

# **Brekeke PBX**

**Version 2.1**

**ARS プラグイン開発ガイド**

**Brekeke Software, Inc.**

## バージョン

Brekeke PBX v2.1 ARS プラグイン開発ガイド, 2008 年 2 月

## 著作権

本書の著作権は、Brekeke Software, Inc. にあります。

Copyright ©2003-2008 Brekeke Software, Inc.

本書の一部または全部を、Brekeke Software, Inc. との書面による同意なしに、複製、複製、転載、多言語への翻訳、書き換え、あるいは、転送することは法律で禁じられています。

## 免責事項

Brekeke Software, Inc. は予告なしに本書の内容を変更する権利を有します。

## 登録商標

- ◆ Linux は Linus Torvalds 氏の米国及びその他の国における登録商標あるいは商標です。
- ◆ Red Hat は米国 Red Hat, Inc. の登録商標です。
- ◆ Windows は米国 Microsoft Corporation の米国及びその他の国における登録商標です。
- ◆ Mac は米国及びその他の国で登録されている Apple Computer, Inc の登録商標です。
- ◆ その他製品名と会社名は、一般にその会社の登録商標です。

---

1.	はじめに .....	3
2.	開発環境 .....	3
3.	作成の手順.....	3
4.	インストール・設定方法 .....	4
4.1.	クラスパスを通す .....	4
4.2.	管理ツールから、プラグイン名を指定する。.....	4
5.	サンプル .....	5
5.1.	プラグインの記述例 - ARSMATCHINGSAMPLE.JAVA .....	5
5.2.	ARS 設定例.....	6
5.3.	ノート設定例.....	6
6.	デフォルトのプラグイン.....	7
7.	プラグインから利用できる API.....	7
7.1.	クラス NOTEUTILS.....	7
7.1.1.	READ.....	7
7.1.2.	WRITE .....	7
7.1.3.	LASTMODIFIED .....	8
7.1.4.	EXISTS .....	8

## 1. はじめに

Brekeke PBX v2.1 以降では、ARS のルート検索機能を拡張するためのプラグインが使用できるようになりました。プラグインは Java により開発できます。このドキュメントは、Java によるプログラム開発ができる方を対象に書かれています。

ARS のルート検索では、SIP ヘッダー (From, To) などに対して、正規表現によるマッチングを行います。正規表現によるマッチングは、とても強力ですが、下記のようなケースで本プラグインは有効です。

- ・ 特定の番号リストに存在するかどうかを検索する。例えば、迷惑電話番号リストに存在する電話番号からのコールは拒否する。
- ・ リーストコストルーティング(LCR)のために、通話料金が最も安いルートを特定の国番号、市外局番をキーにリストから検索する。
- ・ 発信者を元に電話帳から、名前を検索する。ディスプレイネームを置き換える事で、電話機に発信者の名前を表示することもできるようになります。

ここでいうリストは、[オプション] > [ノート] メニューを利用することもできます。[ノート] の検索については、単純な検索を行うデフォルトのプラグインが用意されているため、別途プラグインを記述する必要はありません。

## 2. 開発環境

JDK1.4 以降

Brekeke PBX v2.1 以降

## 3. 作成の手順

下記、いずれかの形式でメソッドを作成してください。任意のパッケージ名、クラス名でかまいません。

```
public static String plugin( String param )  
public static boolean plugin( String param )
```

## 4. インストール・設定方法

### 4.1. クラスパスを通す

コンパイルしたクラスファイルを下記ディレクトリ下に、パッケージ名のディレクトリ構造で置きます。

<Brekeke PBX インストールディレクトリ>/webapps/pbx/WEB-INF/classes

例) YourClass というクラスのパッケージ名が com.yourdomain の場合は、

<Brekeke PBX インストールディレクトリ>/webapps/pbx/WEB-INF/classes/com/yourdomain

というフォルダの下に、YourClass.class を置きます。

または、クラスファイルをディレクトリ階層ごとJAR ファイルに圧縮して、下記ディレクトリの下に置いてください。

<Brekeke PBX インストールディレクトリ>/webapps/pbx/WEB-INF/lib

### 4.2. 管理ツールから、プラグイン名を指定する

ARS 設定の マッチングパターンに以下のフィールドに値を設定します。

#### [Plugin] フィールド

パッケージを含むクラス名とメソッド名を指定します。

例) yourpackage.YourPluginClass.yourMethod

#### [Param] フィールド

プラグインに渡すパラメータを設定します。マッチングパターンの各フィールドによる置換対象文字列(括弧で指定する部分)を利用することもできます。

各フィールドの参照の仕方

[From]        &f1, &f2 .... &f9

[To]            &t1, &t2 .... &t9

[User]        &u1,&u2 .... &u9

#### [Return] フィールド

boolean もしくは、String により戻り値を指定します。

プラグインのメソッドが、boolean を戻り値とするケースでは、このフィールドは使用しません。その場合は、プラグインが、true を返した時には、このパターンが適用されます。

プラグインのメソッドが、String を戻り値とするケースでは、このフィールドに正規表現による条件を指定できます。この条件がマッチすれば、パターンが適用されます。また、[デプロイパターン] の各フィールドに、[Return] フィールドによる置換文字列を指定できます。(&p1, &p2.... &p9)

## 5. サンプル

### 5.1. プラグインの記述例 – ARSMatchingSample.java

```
package yourpackage;

import java.util.*;
import java.util.regex.*;
import com.brekeke.pbx.common.*;

public class ARSMatchingSample {

    private static long lastmodified = 0;
    private static ArrayList patternlist = null;

    public static synchronized String regex( String param ) throws Exception {
        long l = NoteUtils.lastModified( "Regex" );
        if( l == 0 ){
            return null;
        }
        if( lastmodified != l ){
            lastmodified = l;
            String s = NoteUtils.read( "Regex" );
            if( s == null ){
                return null;
            }
            ArrayList al = new ArrayList();
            StringTokenizer st = new StringTokenizer( s );
            while( st.hasMoreTokens() ){
                String token = st.nextToken();
                Pattern pt = Pattern.compile( token );
                al.add( pt );
            }
            patternlist = al;
        }
        for( int i = 0; i < patternlist.size(); i++ ){
            Pattern pt = (Pattern) patternlist.get(i);
            Matcher mt = pt.matcher( param );
            if( mt.matches() ){
                return mt.group(1);
            }
        }
        return null;
    }
}
```

”Regex” という名前のノートを読み込んで、正規表現によるマッチングを行います。ノートに記述してある各行を上から順番にマッチングします。マッチした行が見つかったら、() で示された部分に該当する文字列を返します。マッチする行が見つからない場合は、null を返します。(null を返すと、そのパターンは不一致としてみなされます。) ノートの内容は、変数にキャッシュされますが、ノートが更新された場合には、再読み込みするようになっています。

## 5.2. ARS 設定例

ARS 設定の マッチングパターンの以下のフィールドに値を設定します。パターン IN に設定するか、パターン OUT に設定するかは、状況に応じて決めてください。

,マッチングパターン

From	
To	<a href="#">sip:(.+)</a> @
User	
Plugin	yourpackage.Sample.regex
Param	&t1
Return	(.+)

デプロイパターン

From	
To	sip:&p1@domain.com
DTMF	
Target	

この例では、To の user-info 部をパラメータとして、プラグインに渡しています。戻り値を&p1 として取り出して、発信先の To にセットします。

## 5.3. ノート設定例

```
^1650(.+)$
^1888(.+)$
^1800(.+)$
```

1650, 1888, 1800 のいずれかで開始する番号の場合は、それに続く番号をパラメーターに指定された情報から取り出します。

## 6. デフォルトのプラグイン

Brekeke PBX には、デフォルトで以下のプラグインが用意されています。

### contains

パラメータ	ノート名, 文字列
戻り値	boolean 値。該当するノートから、指定された文字列と、同一の行があれば true を返す。

### lookup

パラメータ	ノート名, 文字列, [検索対象インデックス(default=1)], [戻り値となる列のインデックス(default=2)]
戻り値	String 値。指定されたノートから検索を行う。 ノートには、各行にカンマ区切りの複数の文字列が記述されており、検索対象となる列において指定された文字列と一致する行を検索し、完全一致する行が見つかった場合は、その行の戻り値となる列のインデックスとして指定された文字列を返す。見つからなかった場合は、長さ 0 の文字列を返す。

## 7. API

### 7.1. クラス NoteUtils

パッケージ: com.brekeke.pbx.common

ノートへのアクセスを行うためのクラスです。プラグインから、ノートの内容を参照、編集を行うには、このクラスのメソッドを利用するようにしてください。

#### 7.1.1. read

```
public static String read( String name )
```

ノートの内容を読み込みます。

パラメータ

name ノート名

戻り値

ノートの内容

#### 7.1.2. write

```
public static boolean write( String name, String text )
```

ノートに文字列を書き込みます。

#### パラメータ

name ノート名  
text ノートに書き込む文字列

#### 戻り値

成功した場合は、true、そうでない場合は false を返します。

### 7.1.3. lastModified

```
public static long lastModified( String name )
```

ノートが編集された時刻を取得します。

#### パラメータ

name ノート名

#### 戻り値

ノートが最後に編集された時刻を long 値で返します。指定されたノートが存在しない場合や、その他のエラーが発生した場合は、0L を返します。

### 7.1.4. exists

```
public static boolean exists( String name )
```

ノートが存在するかどうかを調べます。

#### パラメータ

name ノート名

#### 戻り値

指定されたノートが存在する場合は true、そうでない場合は false を返します。