklik. Boş olabilir. |
| 14 | squawk | string | Aktarıcı kodu. Boş olabilir. |
| 15 | spi | boolean | Uçuş durumunun özel amaçlı göstermesi olup olmadığını. |
| 16 | position_source | int | Konumunun kökeni: 0 = ADS-B, 1 = ASTERIX, 2 = MLAT |

1.2.3.4. Kısıtlamalar

1.2.3.4.1. Anonim (Kimliği Doğrulanmamış) Kullanıcılar İçin Kısıtlamalar

Kimlik bilgilerini kullanmadan API'ye erişen kullanıcılar anonimdir. Anonim kullanıcılar için kısıtlamalar şunlardır:

- Anonim kullanıcılar, yalnızca en son durum vektörlerini alabilir, yani zaman parametresi yok sayılacaktır.
- Anonim kullanıcılar, yalnızca 10 saniyelik bir zaman aralığına sahip verileri alabilir. Bu API'nin zaman için durum vektörlerini $[now - (now \bmod 10)]$ döndüreceği anlamına gelir.

1.2.3.4.2. OpenSky Kullanıcıları İçin Kısıtlamalar

OpenSky kullanıcısı, API'ye erişmek için geçerli bir OpenSky hesabı kullanan kişilere denir. OpenSky kullanıcılar için kısıtlamalar şunlardır:

- OpenSky kullanıcıları geçmişte 1 saate kadar verileri alabilir. Zaman parametresinin bir değeri varsa $[t < now \text{ (şimdi)} - 3600]$ API, 400 bozuk istek döndürür.
- OpenSky kullanıcıları, 5 saniyelik bir zaman hassasiyeti ile verileri alabilir. Bunun anlamı eğer zaman parametresi olarak ayarlanmışsa `t` API zaman için $[t - (t \bmod 5)]$ durum vektörlerini döndürür.

1.3. OpenSky Network İçin Raspberry Pi İle ADS-B Baz İstasyonu

OpenSky gibi siteler uçak verilerini yakalayabilmek için baz istasyonlarına ihtiyaç duyar. ADS-B (Automatic Dependent Surveillance-Broadcast) uçak verilerini yakalayabilmek için kullanılan bir baz istasyonu sistemidir.

1.3.1. Tanıtım

Bu yardımcı kılavuz, Raspberry Pi ve hazır parçalar kullanılarak basit bir ADS-B baz istasyonunun nasıl işlevsel hale getirileceğini ve istasyonunuzu OpenSky ağına nasıl bağlayacağınızı açıklar.

Yapılacak olan baz istasyonu, 200-300 km'ye kadar yarıçaptaki uçaklardan yayın sinyallerini alabilecek ve kodunu çözebilecek kadar etkili olabilmektedir.

SON VERİLERİN ÇEKİLİP ÇÖZÜLMÜŞ HALİNİN RESİM GELECEK

Aynı zamanda verilerinizi [OpenSky Network](#) ve diğer ağlara aktarılabilir ve [PlanePlotter](#) gibi yerel uygulamalar üzerinden de kullanılabilir.

OPENSKY ÜZERİNDEN ANKET GÖRÜŞ ALANI RESMİN GELECEK

Anlatılan yöntem, Raspberry Pi kılavuzu için dump1090 kullanan ADS-B'ye dayanmaktadır.

İnternet üzerinde veri aktarımı sağlamak için internet bağlantısı olan bir ortama ihtiyaç vardır.

1.3.2. Önemli Detaylar

- Anketinin yerleştirileceği konum, gökyüzünü net görebilecek ve etrafı açık bir şekilde konumlandırılmış olması gerekmektedir.
- Topladığınız verileri OpenSky Network'üne göndermek istiyorsanız internet bağlantınızı yapıp veri iletmeniz gerekmektedir.
- OpenSky Network, baz istasyonunuza bağlamak için statik (sabit) bir ana bilgisayar adına veya IP adresine ihtiyaç duyar. Statik bir IP (Internet Protocol) adresiniz yoksa (genellikle olmaz), internet sağlayıcınızın [DuckDNS.org](https://duckdns.org) veya [No-IP.com](https://no-ip.com) gibi bir sağlayıcı kullanarak dinamik DNS'yi (Domain Name System) desteklemesi gerekir.
- Raspberry Pi'yi internete kablosuz olarak bağlayabilmeniz için modeminizin WIFI'yı (Wireless Fidelity: Kablosuz Ağ Bağlantısı) desteklemesi gerekir.

1.3.3. Baz İstasyonu İçin Gerekli Donanımlar

KENDİ KULLANDIĞIMIZ ALETLERİM RESMİ GELECEK

AŞAĞIDA YAZILAN EKİPMANLAR LİSTESİ DÜZENLENECEK...

● Raspberry Pi Ekipmanları:

(1) Raspberry Pi 4

(2) Hafıza Kartı

...

● Alıcının Ekipmanları:

(3) ADS-B Anten

(4) N Erkek Kablo

...

Bunlara ek olarak hazır microSD veya SD kart okuyuculu bir bilgisayarınız, HDMI (High-Definition Multimedia Interface) bağlantılı monitörünüz ve kurulum için gereken USB (Universal Serial Bus) tuş takımı ve fareniz olduğunu ve ayrıca sipariş vermeniz gerekmeyeceğini varsayıyoruz.

1.3.4. Genel Kurulum Adımları

- Raspberry Pi'nin birleştirilmesi
- Ekipmanların birleştirilmesi
- Raspberry'nin ayarlanması
 - ◆ İşletim sistemi ile SD kartın hazırlanması
 - ◆ İşletim sisteminin kurulumunun gerçekleştirilmesi
 - ◆ RTL-SDR alıcısı için sürücüler oluşturma ve yükleme
 - ◆ dump1090 kod çözücünün oluşturulması ve kurulumu
- Baz istasyonunu OpenSky Network'e bağlama
 - ◆ Raspberry Pi için dinamik (sabit olmayan) DNS kurulumu
 - ◆ Port (bağlantı noktası) 30005 bağlantı noktasını internete erişebilir hale getirilmesi
 - ◆ OpenSky Network hesabının oluşturulması
 - ◆ OpenSky Network'e yeni bir alıcı eklenmesi

1.3.4.1. Raspberry Pi'nin birleştirilmesi

1.3.4.2. Ekipmanların birleştirilmesi

1.3.4.3. Raspberry'nin ayarlanması

1.3.4.3. 1. İşletim sistemi ile SD kartın hazırlanması

1.3.4.3. 2. İşletim sisteminin kurulumunun gerçekleştirilmesi

1.3.4.3. 3. RTL-SDR alıcısı için sürücüler oluşturma ve yükleme

1.3.4.3. 4. dump1090 kod çözücünün oluşturulması ve kurulumu

1.3.4.4. Baz istasyonunu OpenSky Network'e bağlama

1.3.4.4. 1. Raspberry Pi için dinamik (sabit olmayan) DNS kurulumu

1.3.4.4. 2. Port (bağlantı noktası) 30005'i internete erişebilir hale getirilmesi

1.3.4.4. 3. OpenSky Network hesabının oluşturulması

1.3.4.4. 4. OpenSky Network’e yeni bir alıcı eklenmesi

Uygulamalar:

```
""" Kütüphaneler """
import requests
import pandas as pd
import numpy as np
from bokeh.plotting import figure
from bokeh.tile_providers import get_provider, OSM
from bokeh.io import output_notebook, show
from bokeh.models import ColumnDataSource

""" Kodun Notebook üzerinde çıktı vermesi için """
output_notebook()

def modify_doc(doc):

    """ İlk figürlerin oluşmasından önce hata vermemesi için verileri null
    ata """
    flight_source = ColumnDataSource({
        'icao24':[], 'callsign':[], 'origin_country':[],
        'time_position':[], 'last_contact':[], 'long':[], 'lat':[],
        'baro_altitude':[], 'on_ground':[], 'velocity':[], 'true_track':[],
        'vertical_rate':[], 'sensors':[], 'geo_altitude':[], 'squawk':[], 'spi':
    [],
        'position_source':[], 'MercatorX':[], 'MercatorY':[], 'rot_angle':
    [], 'url_data':[] })

    def update():
        """ Gelen coğrafi koordinatları web mercator dönüştür """
        def wgs84_to_web_mercator(df, lon="long", lat="lat"):
            k = 6378137
            df["MercatorX"] = df[lon] * (k * np.pi/180.0)
            df["MercatorY"] = np.log(np.tan((90 + df[lat]) * np.pi/360.0)) *
k
            return df

        """ Türkiye'yi içine alan coğrafi koordinatlar """
        lon_min, lat_min = 25, 35
        lon_max, lat_max = 45, 45
        """ REST API üzerinden veri çekmek için gerekli argümanlar """
        user_name = ''
        password = ''
        url_data =
'https://' + user_name + ':' + password + '@opensky-network.org/api/states/
all?' + 'lamin=' + str(lat_min) + '&lomin=' + str(lon_min) + '&lamax=' + str(lat_max)
+ '&lomax=' + str(lon_max)

        """ REST API üzerinden istekte bulun """
        response = requests.get(url_data).json()
        #print(response)
        #print(type(response))
```

```

        """ Gelen verinin sütun ismi """
        col_name =
['icao24','callsign','origin_country','time_position','last_contact','long','
lat','baro_altitude','on_ground','velocity','true_track','vertical_rate','sen
sors','geo_altitude','squawk','spi','position_source']
        """ Verileri DataFrame tipine çevir """
        flight_df = pd.DataFrame(response['states'])
        #print(flight_df)
        #print(type(flight_df))
        """ Sütun isimlerini numaralandır """
        flight_df = flight_df.loc[:,0:16]
        #print(flight_df)
        #print(type(flight_df))
        """ Sütun numaraları yerine col_name kısmındaki isimlendirmeleri ile
değiştir """
        flight_df.columns = col_name
        #print(flight_df.columns)
        #print(type(flight_df.columns))
        """ NAN tipinde gelen boş alanları hata almamak amaçlı No Data yap
flight_df"""
        flight_df = flight_df.fillna('No Data') #replace NAN with No Data
        #print(flight_df)
        #print(type(flight_df))

        """ Fonsiyonu çağır """
        wgs84_to_web_mercator(flight_df)
        flight_df['rot_angle'] = flight_df['true_track']* -1
        icon_url = 'https://cdn-icons-png.flaticon.com/512/1679/1679938.png'
        flight_df['url_data'] = icon_url

        """ Verilerin güncelleme işlemini yap """
        n_roll = len(flight_df.index)
        flight_source.stream(flight_df.to_dict(orient="list"),n_roll)

doc.add_periodic_callback(update, 5000)

        """ Nerelere şekil çizileceğine dair bilgileri gir """
        p = figure(plot_width=900, plot_height=700, x_range=(3000000, 5000000),
y_range=(3500000, 6000000),
                x_axis_type="mercator", y_axis_type="mercator",
tooltips=[("Ülke", "@origin_country"),
                ("Uçak Adı", "@callsign"), ("(Long, Lat)", "@long, @lat)"]],
title = "Anlık Uçak Konumları")

        """ Uçak figürlerini çizdir """
        p.image_url(url='url_data', x='MercatorX', y='MercatorY',
source=flight_source, anchor='center', angle_units='deg', angle='rot_angle',
h_units='screen', w_units='screen', w=25, h=25)

```

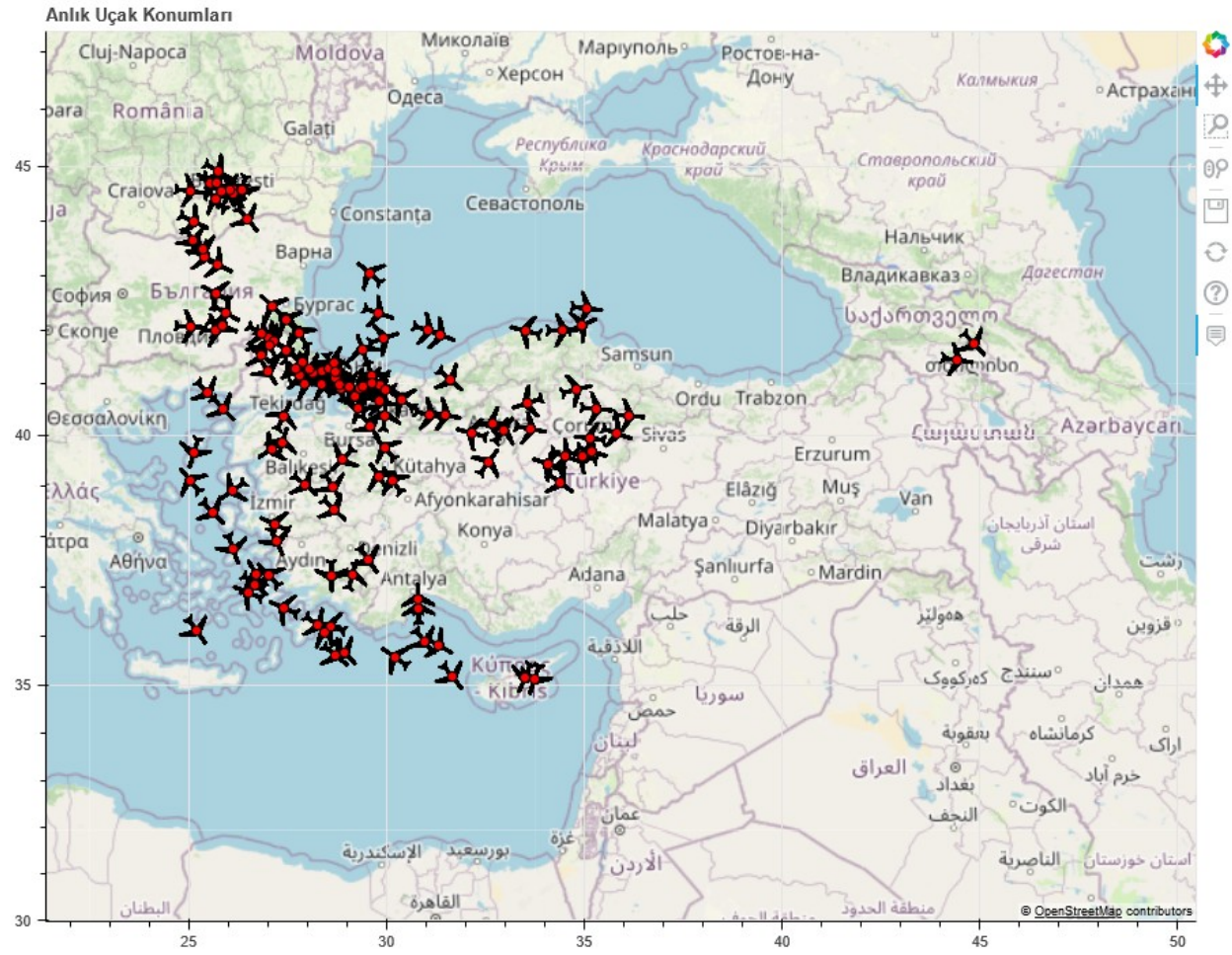
```
""" Çizilecek şekilleri ayarla """
p.circle(x="MercatorX", y="MercatorY", size=7, fill_color="red",
line_color="black", fill_alpha=1, source=flight_source)

""" Harita altlığını çağır """
tile_provider = get_provider(OSM)
""" Harita altlığını p şekilleri ile birleştir """
p.add_tile(tile_provider)

doc.add_root(p)

""" Haritayı göster """
show(modify_doc)
```

Sonuçlar:



KAYNAKLAR:

İNTERNET KAYNAKLARI:

https://en.wikipedia.org/wiki/History_of_Python

[https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language))

<https://docs.python.org/3/license.html>

<https://wiki.python.org/moin/>

<https://www.python.org/doc/essays/blurb/>

<https://opensky-network.org/index.php>

<https://openskynetwork.github.io/opensky-api/rest.html>

<https://www.redhat.com/en/topics/api/what-is-a-rest-api>

<https://openskynetwork.github.io/opensky-api/rest.html>

<https://github.com/openskynetwork/raspberry-pi-adsb>

EKLER

ÖZGEÇMİŞ