◆Java でメールを送信する - SMTP & Email Sending API 2019-07-01
http://blog.smtps.jp/entry/2019/07/01/124359

SMTP を使ったメール送信について
標準ライブラリだけで実装します。SMTP サーバのアドレスは指定されたものに置き換えてください。まず必要なライブラリを読み込みます。

```
import java.io.UnsupportedEncodingException;
import java.util.Properties;

import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.Authenticator;
import javax.mail.PasswordAuthentication;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
```

続いて認証情報を提供するクラスを作ります。これは Authenticator を継承します。この認証情報は先ほど作成した API ユーザ名と API キーになります。

```
class myAuth extends Authenticator {
  protected PasswordAuthentication getPasswordAuthentication(){
    String apiUser = "api@smtps.jp";
    String apiKey  = "YOUR_API_KEY";
    return new PasswordAuthentication(apiUser, apiKey);
  }
}
```

次にメール送信を行うクラスを定義します。この main メソッドを実行することとします。

```
public class SendMail {
  public static String smtpHost = "sandbox.smtps.jp";
  public static String smtpPort = "10025";
  public static String smtpAuth = "true";

  public static void main(String[] args){
    smtp();
  }
  private static void smtp() {
  }
}
```

この smtp メソッドの内容は、まずプロパティを使って SMTP サーバを定義します。さらにネットワークセッションを作る際に、先ほどの認証情報を呼び出します。

```
private static void smtp() {
   Properties objPrp=new Properties();
   objPrp.put("mail.smtp.host", SendMail.smtpHost);
   objPrp.put("mail.host", SendMail.smtpHost);
   objPrp.put("mail.smtp.port", SendMail.smtpPort);
   objPrp.put("mail.smtp.auth", SendMail.smtpAuth);
   // メールセッションを確立
   Session session = Session.getDefaultInstance(objPrp, new myAuth());
   // 省略
}
```

そしてメールを組み立てます。メールの送信は Transport.send(msg); で行います。メール送信はネットワーク状態などでエラーが出る可能性があるので、必ず try/catch を使います。

```
// 送信メッセージを生成
MimeMessage msg = new MimeMessage(session);
try {
   String to = "to@smtps.jp";
   String fromName = "Mailer";
   String fromAddress = "info@smtps.jp";
   String subject = "テストメール from Customers Mail Cloud";
   String body = "こんにちは";

   msg.setRecipients(Message.RecipientType.TO, to);
   InternetAddress objFrm= new InternetAddress(fromAddress, fromName);
   msg.setFrom(objFrm);
   msg.setSubject(subject,"UTF-8");
   msg.setText(body,"UTF-8");
   Transport.send(msg);
} catch (UnsupportedEncodingException e) {
   e.printStackTrace();
} catch (MessagingException e) {
   e.printStackTrace();
}
```

これで SMTP 経由でのメール送信が完了します。

API 経由でのメール送信について

API サーバについて

API サーバのエンドポイント URL は契約しているプランによって異なりますのでご注意ください。

プラン名　　エンドポイント URL
無料トライアル　https://sandbox.smtps.jp/api/v2/emails/send.json
Standard プラン　https://te.smtps.jp/api/v2/emails/send.json
Pro プラン　　https://SUBDOMAIN.smtps.jp/api/v2/emails/send.json
SUBDOMAIN は、サービス利用開始時に申請いただいたものです

クラスを作成する
API には JSON を送信します。そのためのクラスを用意します。JSON 全体を表すのが MailJson クラスです。

```
package sendmail;

public class MailJson {
   public String api_user;
   public String api_key;
   public String subject;
   public String text;
   public MailAddress from;
   public MailAddress[] to;
}
```

メールアドレス部分だけを表すのが MailAddress クラスです。

```
package sendmail;

public class MailAddress {
   public String name;
   public String address;
}
```

変数を設定する
変数は SMTP サーバ利用時とほぼ同等、SMTP サーバの代わりにエンドポイント URL を指定します。

```
String toName = "User";
String toAddress = "to@smtps.jp";
String fromName = "Mailer";
String fromAddress = "info@smtps.jp";
String subject = "テストメール  from Customers Mail Cloud";
String text = "こんにちは";
String apiUser = "api@smtps.jp";
String apiKey   = "YOUR_API_KEY";
String url = "https://sandbox.smtps.jp/api/v2/emails/send.json";
```

配信内容は先ほどのクラスを使って定義します。

```
// JSON の組み立て
```

```
MailJson obj = new MailJson();
obj.api_user = apiUser;
obj.api_key = apiKey;
obj.subject = subject;
obj.text = text;

// 送信元
MailAddress from = new MailAddress();
from.name = fromName;
from.address = fromAddress;
obj.from = from;

// 送信先
MailAddress to = new MailAddress();
to.name = toName;
to.address = toAddress;
obj.to = new MailAddress[1];
obj.to[0] = to;
```
そしてクラスを Jackson を使って JSON 文字列にします。

```
ObjectMapper mapper = new ObjectMapper();
String json = mapper.writeValueAsString(obj);
```
JSON の内容については 共通仕様 | Customers Mail Cloud を参照してください。

送信を行う

では API サーバのエンドポイント URL と配信情報のパラメータを使ってメール送信を行います。メール送信 API は POST メソッドを使います。HTTP アクセスは org.apache.http を使います。JSON 文字列にする Jackson も読み込みます。

```
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.util.EntityUtils;
import org.apache.http.entity.StringEntity;
// Jackson
import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.ObjectMapper;
```
HTTP メソッドとヘッダーの定義をし、HTTP リクエストオブジェクトを作成します。

```
StringEntity entity = new StringEntity(json, "UTF-8");
HttpPost httpPost = new HttpPost(url);
```

```
httpPost.setHeader("Content-type", "application/json; charset=UTF-8");
httpPost.setEntity(entity);
```
そして HTTP リクエストを実行します。

```
CloseableHttpClient client = HttpClients.createDefault();
CloseableHttpResponse response = client.execute(httpPost);
System.out.println(EntityUtils.toString(response.getEntity()));
// 閉じる
client.close();
```
送信が成功すると、下記のようにメール ID が返ってきます。

```
{"id":"¥u003C314195997.17022.1560845073906@mta02.sandbox.smtps.jp¥u003E"}
```
エラーがあると、下記のようにエラー内容が返ってきます。

```
{"errors":[{"code":"02-001","field":"text","message":"text is required."}]}
```
まとめ

サーバ上の制限によって SMTP が使えない場合でも HTTP 経由でメール配信が行えますので、API をぜひ使ってみてください。今回は Java を使いましたが、汎用的な HTTP アクセスを行っていますので、他のプログラミング言語でも簡単に実装できるでしょう。

さらに詳しい使い方については Email Sending API を参照してください。

◆Email Sending API

https://smtps.jp/docs/apiv2/es/index.html#email-sending-api

Email Sending API

一括メール、または、単一メールを送信することができる API を提供します。

宛先アドレスのリストとメール本文をパラメータに指定して一括メールを送信する仕組みを提供します。 差込み文字を使用して、宛先アドレスごとに内容の異なるメールを生成することもできます。

また、1 通のメールを簡単に送信することができる API を提供します。 アプリケーションを配置している ISP やクラウドの制限により SMTP による外部通信ができない場合、 この API を利用することでメールを送信することができます。

バージョン

POST パラメータや応答メッセージのフォーマットは API のバージョンによって異なります。 利用するバージョンの API 仕様を参照ください。

現在のバージョンは２です。

バージョン 1 の API 仕様はこちらを参照ください。

共通仕様
このセクションでは、Email Sending API の共通仕様について説明します。

項目　　値
プロトコル　　HTTPS
文字コード　　UTF-8
改行コード　　LF
接続先ホストと URL
https://[hostname]/api/[version]/[resource]/[action].[format]

[hostname]は、この API を提供するホスト名を表します。
[version]は、API バージョンを表します。v2 と v1 が指定できます。
[resource]は、操作対象のリソースを表します。
[action]は、操作を表します。
[format]は、レスポンスメッセージのフォーマットを表します。json と xml が指定できます。
Email Sending API は、SMTP と同様にメールサーバーが機能を提供します。
サービスプランにより接続先ホストを以下の通り指定してください。

sandbox
無料トライアルでは、以下のエンドポイントに接続することができます。

項目　　値
エンドポイント URL　　https://sandbox.smtps.jp/api/v2/emails/send.json
transactional-email
Standard プランでは、以下のエンドポイントに接続することができます。

項目　　値
エンドポイント URL　　https://te.smtps.jp/api/v2/emails/send.json
Pro プランでは、以下のエンドポイントに接続することができます。

項目　　値
エンドポイント URL　　https://SUBDOMAIN.smtps.jp/api/v2/emails/send.json
*) SUBDOMAIN は、サービス利用開始時に申請頂いたドメイン名です。

認証
Email Sending API は、HTTP リクエストに含まれる api_user, api_key パラメータを使用した ユーザ認証を行います。API ユーザの登録方法については、ユーザガイドの「API ユーザ」 を参照ください。

HTTP リクエスト
このセクションでは、Email Sending API が受信する HTTP リクエストの仕様について説明します。

メソッド
Email Sending API は、POST メソッドのみを受け付けます。

ヘッダ
Email Sending API は、以下のヘッダフィールドを参照します。

ヘッダフィールド　　説明
Accept-Language Email Sending API が応答するメッセージの言語タイプを指定します。en または ja のいずれかを指定します。言語指定が無いまたは左記以外の言語を指定された場合、en で動作します。
コンテンツタイプ
通常は、application/x-www-form-urlencoded または application/json を指定します。 添付ファイル付きメールを送信する場合はファイルアップロードが必要となるため、multipart/form-data を指定します。

パラメータ
Email Sending API は全ての HTTP リクエストに、ユーザ認証を行うための api_user, api_key, を含める必要があり、かつ、URL ごとに定義されたパラメータを送信する必要があります。

このドキュメントは以下フォーマットに従い、パラメータの説明を記述します。

パラメータ　　必須　　データ型　　説明
api_user　Yes　　ASCII　　ユーザ認証に使用する ID
api_key　Yes　　ASCII　　ユーザ認証に使用するシークレットキー
パラメータ：POST するパラメータの名前。
必須：このパラメータが必須である場合、Yes。
データ型：パラメータ値のデータ型。（後述、データ型を参照）
説明：このパラメータに関する説明。
デフォルト値、入力値の範囲などの制限があるパラメータについては、説明欄に以下の書式で記載します。

デフォルト：(デフォルト値)
範囲：min=(最小値) / max=(最大値)
制約：パラメータに関する制約の記述。例えば、start_date は end_date と等しいか過去日でなければならないなど。
application/x-www-form-urlencoded (multipart/form-data) を指定する場合
パラメータの名前・値のセットを作成して POST してください。

application/json を指定する場合
パラメータの名前をプロパティとした JSON を作成して POST してください。

```
{
  "api_user" : "smtp_api_user@example.com",
  "api_key" : "a29yZWhhcGFzc3dvcmRkZXN1",
```

```
  "to" : [
    {
      "name" : "Personal Name1",
      "address" : "user1@example.com"
    },
    {
      "name" : "Personal Name2",
      "address" : "user2@example.com"
    }
  ],
  "from" : {
    "name" : "カスタマーサポート",
    "address" : "support@example.co.jp"
  },
  "subject" : "◯◯ショップ：会員登録完了通知",
  "text" : "((#name#)) 様　この度は、....."
}
```

データ型

Email Sending API は以下の仕様に従い、データ型を取扱います。

データ型	説明
UTF-8	ASCII を含むマルチバイト文字を表します。
ASCII	印刷可能な ASCII 文字を表します。
INTEGER	-2147483648 から 2147483647 の範囲の数値を表します。
DATE	YYYY-MM-DD 形式で日付を表します。DD=99 である場合、指定月の末日と解釈します。
TIME	HH:mm 形式で、時および分を 24 時間表記で表します。この API では秒をリクエストパラメータとして取り扱う処理はありません。
DATETIME	YYYY-MM-DD HH:mm 形式で年月日および時分を表します。
BOOLEAN	true または false のいずれかを表します。

HTTP レスポンス

このセクションでは、Email Sending API が送信する HTTP レスポンスの仕様について説明します。

ステータスコード

Email Sending API は、以下のステータスコードを応答します。

コード	メッセージ	説明
200	OK	リクエストは正常に処理されました。
400	Bad Request	不正なパラメータがリクエストされたなどの理由により、リクエストは正常に処理されませんでした。
401	Unauthorized	ユーザ認証に失敗しました。
403	Forbidden	IP 制限やアクセス制御により、ユーザからのリクエスト実行を拒否しました。
404	Not Found	リクエストされた URL は存在しません。

500 Internal Server Error システム内部の問題により、リクエストを実行できませんでした。
503 Service Unavailable アクセス数の超過やサーバの過負荷により、リクエストを実行できませんでした。

ヘッダ

Email Sending API は、URL で指定されたフォーマットに従い、JSON または、XML フォーマットの メッセージを応答します。これらの応答メッセージのフォーマットは、HTTP レスポンスヘッダ Content-Type により判別します。

Content-Type	説明
application/json	JSON フォーマットのメッセージを応答します。
application/xml	XML フォーマットのメッセージを応答します。

メッセージ

Email Sending API は以下のメッセージを応答します。

成功

リクエストが成功した時に応答するメッセージについては、各 API 仕様の「レスポンス」を参照ください。

エラー

このメッセージは、リクエストが失敗した時に応答します。

JSON

```
{
    "errors" : [
        {
            "code" : "01-004",
            "field" : "api_user",
            "message" : "api_user is required."
        },
        {
            "code" : "01-004",
            "field" : "api_key",
            "message" : "api_key is required."
        }
    ]
}
```

XML

```
<errors>
    <error>
        <code>01-004</code>
        <field>api_user</field>
```

```
        <message>api_user is required.</message>
    </error>
    <error>
        <code>01-004</code>
        <field>api_key</field>
        <message>api_key is required.</message>
    </error>
</errors>
```

エラーメッセージ

Email Sending API の共通エラーメッセージを以下に説明します。 共通エラーメッセージ以外にも、各 API ごとに定義したエラーメッセージを応答することがあります。

コード　メッセージ（en）　　メッセージ(ja)

01-001　System error was occurred. Please contact system administrator. システムエラーが発生しました。システム管理者に連絡してください。

01-002　HTTP Request which use GET Method is not permitted. Please use POST Method. GET メソッドを使用した HTTP リクエストは許可していません。POST メソッドを使用してください。

01-004　{0} is required.　{0}は必須項目です。

01-006　Connected IP does not allow to access API. Please confirm source IP setting.　接続 IP は API にアクセスすることはできません。接続元 IP 設定を確認してください。

01-009　Request data is invalid. Check the character code.　リクエストデータが不正です。文字コードを確認ください。

セキュリティ

このセクションでは、Email Sending API のセキュリティについて説明します。

ユーザ認証

Email Sending API は、HTTP リクエストに含まれる api_user, api_key パラメータを使用した ユーザ認証を行います。API ユーザの登録方法については、ユーザガイドの「API ユーザ」 を参照ください。

Email Sending API は SMTP 認証でユーザ認証するため、「SMTP」をチェックオンとする必要があります。