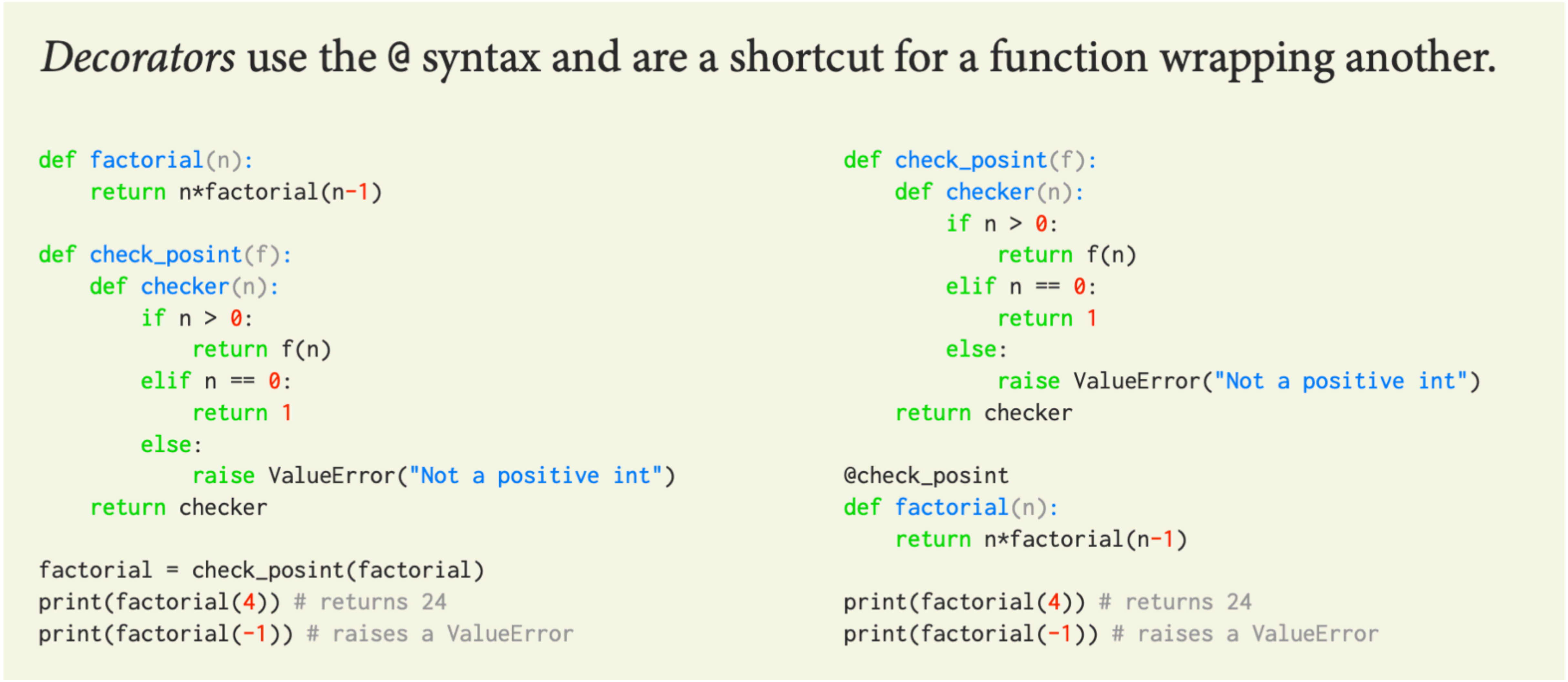
HTTP and API

Rahul Dave, Hult University

Decorators



Decorators use the @ syntax and are a shortcut for a function wrapping another.

```python
def factorial(n):
    return n*factorial(n-1)

def check_posint(f):
    def checker(n):
        if n > 0:
            return f(n)
        elif n == 0:
            return 1
        else:
            raise ValueError("Not a positive int")
    return checker

factorial = check_posint(factorial)
print(factorial(4)) # returns 24
print(factorial(-1)) # raises a ValueError
```

```python
def check_posint(f):
    def checker(n):
        if n > 0:
            return f(n)
        elif n == 0:
            return 1
        else:
            raise ValueError("Not a positive int")
    return checker

@check_posint
def factorial(n):
    return n*factorial(n-1)

print(factorial(4)) # returns 24
print(factorial(-1)) # raises a ValueError
```

Decorators as business value creator

- Decorators exist for just about any activity. You can run functions on specific machines using metaflow/prefect/dagster.
- You can get system observability with tools like datadog
- You can cache your intermediate function results on disk or in memory
- You can create logs to send to log-files for data analysis
- I use it to track my machine learning runs for clients.

- Anytime you want to wrap a function with another function to do “orthogonal” work, use a decorator.

Lets use Decorators

https://github.com/hult-cm3-rahul/mysite

https://www.pythonanywhere.com/

HTML

- angle brackets
- should be in pairs, eg `<p>Hello</p>`
- maybe in implicit pairs, such as `<br/>`

```
<!DOCTYPE html>
<html>
  <head>
    <title>Title</title>
  </head>
  <body>
    <h1>Body Title</h1>
    <p>Body Content</p>
  </body>
</html>
```

Developer Tools

- ctrl/cmd shift i in chrome
- cmd-option-i in safari
- look for "inspect element"
- locate details of tags

Web Servers

- A server is a long running process (also called daemon) which listens on a pre-specified port
- and responds to a request, which is sent using a protocol called HTTP
- A browser must first we must parse the url. Everything after a # is a fragment. Until then its the DNS name or ip address, followed by the URL.

```
http://localhost:8888/Documents/ml-1/
BLA.ipynb#something
```

- protocol is `http`, hostname is `localhost`, port is `8888`
- url is `/Documents/ml-1/BLA.ipynb`
- url fragment is `#something

Request is sent to localhost on port 8888. It says:

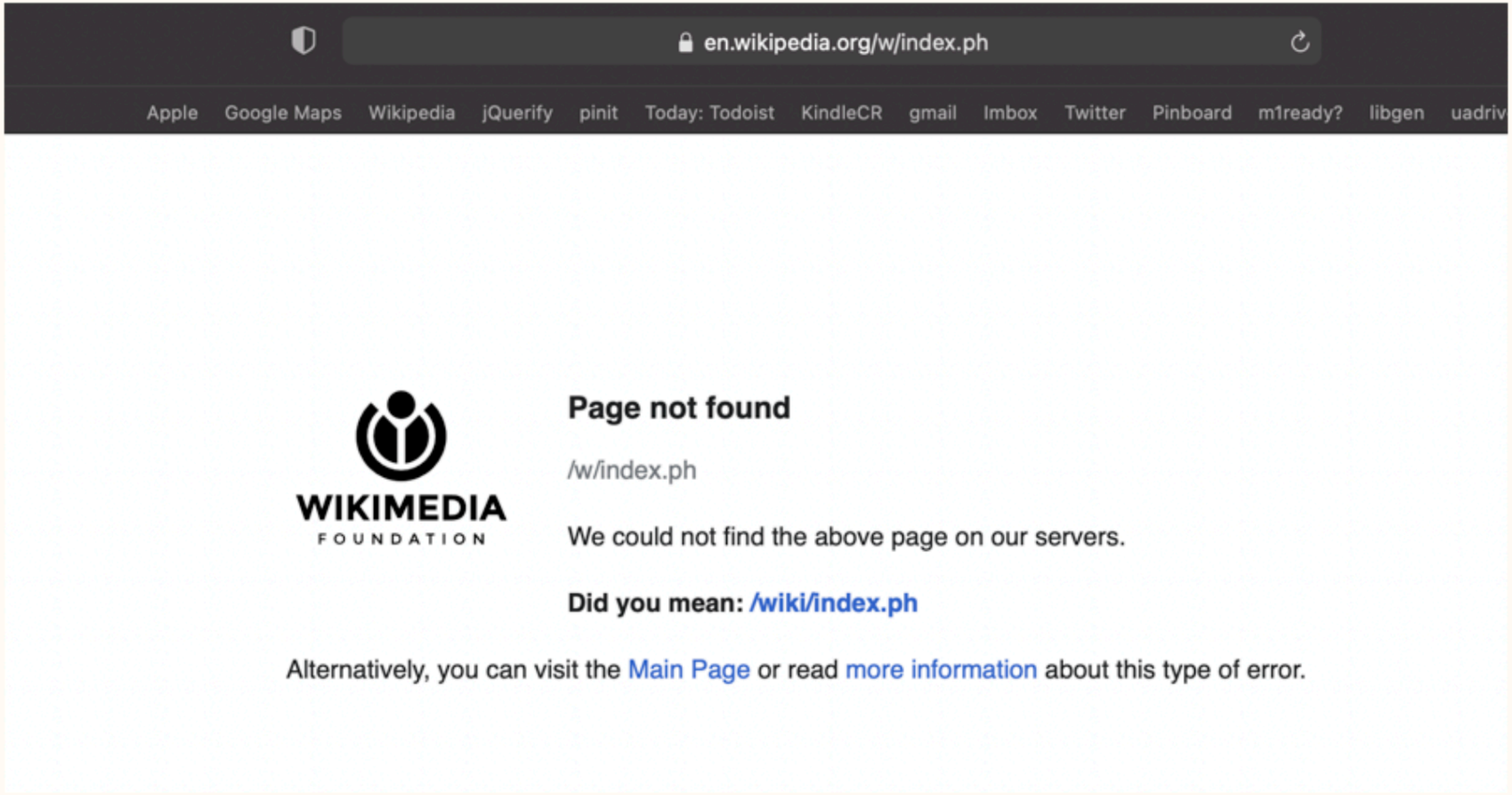
```
Request:
GET /Documents/ml-1/BLA.ipynb HTTP/1.0
```

Example with Response: Google

```
GET / HTTP/1.0
Host: www.google.com
```

```
HTTP/1.0 200 OK
Date: Mon, 14 Nov 2016 04:49:02 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
P3P: CP="This is ..."
Server: gws
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN
Set-Cookie: NID=90=gb5q7b0...; expires=Tue, 16-May-2017 04:49:02 GMT; path=/; domain=.google.com; HttpOnly
Accept-Ranges: none
Vary: Accept-Encoding

<!doctype html><html itemscope=""
itemtype="http://schema.org/WebPage" lang="en">
<head><meta content="Search the world's information,
```



Page not found

/w/index.ph

We could not find the above page on our servers.

Did you mean: /wiki/index.ph

Alternatively, you can visit the Main Page or read more information about this type of error.

HTTP Status Codes[1]

- 200 OK:
  Means that the server did whatever the client wanted it to, and all is well.
- 201 Created:
  The request has been fulfilled and resulted in a new resource being created. The newly created resource can be referenced by the URI(s) returned in the entity of the response, with the most specific URI for the resource given by a Location header field.
- 400: Bad request
  The request sent by the client didn't have the correct syntax.
- 401: Unauthorized
  Means that the client is not allowed to access the resource. This may change if the client retries with an authorization header.
- 403: Forbidden
  The client is not allowed to access the resource and authorization will not help.
- 404: Not found
  Seen this one before? :) It means that the server has not heard of the resource and has no further clues as to what the client should do about it. In other words: dead link.
- 500: Internal server error
  Something went wrong inside the server.
- 501: Not implemented
  The request method is not supported by the server.

[1] (from http://www.garshol.priv.no/download/text/http-tut.htm)

APIs

A lot of web sites provide APIs for data to be accessed in a programmatic way.

APIs are specific URL endpoints, different from the standard web endpoints, where information about an object (say, me, on github) is provided in a structured way

- The advantage of this is structured, specifically, `JSON`, output.
- Json looks like a dictionary and can be used directly in your program.
- Often you need special keys to access APIs

API Example

```
APISTART="https://api.github.com/"
resp_api = requests.get(APISTART+"users/rahuldave")
resp_api
```

```
<Response [200]>
```

```
resp_api.text
```

```
'{"login":"rahuldave","id":43227,"node_id":"MDQ6VXNlcjQzMjI3",
"avatar_url":"https://avatars3.githubusercontent.com/u/43227?
v=4",
"gravatar_id":"","url":"https://api.github.com/users/rahuldave",
"html_url":"https://github.com/rahuldave",
"followers_url":"https://api.github.com/users/rahuldave/followers",
"following_url":"https://api.github.com/users/rahuldave/following{/other_user}",
"gists_url":"https://api.github.com/users/rahuldave/gists{/gist_id}",
"starred_url":"https://api.github.com/users/rahuldave/starred{/owner}{/repo}",
"subscriptions_url":"https://api.github.com/users/rahuldave/subscriptions",
"organizations_url":"https://api.github.com/users/rahuldave/orgs",
"repos_url":"https://api.github.com/users/rahuldave/repos",
"events_url":"https://api.github.com/users/rahuldave/events{/privacy}",
"received_events_url":"https://api.github.com/users/rahuldave/received_events",
"type":"User","site_admin":false,"name":"Rahul Dave",
"company":"Harvard University/univ.ai","blog":"https://univ.ai",
"location":"Somerville, MA","email":null,"hireable":null,"bio":null,
"twitter_username":null,"public_repos":119,"public_gists":9,
"followers":317,"following":1,
"created_at":"2008-12-29T23:34:27Z","updated_at":"2021-01-19T20:51:21Z"}'
```

Python Conversion

```
resp_api.json()
```

```
{'login': 'rahuldave',
 'id': 43227,
 'node_id': 'MDQ6VXNlcjQzMjI3',
 'avatar_url': 'https://avatars3.githubusercontent.com/u/43227?v=4',
 'gravatar_id': '',
 'url': 'https://api.github.com/users/rahuldave',
 'html_url': 'https://github.com/rahuldave',
 'html_url': 'https://github.com/rahuldave',
 'followers_url': 'https://api.github.com/users/rahuldave/followers',
 'following_url': 'https://api.github.com/users/rahuldave/following{/other_user}',
 'gists_url': 'https://api.github.com/users/rahuldave/gists{/gist_id}',
 'starred_url': 'https://api.github.com/users/rahuldave/starred{/owner}{/repo}',
 'subscriptions_url': 'https://api.github.com/users/rahuldave/subscriptions',
 'organizations_url': 'https://api.github.com/users/rahuldave/orgs',
 'repos_url': 'https://api.github.com/users/rahuldave/repos',
 'events_url': 'https://api.github.com/users/rahuldave/events{/privacy}',
 'received_events_url': 'https://api.github.com/users/rahuldave/received_events',
 'type': 'User',
 'site_admin': False,
 'name': 'Rahul Dave',
 'company': 'Harvard University/univ.ai',
 'blog': 'https://univ.ai',
 'location': 'Somerville, MA',
 'email': None,
 'hireable': None,
 'bio': None,
 'twitter_username': None,
 'public_repos': 119,
 'public_gists': 9,
 'followers': 317,
 'following': 1,
 'created_at': '2008-12-29T23:34:27Z',
 'updated_at': '2021-01-19T20:51:21Z'}
```