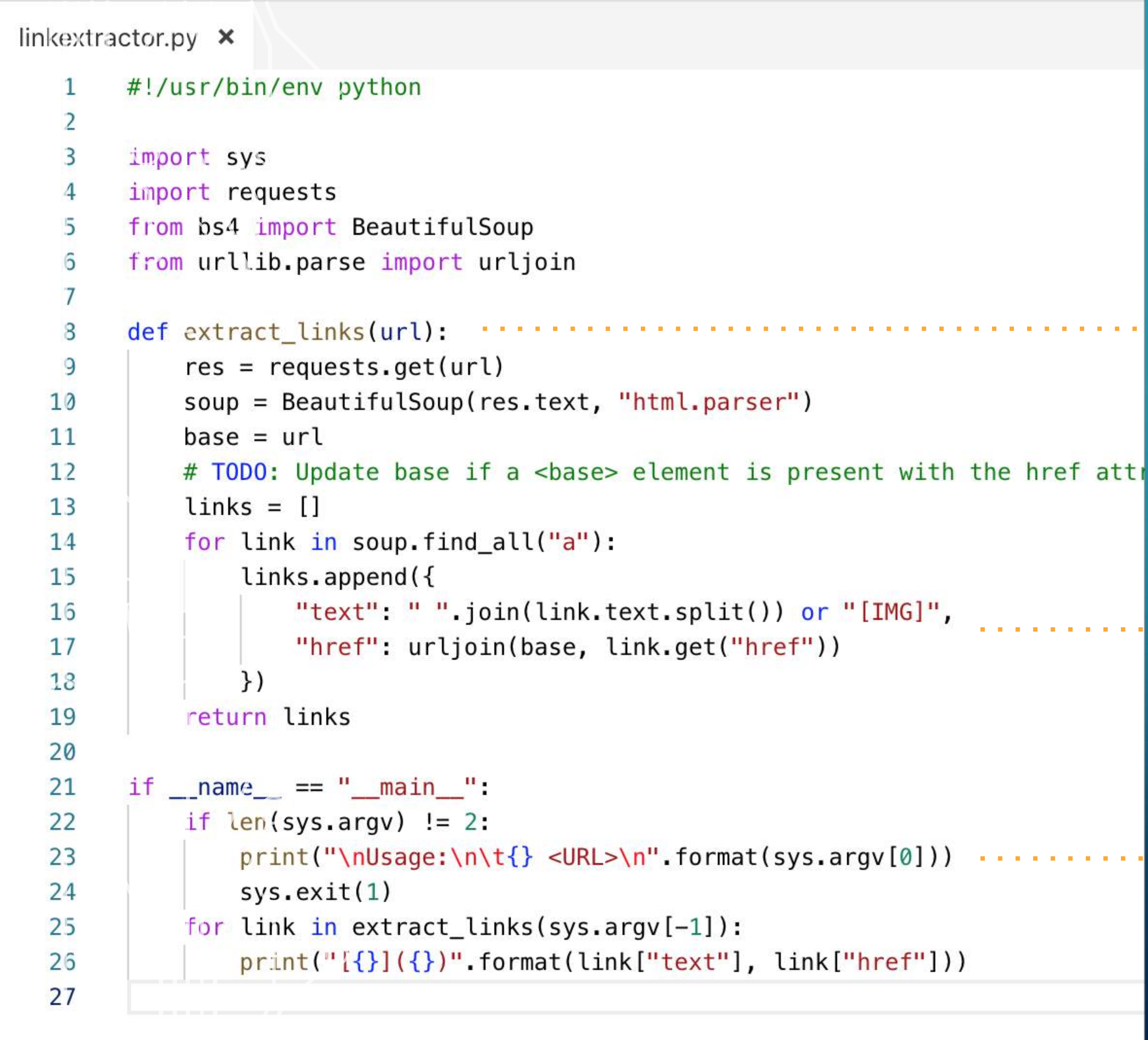


linkextractor.py ×

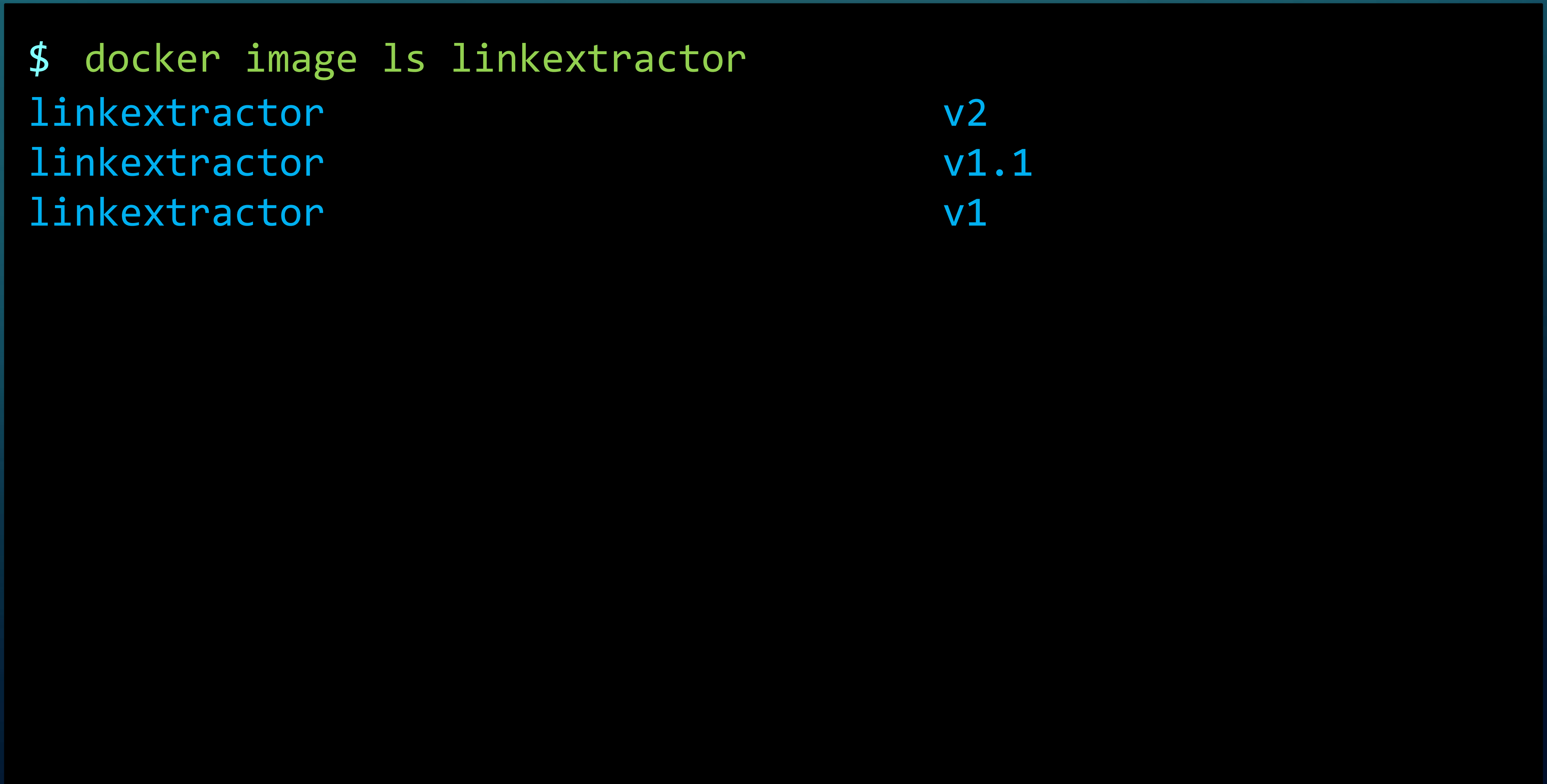
```python
#!/usr/bin/env python

import sys
import requests
from bs4 import BeautifulSoup
from urllib.parse import urljoin

def extract_links(url):
    res = requests.get(url)
    soup = BeautifulSoup(res.text, "html.parser")
    base = url
    # TODO: Update base if a <base> element is present with the href attr
    links = []
    for link in soup.find_all("a"):
        links.append({
            "text": " ".join(link.text.split()) or "[IMG]",
            "href": urljoin(base, link.get("href"))
        })
    return links

if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("\nUsage:\n\t{} <URL>\n".format(sys.argv[0]))
        sys.exit(1)
    for link in extract_links(sys.argv[-1]):
        print("[{}]({})".format(link["text"], link["href"]))

```

Modular
!

Highly
informative!

Usage
hints!

```
$  docker build -t linkextractor:v2 .

$  docker run --rm linkextractor:v2 https://docker.com
[[IMG]](https://docker.com/dockercon)
[Watch live >](https://docker.com/dockercon/register-livestream)
[[IMG]](https://docker.com/)
[Why Docker?](https://docker.com/why-docker)
[What is a Container?](https://docker.com/resources/what-
container)
[Company](https://docker.com/company)
[Partners](https://docker.com/partners)
[Products](https://docker.com/products)
[Docker Enterprise](https://docker.com/products/docker-
enterprise)
[Docker Hub](https://docker.com/products/docker-hub)
```



```
$  docker image ls linkextractor
linkextractor                          v2
linkextractor                          v1.1
linkextractor                          v1
```

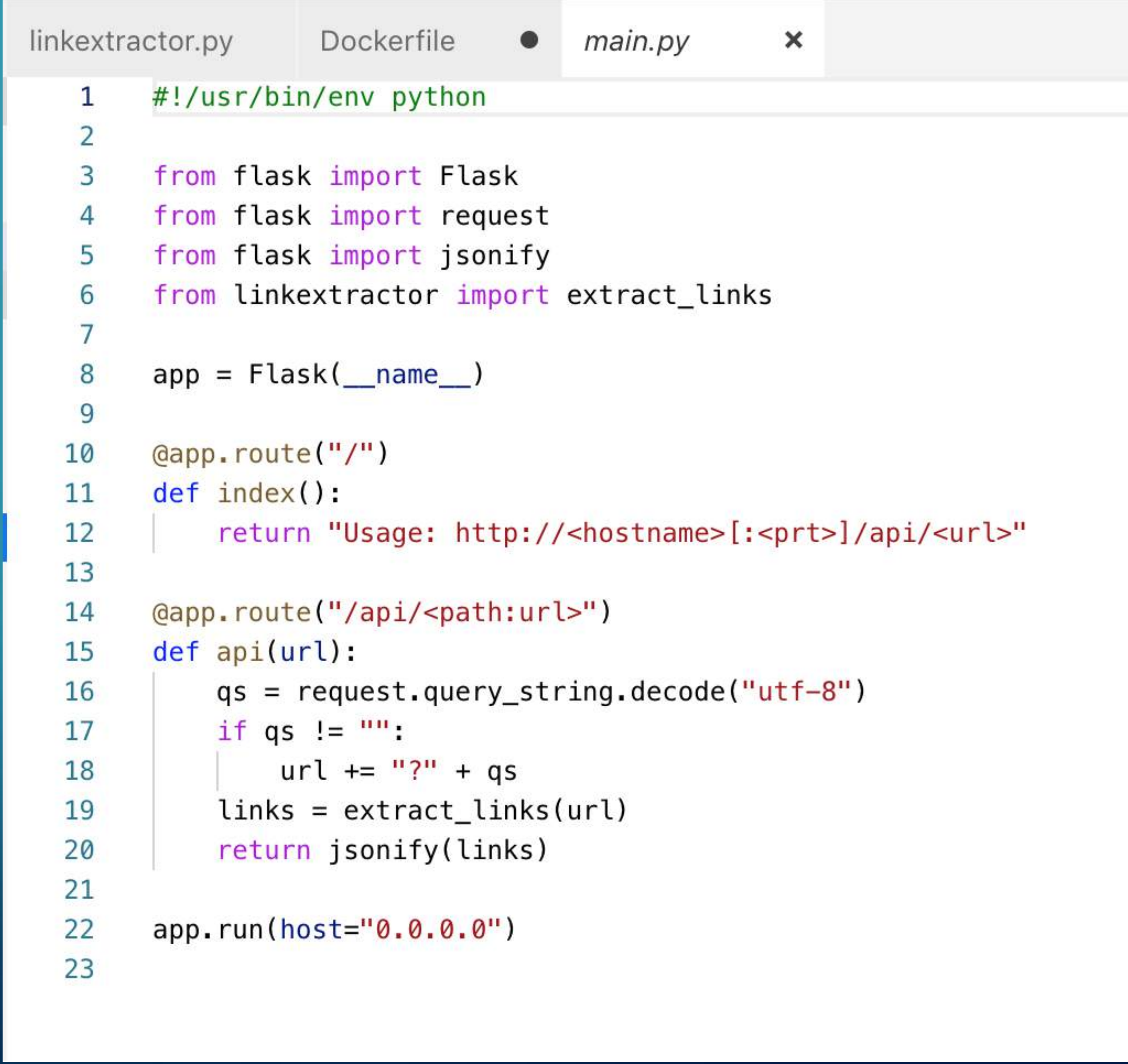
LINKEXTRACTOR ROADMAP

TURN IT INTO AN API SERVICE INSTEAD OF A ONE-OFF COMMAND

PROFIT!

1. "MAGICALLY" CODE MAIN.PY

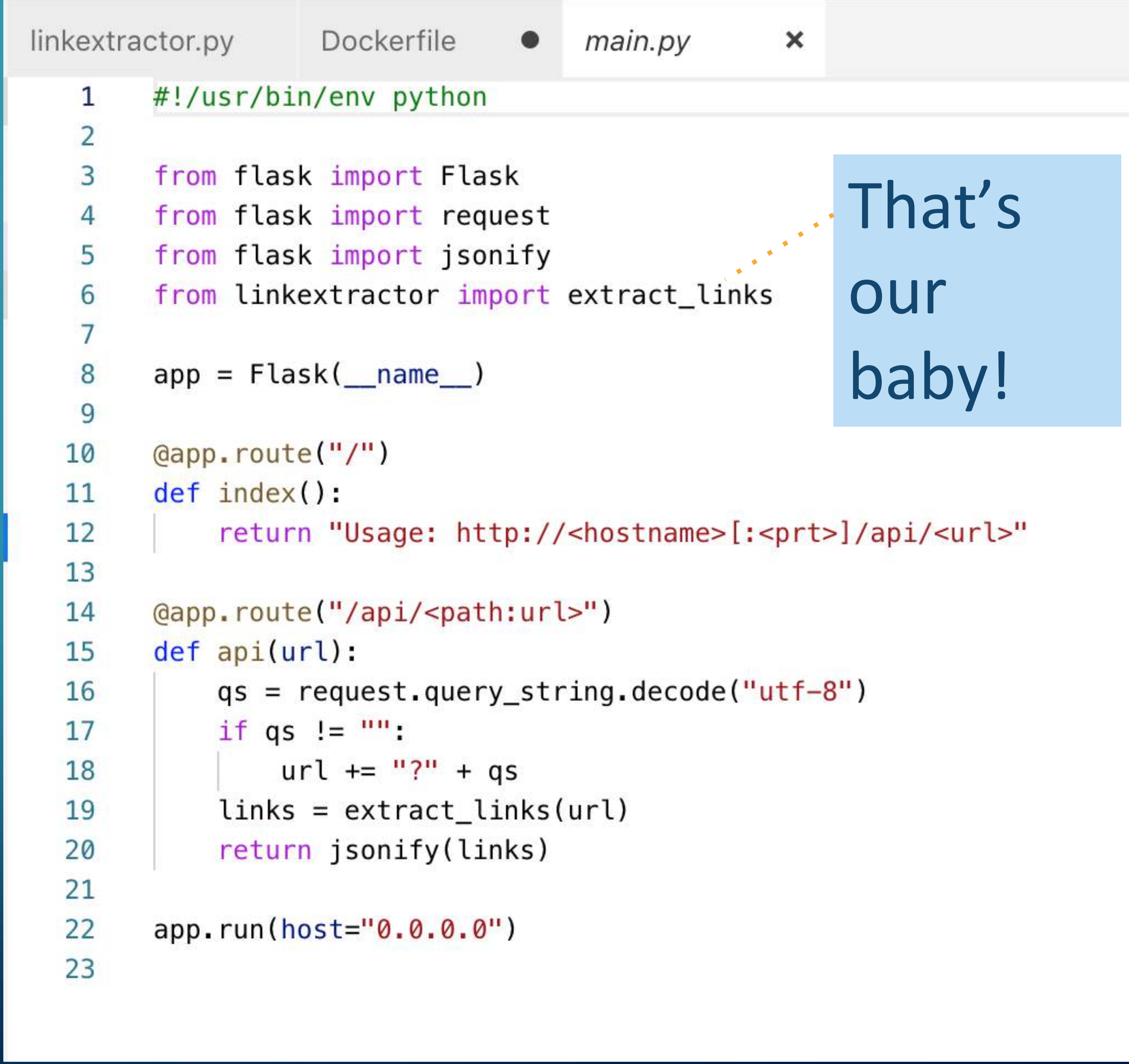
```
$ cp cheats/main.py .
```



```python
#!/usr/bin/env python

from flask import Flask
from flask import request
from flask import jsonify
from linkextractor import extract_links

app = Flask(__name__)

@app.route("/")
def index():
    return "Usage: http://<hostname>[:<prt>]/api/<url>"

@app.route("/api/<path:url>")
def api(url):
    qs = request.query_string.decode("utf-8")
    if qs != "":
        url += "?" + qs
    links = extract_links(url)
    return jsonify(links)

app.run(host="0.0.0.0")
```

1. "MAGICALLY" CODE MAIN.PY

```
$ cp cheats/main.py .
```



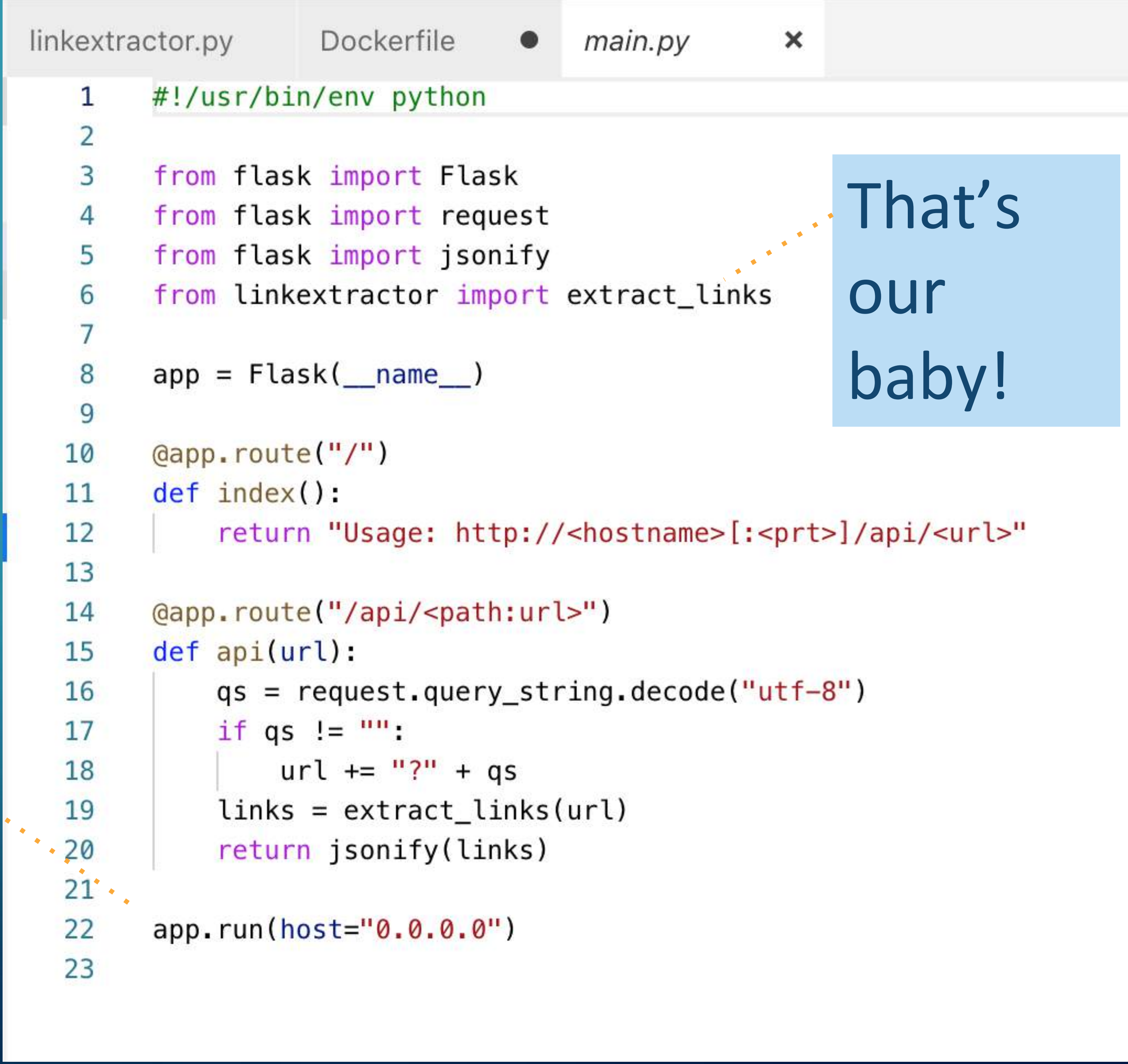
```python
#!/usr/bin/env python

from flask import Flask
from flask import request
from flask import jsonify
from linkextractor import extract_links

app = Flask(__name__)

@app.route("/")
def index():
    return "Usage: http://<hostname>[:<prt>]/api/<url>"

@app.route("/api/<path:url>")
def api(url):
    qs = request.query_string.decode("utf-8")
    if qs != "":
        url += "?" + qs
    links = extract_links(url)
    return jsonify(links)

app.run(host="0.0.0.0")
```

That's our baby!

”MAGICALLY” CODE MAIN.PY

```
$ cp cheats/main.py .
```

“Go ahead caller, I’m listening.”



```python
#!/usr/bin/env python

from flask import Flask
from flask import request
from flask import jsonify
from linkextractor import extract_links

app = Flask(__name__)

@app.route("/")
def index():
    return "Usage: http://<hostname>[:<prt>]/api/<url>"

@app.route("/api/<path:url>")
def api(url):
    qs = request.query_string.decode("utf-8")
    if qs != "":
        url += "?" + qs
    links = extract_links(url)
    return jsonify(links)

app.run(host="0.0.0.0")
```

That’s our baby!

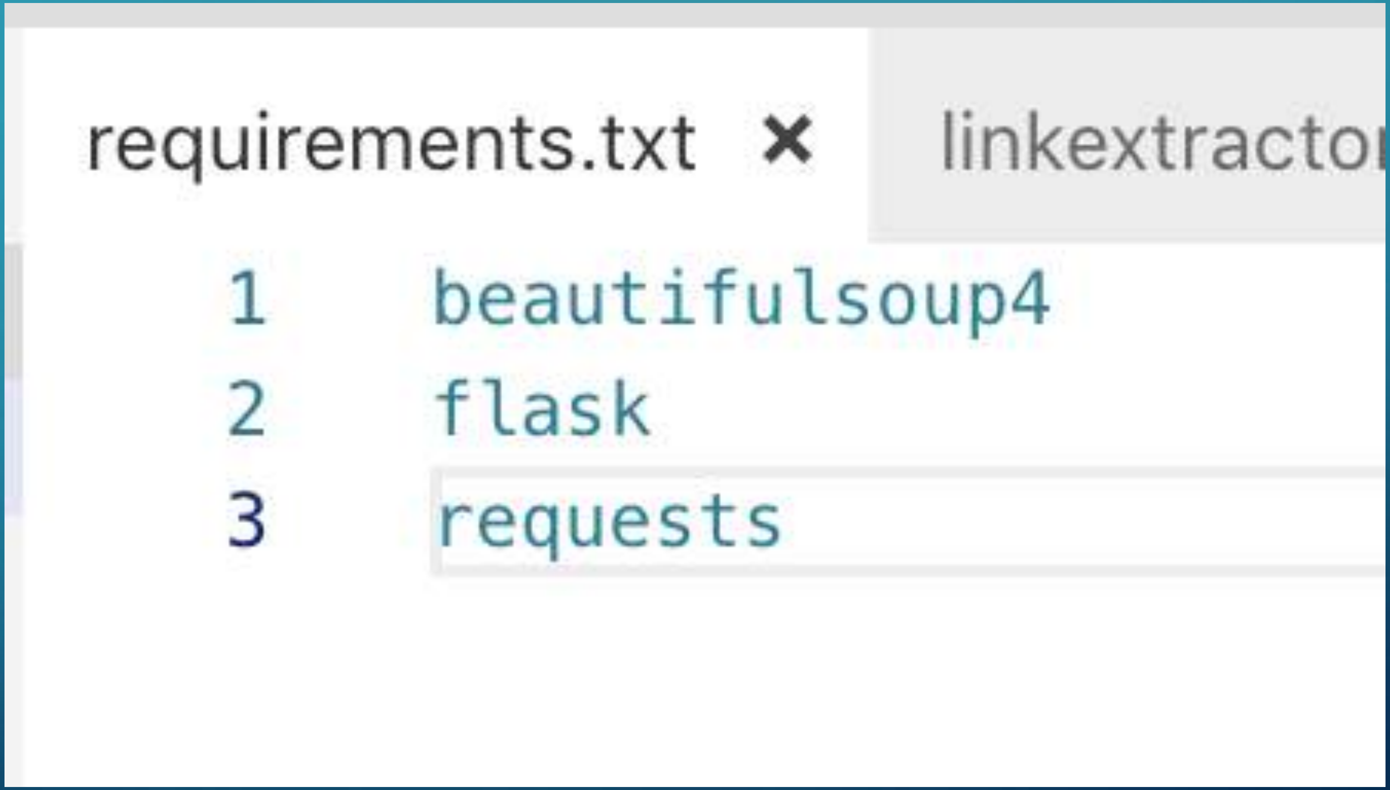
OPTIMIZE OUR DOCKERFILE

```
FROM python:3
LABEL maintainer="<your name>"

RUN pip install beautifulsoup4
RUN pip install requests

WORKDIR /app
COPY linkextractor.py /app/
RUN chmod a+x linkextractor.py

ENTRYPOINT ["./linkextractor.py"]
```

Eliminate extra manual install steps

- Reduce layers
- Maintain dependencies the way we normally do in our language of choice

2. PYTHON REQUIREMENTS FILE



```
beautifulsoup4
flask
requests
```

OPTIMIZE OUR DOCKERFILE

```
FROM python:3
LABEL maintainer="<your name>"

RUN pip install beautifulsoup4
RUN pip install requests

WORKDIR /app
COPY linkextractor.py /app/
RUN chmod a+x linkextractor.py

ENTRYPOINT ["./linkextractor.py"]
```

Copy all our code in at once

OPTIMIZE OUR DOCKERFILE

```
FROM python:3
LABEL maintainer="<your name>"

RUN pip install beautifulsoup4
RUN pip install requests

WORKDIR /app
COPY linkextractor.py /app/
RUN chmod a+x linkextractor.py

ENTRYPOINT ["./linkextractor.py"]
```

We no longer want a single, one-off command…we want a service

OPTIMIZE OUR DOCKERFILE

```
FROM python:3
LABEL maintainer="<your name>"

WORKDIR /app
COPY requirements.txt /app/
RUN pip install -r requirements.txt

COPY *.py /app/
RUN chmod a+x *.py

CMD ["./main.py"]
```

Note the switch from ENTRYPOINT to CMD

- ENTRYPOINT - a command to run every time
  - good for executables
- CMD - a *default* command for our container to run but it can be substituted

LINKEXTRACTOR V3 - API-IFIED

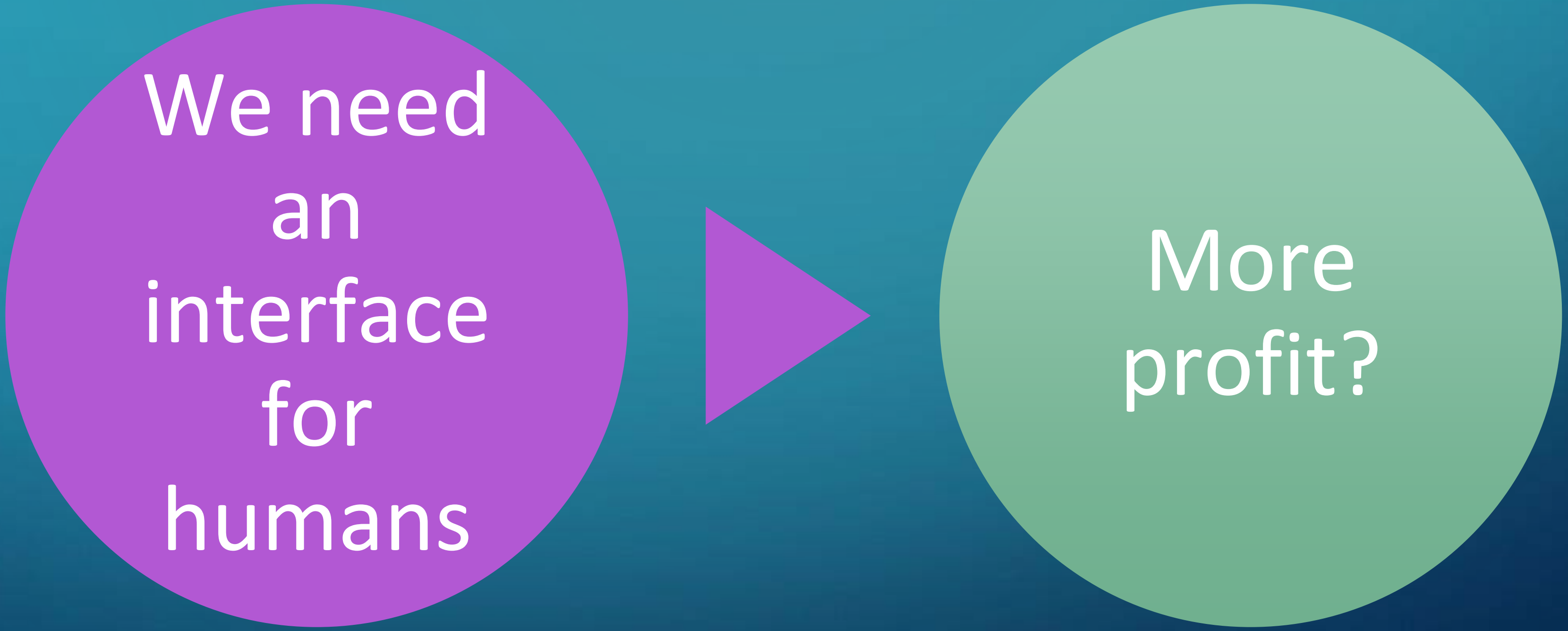
```
$  docker build -t linkextractor:v3 .

$  docker run -d -p 5000:5000 --name=linkextractor
   linkextractor:v3

$  curl localhost:5000
Usage: http://<hostname>[:<prt>]/api/<url>

$  curl localhost:5000/api/http://docker.com
[{"href":"http://docker.com/dockercon","text":"[IMG]"},{"href":"h
ttp://docker.com/dockercon/register-livestream","text":"Watch
live
>"},{"href":"http://docker.com/","text":"[IMG]"},{"href":"http://
docker.com/why-docker","text":"Why
Docker?"},{"href":"http://docker.com/resources/what-
container","text":"What is a
```

DUDE, WHERE ARE MY LOGS?

```
$ docker container logs linkextractor
 * Serving Flask app "main" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production
environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
172.17.0.1 - - [01/May/2019 09:14:23] "GET / HTTP/1.1" 200 -
172.17.0.1 - - [01/May/2019 09:14:37] "GET
/api/http://example.com HTTP/1.1" 200 -
172.17.0.1 - - [01/May/2019 09:16:33] "GET /api/http://docker.com
HTTP/1.1" 200 -
```

```
$  docker container rm -f linkextractor
```

NEW ROADMAP!



TIME WARP

```
$ git checkout step4 -f
```