Spring Boot WebHook Handler

Requirements

- `docker` and `docker-compose` installed
- Images to use:

  ```
  docker pull alpine
  docker pull nginx:stable-alpine
  ```

Spring Boot

1. Go to https://start.spring.io
2. Dependencies: ***Web***, ***Lombok***, ***Actuator***
3. Important classes:

src/main/java/com/webhook/external/SystemExecution.java

```
package com.webhook.external;

import lombok.extern.log4j.Log4j2;

import java.io.File;
import java.io.IOException;
import java.util.concurrent.Executors;

@Log4j2
public abstract class SystemExecution<T extends SystemCommonProperties> {

    public void build(T project)  {
        try {
            ProcessBuilder processBuilder = new ProcessBuilder();

            processBuilder.command(project.getCommand());
            processBuilder.directory(new File(project.getPath()));

            Process process = processBuilder.start();

            if(project.getLogMethod() != LogMethod.NONE) {
                SystemStream systemStream = new SystemStream(process.
getInputStream(),
                        project.getLogMethod() == LogMethod.STD ? System.out:
:println : s -> log.info(s));
                Executors.newSingleThreadExecutor().submit(systemStream);
            }

            assert process.waitFor() == 0;
        }catch (InterruptedException  | IOException ex) {
            throw new RuntimeException(ex);
        }
    }
}
```

src/main/java/com/webhook/external/SystemCall.java

```java
package com.webhook.external;

import lombok.RequiredArgsConstructor;
import org.springframework.stereotype.Component;

import static java.util.Arrays.stream;

@RequiredArgsConstructor
@Component
public class SystemCall {

    private final SystemProject systemProject;
    private final SystemExecution systemExecution;

    public void build() {
        stream(systemProject.getProjects()).forEach(systemExecution::build);
    }
}
```

src/main/java/com/webhook/external/SystemProject.java

```java
package com.webhook.external;

public interface SystemProject<T extends SystemCommonProperties> {

    T[] getProjects();
}
```

src/main/java/com/webhook/external/SystemStream.java

```java
package com.webhook.external;

import lombok.AllArgsConstructor;

import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.function.Consumer;

@AllArgsConstructor
public class SystemStream implements Runnable{

    private InputStream inputStream;
    private Consumer<String> consumer;

    @Override
    public void run() {
        new BufferedReader(new InputStreamReader(inputStream)).lines()
                .forEach(consumer);
    }

}
```

src/main/java/com/webhook/external/SystemCommonProperties.java

```java
package com.webhook.external;

import lombok.Data;

@Data
public class SystemCommonProperties {

    private String name;
    private String path;
    private String[] command;
    private LogMethod logMethod = LogMethod.STD;

}

enum LogMethod {
    STD,LOG,NONE
}
```

4. `WebHookController` class that will accept requests from GitHub.

src/main/java/com/webhook/github/WebHookController.java

```
package com.webhook.github;

import com.webhook.external.SystemCall;
import lombok.AllArgsConstructor;
import lombok.extern.log4j.Log4j2;
import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.*;

@Log4j2
@AllArgsConstructor
@RequestMapping("/api/github")
@RestController
public class WebHookController {

    private SystemCall systemCall;

    @PostMapping("/event")
    @ResponseStatus(HttpStatus.OK)
    public void eventLog(@RequestBody WebHook webHook) {
       log.info("WebHook received: {}", webHook);
       systemCall.build();
    }

}
```

TIP Analyze the classes and take a look at the `Ansible` implementation.

Labs

- Go to the `docker-compose` folder and take a look at the instructions.