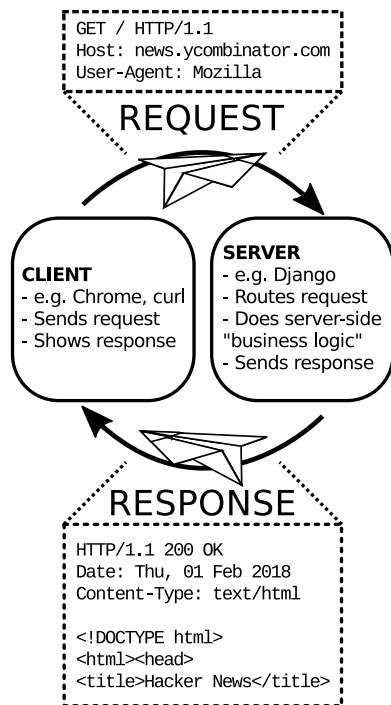


## REQUESTS &amp; RESPONSES



## TERMINOLOGY

**Protocol** An agreed-upon standard or set of "formalities" of how computers should talk to each other and exchange data

**IP address** Internet Protocol address, a "phone number" for computers

**Domain name** E.g. `google.com`, a mnemonic that's turns into the IP address of a server

**TCP** The way computers "call each other up" and send data to each other over the internet (*transmission control protocol*)

**HTTP** A protocol where a *client* uses TCP to connect and send a *request*, containing a particular *method* and *path* to a *server*, which in turn replies with a *response*.

**Headers & body** Requests and responses both are split into extra info (headers), and data (body).

**Method** The first text in the request, conventionally is one of 4 words in all caps. The backend gets to decide how to process different HTTP methods differently.

## HTTP METHODS

**GET** Fetch information without modifying anything

**POST** Create a new item

**PUT** Update an existing item

**DELETE** Delete an item

## PYTHON REQUESTS

```
import requests

url = "https://api.openaq.org" +
      "/v1/measurements" +
      "?location=Oakland"
resp = requests.get(url)
d = resp.json()
print(d["results"][0]["value"])
```

## DJANGO ROUTING

```
from django.urls import path
from django.http import HttpResponseRedirect
```

```
def hi_world(request):
    return HttpResponseRedirect(
        <h1>Hello Django World!</h1>
    )
```

```
urlpatterns = [
    path("hello-world/", hi_world),
]
```

## API TERMINOLOGY

**API** Application programming interface: The interface that software uses to communicate with other software. Some APIs are free, others might cost money.

**REST API** The most common type of API, REST APIs are based on HTTP, utilizing the different request methods to allow third party software to connect over the internet.

**API Key** A unique identifier for a user of an API (like a username).

**API Secret** Some APIs require this, basically a password for an API.

## HEROKU

```
# Test Procfile locally
pipenv shell
heroku local
```

```
# Create a new app
heroku create
```

```
# Ensure git is hooked up
git remote -v
```

```
# Deploy to Heroku via git
git push heroku master
```

```
# Check site in browser
heroku open
```

```
# Debug (view remote logs)
heroku logs
```

```
# Inspect environment variables
heroku config
```

```
# SSH into remote Heroku bash
heroku ps:exec
```

## ADVANCED PYTHON

**Try / except exceptions** Handle or ignore errors.

```
a = [1, 2]
try:
    lucky_13 = a[13]
except Exception as e:
    print("It broke:", e)
```

**List comprehension** Like a for-loop, but also creates a new list

```
names = ["John", "Paul", "G"]
long_names = [
    n.lower() for n in names
    if len(n) > 2
]
# = ["john", "paul"]
```

**Unpacking assignment** Can assign to two or more at once, in both loops and elsewhere

```
pairs = [(10, 5), (8, 100)]
for left, right in pairs:
    print(left * right)
x, y = [35, 15]
```

**Variable length arguments** Can provide "catch-alls" for positional and named arguments.

```
def do_all(*args, **kwargs):
    print(args, kwargs)
    return sum(args)
do_all(3, 5, b=3)
```