### groovy

Groovyで作成されたスクリプトを実行します。

#### 構文

groovy CLASS\_NAME

必須パラメータ

**CLASS\_NAME**

実行するクラス名。

#### 説明

Groovyは、PythonやRubyなどの言語の影響を受けて開発された動的オブジェクト指向言語であり、JVM上で動作します。実行可能なスクリプトファイルには、以下の制約があります。

* スクリプトファイル名は次の形式で指定する必要があります：CLASS\_NAME.groovy
* ログプレッソ・ソナーのインストールディレクトリ配下の **data/araqne-logdb-groovy/query\_scripts** ディレクトリにあるGroovyスクリプトのみ実行可能です。
* ログプレッソ・ソナーが提供するパッケージをインポートして使用する必要があります。必要に応じて、以下のパッケージを利用してください。
* groovy.transform.CompileStatic（パフォーマンス向上のため推奨）
* org.araqne.logdb.groovy.GroovyQueryScript（必須）
* org.araqne.logdb.QueryStopReason
* org.araqne.logdb.Row（必須）
* org.araqne.logdb.RowBatch
* org.araqne.logdb.RowPipe

Groovyスクリプトのパフォーマンスを向上させるため、以下の点にご注意ください。

* 文字列処理メソッドの使用は可能な限り避けてください。文字列オブジェクトが増えると、JVMでのガベージコレクションが頻繁に発生します。
* split() や tokenize() メソッドの代わりに、indexOf() や substring() を使用してください。コードは長くなりますが、より高い処理性能が得られます。
* Pattern.compile() を繰り返し使用しないでください。Matcher.reset() を呼び出して Matcher インスタンスを再利用する方法が、より良いパフォーマンスを提供します。
* 例外発生を最小限に抑えてください。
* 例外が頻繁に発生すると、処理性能が著しく低下します。
* 可能であれば、発生しうるエラーケースは条件判定によって処理してください。

#### 使用例

以下のようなスクリプトを **ToAscii.groovy** という名前で、ログプレッソ・ソナーのインストールディレクトリ配下の **data/araqne-logdb-groovy/query\_scripts** ディレクトリに保存します。

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) }}

このスクリプトは、PCAPファイルからデコードされ、**payload** フィールドに出力されたバイナリ値のうち、32番目から127番目までの文字をASCII形式でエンコードして表示します。

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