Building an HTTP Server - Guide with Commands

This document provides a brief guide on building a simple HTTP server using Python along with commonly used commands. This is particularly useful for educational purposes, local testing, and learning about server-client interactions.

Tools Required

- Python (version 3.x)
- Command-line interface (Terminal or CMD)
- Web browser

Creating a Basic HTTP Server Using Python 3

To quickly start a basic HTTP server using Python, use the following command:

```
python3 -m http.server
```

This will start a server on port 8000 by default. You can access it at: http://localhost:8000

Specifying a Port

To run the server on a different port (e.g., 8080), use:

```
python3 -m http.server 8080
```

Serving a Specific Directory

Navigate to the directory you want to serve and run the server from there:

```
cd /path/to/your/directory
python3 -m http.server
```

Stopping the Server

Press Ctrl + C in the terminal to stop the server.

Custom HTTP Server Using Python Code

You can write a custom server using Python’s built-in http.server module:

```
from http.server import SimpleHTTPRequestHandler, HTTPServer

hostName = "localhost"
serverPort = 8000

class MyServer(SimpleHTTPRequestHandler):
  def do_GET(self):
    self.send_response(200)
    self.send_header("Content-type", "text/html")
    self.end_headers()
    self.wfile.write(bytes("<html><body><h1>My Custom HTTP
Server</h1></body></html>", "utf-8"))

if __name__ == "__main__":
  webServer = HTTPServer((hostName, serverPort), MyServer)
  print("Server started http://%s:%s" % (hostName, serverPort))
  try:
    webServer.serve_forever()
  except KeyboardInterrupt:
    pass
  webServer.server_close()
  print("Server stopped.")
```