Tutorial Log4j2 com Spring Boot

Requisitos de configuração

Para esse tutorial sera usada uma inicialização padrão do spring boot com as dependências `spring-boot-starter-web` e `lombok`.

Para adicionarmos a dependência do Log4j2 devemos remover a dependência do log padrão do Spring Boot e adicionar a do Log4j2 no `pom.xml`:

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
    <exclusions>
        <exclusion>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-logging</artifactId>
        </exclusion>
    </exclusions>
</dependency>

<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-log4j2</artifactId>
</dependency>
```

Software Necessário

Para a criação desse tutorial foi utilizado a IDE IntelliJ IDEA, mas funciona para qualquer IDE java com suporte ao Maven.

Contexto

Logging

Logging é o processo de escrever informação em arquivos de log. Os arquivos de log incluem informações sobre diferentes eventos do sistema operacional, do software e de conexões.

Porque usar Logging

Um sistema de Logging é usado geralmente para as seguintes funções:

- Resoluções de Problemas / Debugging
- Coleta de Informações
- Auditorias
- Profiling

Um ponto a se destacar é que um sistema de Logging não precisa ser limitado a erros de desenvolvimento. Ele pode ser usado para identificar falhas de segurança, violações de politicas e também para encontrar gargalos na aplicação.

Onde usar Logs

Todos os eventos de erros de validação, autenticação e autorização, falhas, erros da aplicação, passagem por pontos criticos do sistema e inicializações e desligamentos da aplicação.

O que não pode estar em um Log

Não pode ser usado para log nenhum tipo de informação sensivel do usuário (senhas, ids, tokens de acesso) ou do sistema (strings de conexão ao DB, chavesde encriptação).

Boas praticas de Logging

- Deve ter sentido e contexto
- Deve ser estruturado e feito em diferentes níveis
- Deve ser ser adaptado para uso tanto nas fases de desenvolvimento como na de produção

Log4j2

O log4j2 é uma biblioteca do projeto Apache Software Foundation e tem 3 componentes principais: Loggers, Appenders e layouts. Os Loggers são os métodos que pegam as mensagens de log e enviam para os Appenders, os Appenders escrevem o Log no seu arquivo de destino e os layouts são usados para formatação das mensagens.

Uma das principais caracteristicas de um sistema de log são os níveis de erro e configuração:

- OFF: Sem Logging
- FATAL: Erros fatais da aplicação
- ERROR: Erros criticos da aplicação
- WARN: Erros provavelmente prejudiciais.
- INFO: Mensagens de Informação
- DEBUG: Pontos de Verificação de Debugging
- TRACE: Mensagens de debug mais detalhadas.
- ALL: Todo tipo de informação

Event Level	LoggerConfig Level						
	TRACE	DEBUG	INFO	WARN	ERROR	FATAL	OFF
ALL	YES	YES	YES	YES	YES	YES	NO
TRACE	YES	NO	NO	NO	NO	NO	NO
DEBUG	YES	YES	NO	NO	NO	NO	NO
INFO	YES	YES	YES	NO	NO	NO	NO
WARN	YES	YES	YES	YES	NO	NO	NO
ERROR	YES	YES	YES	YES	YES	NO	NO
FATAL	YES	YES	YES	YES	YES	YES	NO
OFF	NO	NO	NO	NO	NO	NO	NO

Uso Com Spring Boot

Passo 1

Alterar o `pom.xml` como está informado na primeira seção do tutorial

Passo 2

Adicionar as configurações do log4j2 na pasta resorces com o nome `log4j2-spring.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="WARN" monitorInterval="30">
    <Properties>
        <Property name="LOG_PATTERN">%d{yyyy-MM-dd'T'HH:mm:ss.SSSZ} %p
%m%n</Property>
        <Property name="APP_LOG_ROOT">c:/temp</Property>
    </Properties>
    <Appenders>
        <Console name="console" target="SYSTEM_OUT">
            <PatternLayout pattern="${LOG_PATTERN}" />
        </Console>

        <RollingFile name="file"
            fileName="${APP_LOG_ROOT}/SpringBoot2App/application.log"
            filePattern="${APP_LOG_ROOT}/SpringBoot2App/application-%d{yyyy-MM-
dd}-%i.log">
            <PatternLayout pattern="${LOG_PATTERN}" />
            <Policies>
                <SizeBasedTriggeringPolicy size="19500KB" />
            </Policies>
            <DefaultRolloverStrategy max="1" />
        </RollingFile>

    </Appenders>
    <Loggers>
        <Root level="info">
            <AppenderRef ref="console" />
            <AppenderRef ref="file" />
        </Root>
    </Loggers>
</Configuration>
```

Observe que em `<Root level="info">` é definida a configuração do nivel do log, e que os arquivos de log serão salvos na pasta `c:/temp`.

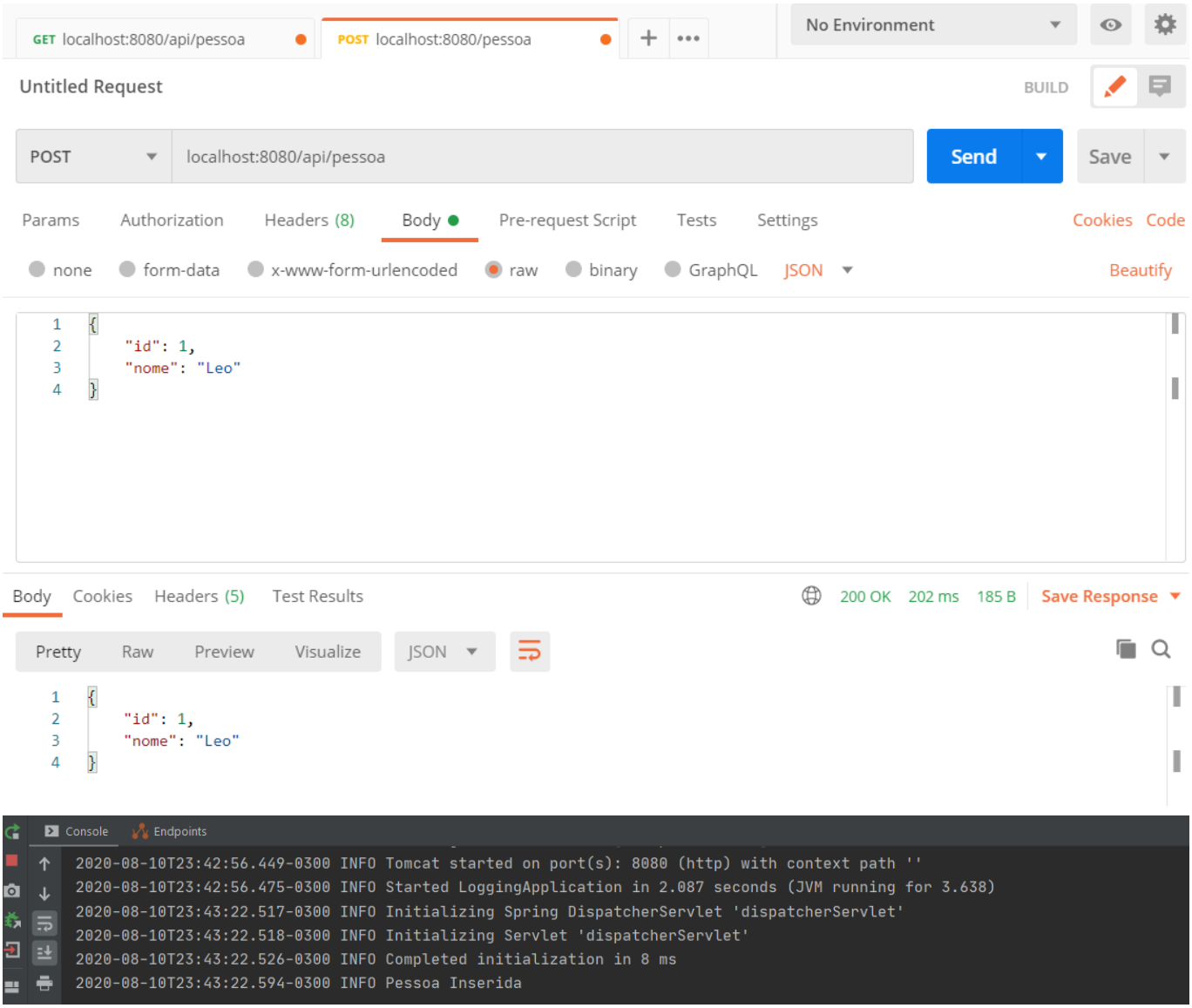
Exemplo

Usando uma api basica com a implementação de inserir uma pessoa, recuperar uma pessoa por ID, e recuperar todas as Pessoas. O log foi inserido na camada service:

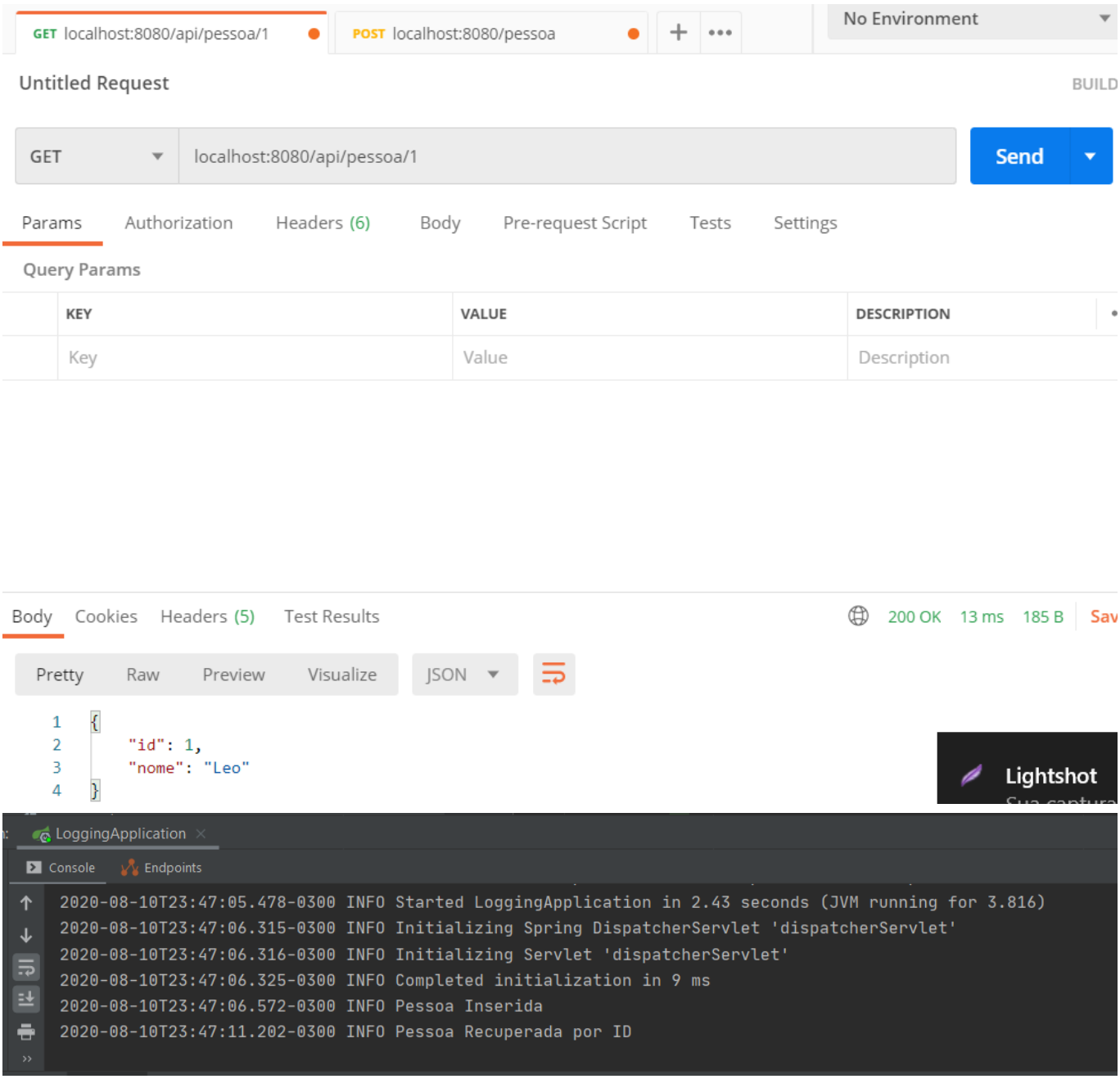
```java
package com.tutorial.logging.services.impl;

import com.tutorial.logging.model.Pessoa;
import com.tutorial.logging.services.PessoaService;
import lombok.extern.log4j.Log4j2;
import org.springframework.stereotype.Service;

import java.util.ArrayList;
import java.util.List;

@Service
@Log4j2
public class PessoaServiceImpl implements PessoaService {
    private static List<Pessoa> DB = new ArrayList<>();

    @Override
    public Pessoa add(Pessoa pessoa) {
        if (DB.stream().anyMatch(p -> p.getId() == pessoa.getId())){
            log.warn("Tentativa de inserção duplicada");
            return null;
        }

        log.info("Pessoa Inserida");
        DB.add(pessoa);
        return pessoa;
    }

    @Override
    public Pessoa getById(int id) {
        if (!DB.stream().anyMatch(p -> p.getId() == id)){
            log.warn("Tentativa de recuperação por ID inexistente");
            return null;
        }

        log.info("Pessoa Recuperada por ID");
        return DB.stream()
                .filter(p -> p.getId() == id)
                .findFirst().get();
    }

    @Override
    public List<Pessoa> getAll() {
        log.info("Todas As Pessoas Recuperadas");
        return DB;
    }
}
```

Aqui pode se observar o uso do `lombok` com a annotation `@Log4j2` que instancia a classe Logger no objeto log. Também deve se observar os niveis warn quando existe um problema no parametro, e info quando as requisições acontecem normalmente.

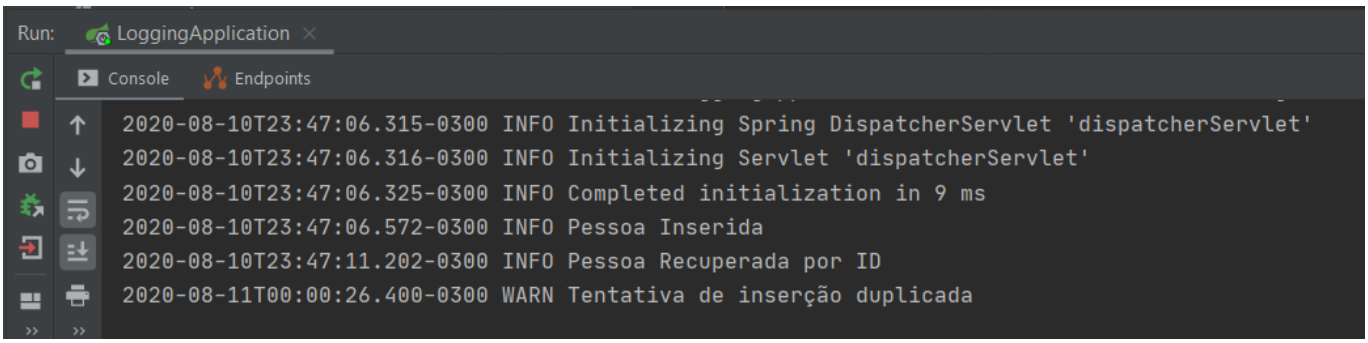
Inserção Pessoa



Recuperação Por ID



Tentativa de Inserção de pessoa com mesmo ID



Zip do Projeto: Tutorial Logging

Autor: Leonardo Giorgiani