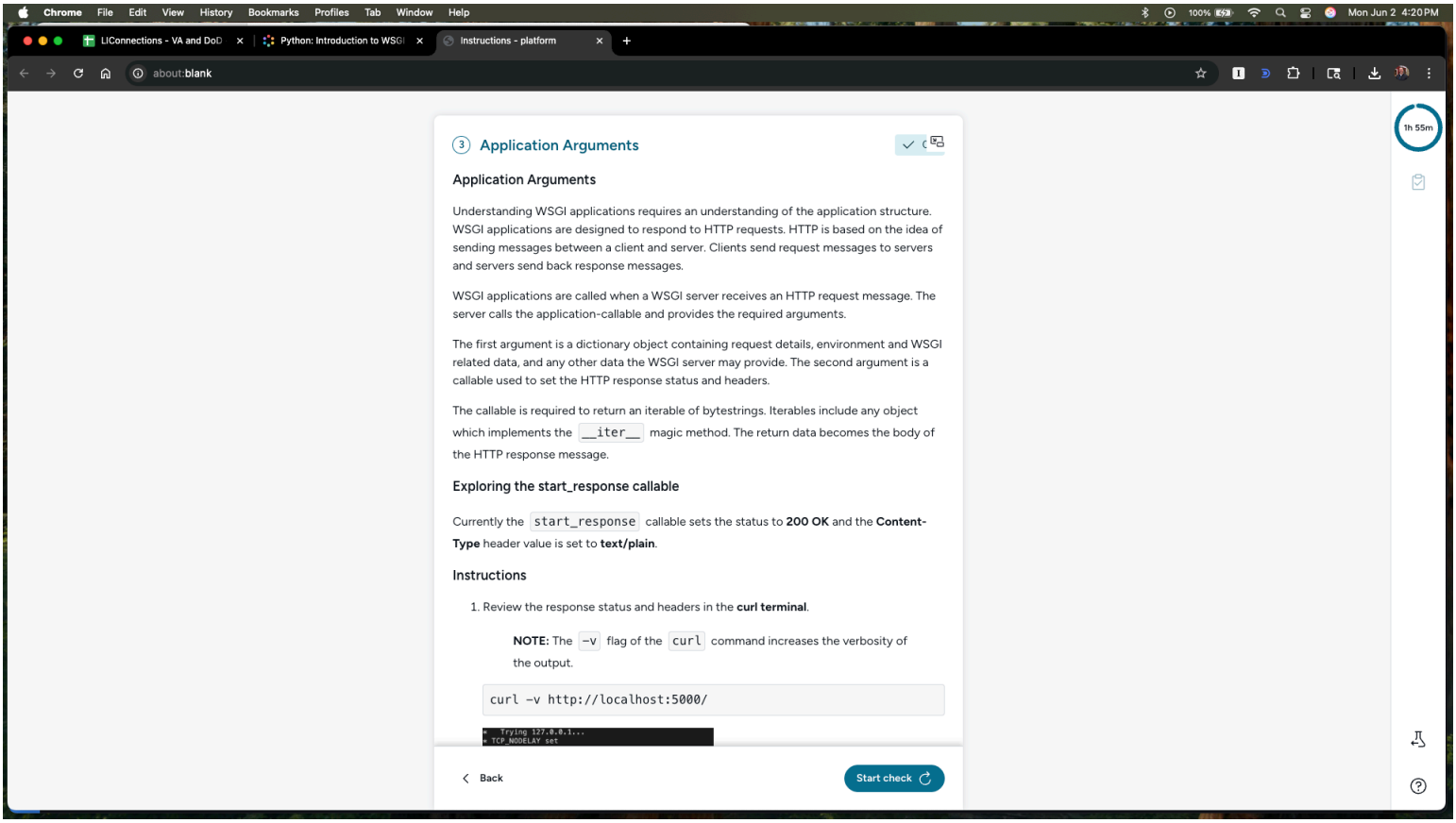

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon Jun 2 4:20 PM
LIConnections - VA and DoD
Python: Introduction to WSGI
Instructions - platform
about:blank
1h 55m
3 Application Arguments
Application Arguments
Understanding WSGI applications requires an understanding of the application structure. WSGI applications are designed to respond to HTTP requests. HTTP is based on the idea of sending messages between a client and server. Clients send request messages to servers and servers send back response messages.
WSGI applications are called when a WSGI server receives an HTTP request message. The server calls the application-callable and provides the required arguments.
The first argument is a dictionary object containing request details, environment and WSGI related data, and any other data the WSGI server may provide. The second argument is a callable used to set the HTTP response status and headers.
The callable is required to return an iterable of bytestrings. Iterables include any object which implements the __iter__ magic method. The return data becomes the body of the HTTP response message.
Exploring the start_response callable
Currently the start_response callable sets the status to 200 OK and the Content-Type header value is set to text/plain.
Instructions
1. Review the response status and headers in the curl terminal.
NOTE: The -v flag of the curl command increases the verbosity of the output.
curl -v http://localhost:5000/
* Trying 127.0.0.1...
* TCP_NODELAY set
Back
Start check

```
curl -v http://localhost:5000/
```

```
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 5000 (#0)
> GET / HTTP/1.1
> Host: localhost:5000
> User-Agent: curl/7.52.1
> Accept: */*
>
* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK                                   HTTP STATUS
< Date: Tue, 21 Jun 2022 21:18:01 GMT
< Server: WSGIServer/0.2 CPython/3.9.9
< Content-Type: text/plain                          CONTENT-TYPE HEADER
<
Hello, WSGI!
* Curl_http_done: called premature == 0
* Closing connection 0
```

2. Edit the arguments for the `start_response` callable.

```
1       start_response('500 Server error', [('Content-Type', 'te
```

```
# A basic WSGI application.
def app(environ, start_response):
    # Begin by sending the HTTP(S) client the reponse status and headers.
    # Web applications support a wide variety of content types such as html, images, plain text, etc.
    # Because the demonstrations in this lab are consumed using the console, plain text is used.
    start_response('500 Server error', [('Content-Type', 'text/plain'), ('Host', 'cloudacademy.com')])
    # Yield a bytestring representing the response body.
    # The yield keyword turns this function into a generator
    # making the function an iterable.
    # The b in front of the open quote makes this a bytestring.
    yield b'Hello, WSGI!\n'
```

A running WSGI server will require a restart before any code changes will take effect.

3. Stop the WSGI server process.

    1. Click anywhere inside the **WSGI terminal** pane to set focus.
    2. Press `CTRL+C` to interrupt the process.

```
^CTraceback (most recent call last):
  File "/home/project/cloudacademy/playground.py", line 24, in <module>
    server.serve_forever()
  File "/usr/local/lib/python3.9/socketserver.py", line 232, in serve_forever
    ready = selector.select(poll_interval)
  File "/usr/local/lib/python3.9/selectors.py", line 416, in select
    fd_event_list = self._selector.poll(timeout)
KeyboardInterrupt
```

4. Start the WSGI server process.

Back

Start check

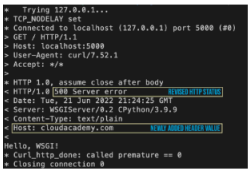
1h 55m

4. Start the WSGI server process.

```
python3 cloudacademy/playground.py
```

5. Review the updated response status and headers using the **curl terminal**.

```
curl -v http://localhost:5000/
```



The `start_response` callable can only be called once per response. Attempts to call it again will result in an exception.

Exploring the WSGI environment variables

The first argument is a dictionary object containing WSGI environment variables. PEP 3333 defines a short list of required variables. These variables are used to describe an HTTP request message. They include details such as: HTTP request method (GET, POST, PUT, etc), URL path, URL query string, etc.

Instructions

1. Append the following code to the bottom of the `app` callable.

Back

Start check

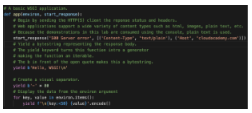
1h 55m

Instructions

1. Append the following code to the bottom of the `app` callable.

   NOTE: Ensure the indentation is correct after pasting.

```
# Create a visual separator.
yield b'-' * 80
# Display the data from the environ argument
for key, value in environ.items():
    yield f'\n{key:<50} {value}'.encode()
```



2. Restart the WSGI server process.

   1. Click anywhere inside the **WSGI terminal** pane to set focus.
   2. Press `CTRL+C` to interrupt the process.
   3. Start the WSGI server process.

      ```
      python3 cloudacademy/playground.py
      ```

   WSGI servers are encouraged to include any additional data which might be useful to the WSGI application. The reference implementation includes OS environment variables and some WSGI server settings. Each WSGI server implementation should include a list of all provided variables.

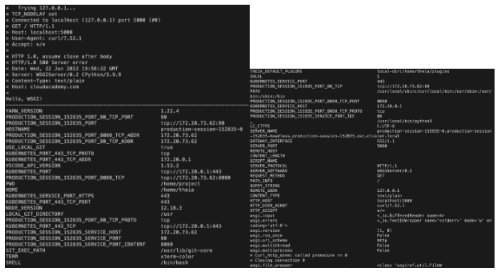
3. Review the updated response body using the **curl terminal**.

Back

Start check

3. Review the updated response body using the **curl terminal**.

```
curl -v http://localhost:5000/
```



⭐ Proceed to the next step ⭐

Validations

- ◯ **On Track - Start Response Callable Arguments**
  Ensure the arguments provided to the `start_response` callable are correct.

- ◯ **On Track - Environment Variables Iterated Over**
  Ensure `environ.items()` is called in the `app` callable.

Back

Start check

1h 55m