How Tos

Useful tips, tricks and practical instructions

- BPMN: Logging using simple http push logger

BPMN: Logging using simple http push logger

If you don't have access to LAPIS server terminal to monitor the server's log files, you may use the following to perform realtime logging over HTTP directly from the groovy scripts:

1. Put the following on top of the groovy script you want to log and change `LOGSERVER` to point to the logging server (see below):

```
import groovy.json.JsonOutput
import java.net.HttpURLConnection
import java.net.URL
import java.text.SimpleDateFormat



// Change to actual log server
final LOGSERVER = 'http://192.168.50.32:8081/log'



class HttpPushLogger {
  String endpointUrl

  HttpPushLogger(String endpointUrl) {
    this.endpointUrl = endpointUrl
  }



  void log(String message) {
    def dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss")
    def payload = [timestamp: dateFormat.format(new Date()), message: message]
    def jsonPayload = JsonOutput.toJson(payload)

    URL url = new URL(endpointUrl)
    HttpURLConnection connection = (HttpURLConnection) url.openConnection()
    connection.setRequestMethod("POST")
    connection.setRequestProperty("Content-Type", "application/json")
    connection.setDoOutput(true)
```

```
    connection.outputStream.withWriter("UTF-8") { writer ->
      writer.write(jsonPayload)
    }

    int responseCode = connection.responseCode
    // println "Response Code: $responseCode"
  }
}

def logger = new HttpPushLogger(LOGSERVER)
```

2. Use it as following:

```
logger.log('This is a test log message.')
```

3. In a machine to which LAPIS can connect through http on some port, create a new `'log-server.py'` file and put the following script in it:

```
import argparse
from http.server import BaseHTTPRequestHandler, HTTPServer
import json


class SimpleHTTPRequestHandler(BaseHTTPRequestHandler):
  def do_POST(self):
    content_length = int(self.headers['Content-Length'])
    post_data = self.rfile.read(content_length)
    log_message = json.loads(post_data)

    # print(f"Received log message: {log_message}")
    print(log_message['timestamp'],": ", log_message['message'])

    self.send_response(200)
    self.send_header('Content-type', 'application/json')
    self.end_headers()
    response = {'status': 'success'}
    self.wfile.write(json.dumps(response).encode('utf-8'))
```

```
  def log_message(self, format, *args):
    return


def run(server_class=HTTPServer, port=8081):
  server_address = ('', port)
  httpd = server_class(server_address, SimpleHTTPRequestHandler)
  print(f'Starting server on port {port}...')
  httpd.serve_forever()


if __name__ == '__main__':
  parser = argparse.ArgumentParser(description='Simple HTTP server for logging')
  parser.add_argument('--port', type=int, default=8081, help='Port to run the server on
(default: 8081)')
  args = parser.parse_args()
  run(port=args.port)
```

4. Run it with (you need to have python installed of course):

```
python3 log-server.py
```