### groovy

Groovy로 작성된 스크립트를 실행합니다.

#### 문법

groovy CLASS\_NAME

필수 매개변수

**CLASS\_NAME**

실행할 클래스의 이름.

#### 설명

Groovy는 Python, Ruby와 같은 언어의 영향을 받아 개발된 동적 객체 지향 언어로 JVM에서 동작합니다. 실행할 수 있는 스크립트 파일은 다음과 같은 제약 조건을 만족해야 합니다.

* 스크립트 파일 이름은 다음과 같은 형식으로 지정해야 합니다: CLASS\_NAME.groovy
* 로그프레소 설치 디렉터리 아래 **data/araqne-logdb-groovy/query\_scripts** 디렉터리에 있는 Groovy 스크립트만 실행할 수 있습니다.
* 로그프레소가 제공하는 패키지를 임포트해서 사용해야 합니다. 다음과 같은 패키지를 필요에 따라 사용하십시오.
* groovy.transform.CompileStatic (성능상 이점이 있으므로 권장)
* org.araqne.logdb.groovy.GroovyQueryScript (필수)
* org.araqne.logdb.QueryStopReason
* org.araqne.logdb.Row (필수)
* org.araqne.logdb.RowBatch
* org.araqne.logdb.RowPipe

Groovy 스크립트의 성능을 향상시키려면 다음과 같은 사항을 참고하십시오.

* 문자열 처리 메서드는 가능하면 사용하지 않도록 합니다. 문자열 객체가 많아지면 JVM에서 가비지 컬렉션이 빈번하게 일어납니다.
* split(), tokenize() 메서드 대신에 indexOf()이나 substring()을 사용하십시오. 코드는 길어지지만 더 좋은 처리 성능을 제공합니다.
* Pattern.compile()을 반복적으로 사용하지 마십시오. Matcher.reset()을 호출해 Matcher 인스턴스를 재사용하는 방식이 더 좋은 성능을 제공합니다.
* 예외 발생을 최소화하십시오.
* 예외가 빈번하게 발생하면 처리 성능이 현저하게 떨어집니다.
* 가능하다면, 발생할 수 있는 오류 케이스는 조건 검사를 통해 처리하십시오.

#### 사용 예

다음과 같은 스크립트를 **ToAscii.groovy**라는 이름으로 로그프레소 설치 디렉터리 아래 **data/araqne-logdb-groovy/query\_scripts** 디렉터리에 저장합니다.

import groovy.transform.CompileStatic;import org.araqne.logdb.Row;import org.araqne.logdb.groovy.GroovyQueryScript;@CompileStatic(groovy.transform.TypeCheckingMode.SKIP)class ToAscii extends GroovyQueryScript { def void onRow(Row row) { byte[] payload = row.get('payload') char[] chars = new char[payload.length]; for (int i = 0; i < payload.length; i++) { char c = (char) payload[i] if (c < 32 || c > 126) c = '.' chars[i] = c } row.put('text', new String(chars)) pipe.onRow(row) }}

이 스크립트는 PCAP 파일에서 디코딩되어 **payload** 필드에 출력된 바이너리 값 중에서 32번째 문자부터 127번째 문자를 ASCII 형식으로 인코딩해서 보여줍니다.

pcapfile /opt/logpresso/sonar/http-2.pcap | pcapdecode | groovy ToAscii