記事
Toshihiko Minamoto · 2022年2月16日 8m read

InterSystems IRIS で IMAP クライアントを実装する - パート I

この記事では、InterSystems IRIS プラットフォームを使用して基本的な IMAP クライアントを記述する方法を説明します。 はじめに IMAP の概要を確認してから、本題の IMAP コマンドとクライアント実装について説明します。 最後に、IRIS 相互運用性アプリケーションでこの IMAP クライアントを簡単に使用します。

この記事では IMAP を詳しく説明していません。 詳細な情報については、この記事のリソースをご覧ください。

IMAP の概要

ユーザーは IMAP（Internet Message Access Protocol）を使ってメールを取得できます。 IMAP は 1980 年代に Marc Crispin によって提唱されました。以来、このプロトコルは RFC 3501 として公開され、改訂されています。この記事の執筆時点での最新バージョンは、IMAP4rev1 です。

このプロトコルは取得のみを目的に設計されていることに十分に注意してください。メールを送信するのであれば、SMTP（Simple Mail Transfer Protocol）という別のプロトコルを使用する必要があります。 また、メールの取得には、IMAP よりも古い POP3（Post Office Protocol version 3）があります。これも IMAP と同様に一般的に使用されています。

基本的な IMAP コマンドを試す

POP3 と同様に、IMAP はプレーンテキストプロトコルです。 Telnet や OpenSSL などの TCP クライアントを使用して、自分で簡単にコマンドを試すことができます。

まず、IMAP ホストとメールサーバーポートが必要です。 たとえば以下は、この記事の執筆時点での、Microsoft Outlook サービスに接続するためのホストとポートです。

outlook.office365.com:993

TCP クライアントを使ってこのサーバーに接続してみましょう。 以下のコマンドを入力して ENTER キーを押します。 -crlf フラグを使用しているところに注意してください。 IMAP の改行コードはキャリッジリターンとラインフィード（CRLF）であるため、IMAP にはこのフラグが必要です。

```
$ openssl s_client -connect outlook.office365.com:993 -crlf -quiet
```

Enter キーを押すと、IMAP サーバーから情報が提示され、入力待ちとなります。

```
depth=2 C = US, O = DigiCert Inc, OU = www.digicert.com, CN = DigiCert ...
* OK The Microsoft Exchange IMAP4 service is ready. [...]
```

この時点で、最初の IMAP コマンドである CAPACITY コマンドを実行できます。名前からわかるように、これはサーバーが提供できる機能を提示するコマンドです。 最初の行は IMAP

コマンド自体であり、残りの行はその出力です。これは、ここで示すすべての IMAP コマンドに共通しています。

```
TAG1 CAPABILITY
* CAPABILITY IMAP4 IMAP4rev1 AUTH=PLAIN AUTH=XOAUTH2 SASL-
IR UIDPLUS ID UNSELECT CHILDREN IDLE NAMESPACE LITERAL+
TAG1 OK CAPABILITY completed.
```

CAPABILITY コマンドの前にある TAG1 というトークンに注意してください。 すべての IMAP
コマンドの前にはタグを配置する必要があり、任意の文字列を使用できます。
このタグによって、サーバーの応答先であるコマンドインスタンスを識別しています。
一般に、任意のプレフィックスに単純なカウンターを使用したものがあらゆるニーズに対応できます。

IMAP サーバーの応答は、コマンドの発行に使用したタグ、コマンドのステータスを示すフラグ、コマンドとステータスに関連するその他の情報となります。
許可されているステータスは、OK（成功）、NO（失敗）、および BAD（誤ったコマンド）です。

LOGIN コマンドでは、引数としてユーザーとパスワードが想定されています。 まず、誰かが無効なユーザー名とパスワードを入力した場合にサーバーがどのように応答するかを確認してみましょう。

```
TAG2 LOGIN user@outlook.com wrong-password
TAG2 NO LOGIN failed.
```

コマンドタグに続くのは NO ステータスであることに注意してください。

次に、有効なログインを試してみましょう。

```
TAG3 LOGIN user@outlook.com password
TAG3 OK LOGIN completed.
```

これでログインできました。 セッションを終了するには、LOGOUT コマンドを使用します。

```
TAG4 LOGOUT
* BYE Microsoft Exchange Server IMAP4 server signing off.
TAG4 OK LOGOUT completed.
```

IMAP クライアントを実装するには他のコマンドが必要であるため、もう一度 IMAP
サーバーに接続してログインする必要があります。

では、受信トレイに関する情報を取得しましょう。

まず、アクセスしようとしているメールボックスがどれかをサーバーに伝える必要があります。この場合は INBOX です。

```
TAG5 SELECT "INBOX"
* 14 EXISTS
* 0 RECENT
...
TAG5 OK [READ-WRITE] SELECT completed.
```

これで、受信トレイのメッセージにアクセスする準備は整いましたが、メッセージを参照する方法が必要です。

この方法には、メッセージ番号または一意の識別子（UID）を使用する 2 つがあります。

メッセージ番号は、1 から始まり、受信トレイ内のメッセージ数までの単純なカウンターです。
メッセージが削除されると、後続のすべてのメッセージの番号が 1 つずつ減ります。 要素の 1
つが削除された配列として考えるとよいでしょう。

一方 UID は、他のメッセージに何が起ころうとも、その値を保持します。

UID は SEARCH コマンドを使って取得できます。 このコマンドには、特定の UID
を取得する場合はメッセージ番号を、ディレクトリ内のすべての UID
を取得する場合は ALL パラメーターを指定できます。

以下の例では、メッセージ番号 1 の UID を検索しています。UID は 483 です。

```
TAG6 UID SEARCH 1
* SEARCH 483
TAG6 OK SEARCH completed.
```

次に、ヘッダー、本文、添付ファイルなどのメッセージ情報を取得してみましょう。 これには FETCH
コマンドを使用します。 このコマンドには多数のパラメーターがあります。詳細は、RFC 3501
をご覧ください。 この記事では、IMAP
クライアントを実装するというニーズに関連するパラメーターのみを説明しています。

このデモで必要となる最初の情報は、メッセージのサイズです。
この情報は、RFC822.SIZE パラメーターを使って取得できます。

```
TAG7 FETCH 1 RFC822.SIZE
* 1 FETCH (RFC822.SIZE 70988)
TAG7 OK FETCH completed.
```

ここでは、メッセージ番号 1 のサイズは 70,988 バイトであることが示されています。

メッセージ番号の代わりに UID を使用してメッセージ情報をフェッチすることも可能です。

```
TAG8 UID FETCH 483 RFC822.SIZE
* 1 FETCH (RFC822.SIZE 70988 UID 483)
TAG8 OK FETCH completed.
```

メッセージヘッダーの基本的な From（送信者）、To（宛先）、Sbuject（件名）を取得することもできます。

```
TAG9 FETCH 1 (FLAGS BODY[HEADER.FIELDS (FROM TO DATE SUBJECT)])
* 1 FETCH (FLAGS (\Seen) BODY[HEADER.FIELDS (FROM TO DATE SUBJECT)] {157}
Date: Thu, 22 Apr 2021 15:49:05 +0000
From: Another User &lt;anotheruser@outlook.com>
To: user@outlook.com
Subject: Greetings from Another User!
 FLAGS (\Seen))
TAG9 OK FETCH completed.
```

次に、メッセージ本文を取得しましょう。 以下のコマンドを使用して、本文の全コンテンツを取得できます。

```
TAG10 FETCH 1 BODY[]
* 1 FETCH (BODY[] {9599}
...
MIME-Version: 1.0
--00000000000041bd3405c3403048
Content-Type: multipart/alternative; boundary="00000000000041bd3205c3403046"
--00000000000041bd3205c3403046
Content-Type: text/plain; charset="UTF-8"
...
--00000000000041bd3405c3403048
Content-Type: image/png; name="download.png"
Content-Disposition: attachment; filename="download.png"
...
TAG10 OK FETCH completed.
```

上記のコードでは、いくつかのブロックが -- -- の間の 16
進数で区切られているのがわかります。これはパートと呼ばれるもので、
パートが複数含まれるメッセージはマルチパートメッセージと呼ばれます。
コマンドにパートインデックスを渡すと、パートを直接取得することができます。

メッセージを削除するために、プロトコルには、STORE と EXPUNGE
という、メッセージを削除済みとしてマークし、操作をコミットするコマンドがあります。

```
TAG11 STORE 1 +FLAGS (\Deleted)
* 0 EXISTS
* 0 RECENT
TAG11 OK [READ-WRITE] SELECT completed; now in selected state
TAG12 EXPUNGE
EXPUNGE: TAG12 OK EXPUNGE completed
```

最後は NOOP というシンプルなコマンドです。 このコマンドは、keep-alive
ストラテジーを実装するためだけに使用されます。 デフォルトでは、IMAP セッションはコマンドなしで 30
分後に閉じられるようになっています。
そのため、NOOP コマンドを発行することで、接続をアクティブな状態に維持することができます。

```
TAG17 NOOP
TAG17 OK NOOP completed.
```

これで、IMAP の概要は終了です。 さらに詳しい情報が必要な方は、Web
上の多数あるお役立ち記事をご覧ください（一部をリソースのセクションに記載しました）。もちろん、RFC 3501 もご参考ください。

リソース

Atmail の「IMAP 101: Manual IMAP Sessions」
Fastmail の「Why is IMAP better than POP?」
IETF の「Internet Message Access Protocol」
IETF の「Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies」
Nylas の「Everything you need to know about IMAP」
次回パートでは、IRIS 内でこれらのコマンドを使用し、実際の動作を確認します！

それではまた！ 

#InterSystems IRIS

ソースURL:https://jp.community.intersystems.com/post/intersystems-iris-%E3%81%A7-imap-%E3%82%AF%E3%83%A9%E3%82%A4%E3%82%A2%E3%83%B3%E3%83%88%E3%82%92%E5%AE%9F%E8%A3%85%E3%81%99%E3%82%8B-%E3%83%91%E3%83%BC%E3%83%88-i