Network Working Group
Request for Comments: 5259
Category: Standards Track

A. Melnikov, Ed.
Isode Ltd
P. Coates, Ed.
Sun Microsystems

July 2008

Internet Message Access Protocol - CONVERT Extension

Status of This Memo

Abstract

CONVERT defines extensions to IMAP allowing clients to request adaptation and/or transcoding of attachments. Clients can specify the conversion details or allow servers to decide based on knowledge of client capabilities, on user or administrator preferences, or on server settings.

Table of Contents

1. Introduction

This document defines the CONVERT extension to IMAP4 [RFC3501]. CONVERT provides adaptation and transcoding of body parts as needed by the client. Conversion (adaptation, transcoding) may be requested by the client and performed by the server on a best effort basis or, when requested by the client, decided by the server based on the server's knowledge of the client capabilities, user or administrator preferences, or server settings.

This extension is primarily intended to be useful to mobile clients. It satisfies requirements specified in [OMA-ME-RD].

A server that supports CONVERT can convert body parts to other formats to be viewed (for example) on a mobile device. The client can explicitly request a particular conversion or ask the server to select the best available conversion. When allowed by the client, the server determines how to convert based on its own strategy (e.g., based on knowledge of the client as discussed hereafter). If the server knows the characteristics of the device (out of scope for CONVERT) or can determine them (for example, using a conversion parameter containing device type), converted body parts can also be optimized for capabilities of the device (e.g., form factor of pictures). The client is able to control conversions using optional conversion (also referred to as "transcoding" in this document) parameters.

This document relies on the registry of conversion parameters established by [MEDIAFEAT-REG]. The registry can be used to discover the underlying legal values that these parameters can take. Additional conversion parameters, such as those defined by [OMA-STI], are expected to be registered in the future.

2. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

In examples, "C:" and "S:" indicate lines sent by the client and server, respectively. If a single "C:" or "S:" label applies to multiple lines, then the line breaks between those lines are for editorial clarity only and are not part of the actual protocol exchange. The five characters [...] mean that something has been elided.

When describing the general syntax, some definitions are omitted as they are defined in [RFC3501]. In particular, the term "session" is used in this document as defined in Section 1.2 of [RFC3501].

3. Relation with Other IMAP Specifications

Conversion of attachments during streaming is out of scope for the CONVERT extension and is described in a separate Lemonade WG document [LEM-STREAMING].

A server claiming compliance with this specification MUST support the IMAP Binary specification [RFC3516].

3.1. CAPABILITY Response

A server that supports the CONVERT extension MUST return "CONVERT" and "BINARY" in the CAPABILITY response or response code. (Client and server authors are reminded that the order of tokens returned in the CAPABILITY response or response code is arbitrary.)

Example: A server that implements CONVERT.

```
C: a000 CAPABILITY
S: * CAPABILITY IMAP4rev1 CONVERT BINARY [...]
S: a000 OK CAPABILITY completed
```

4. Scope of Conversions

Conversions only affect what is sent to the client; the original data in the message store MUST NOT be altered. This document does not specify how the server performs conversions.

Note: The requirement that original data be unaltered allows such data to remain accessible by other clients, permits replies or forwards of the original documents, permits signature verification (the converted body parts are not likely to contain any signatures), and preserves BODYSTRUCTURE and related information.

5. Discovery of Available Conversions

5.1. CONVERSIONS Command

Arguments: source MIME type
target MIME type

Responses: untagged responses: CONVERSION

Result: OK - CONVERSIONS command completed
BAD - unrecognized syntax of an argument, unexpected extra argument, missing argument, etc.

The CONVERSIONS command is allowed in Authenticated and Selected IMAP states.

The first parameter to the CONVERSIONS command is a source MIME type, the second parameter is the target MIME type. Both parameters are partially (e.g., "text/*") or completely ("*") wildcardable.

Conversions matching the source/target pair and their associated conversion parameters are returned in untagged CONVERSION responses. If source/target doesn't match any conversion supported by the server, no CONVERSION response is returned.

Examples:

For conversion information from GIF to JPEG image format (no untagged CONVERSION response would be returned if no conversion is possible):

```
C: a CONVERSIONS "image/gif" "image/jpeg"
S: * CONVERSION "image/gif" "image/jpeg" ("pix-y" "pix-x"
    "image-interleave")
S: a OK CONVERSIONS completed
```

For conversion information from GIF image format to anything:

```
C: b CONVERSIONS "image/gif" "*"
S: * CONVERSION "image/gif" "image/jpeg" ("pix-y" "pix-x"
    "image-interleave")
S: * CONVERSION "image/gif" "image/png" ([...])
[...]
S: b OK CONVERSIONS completed
```

For conversion of anything to JPEG:

```
C: c CONVERSIONS "*" "image/jpeg"
S: * CONVERSION "image/gif" "image/jpeg" ("pix-y" "pix-x"
    "image-interleave")
S: * CONVERSION "image/png" "image/jpeg" (...)
[...]
S: c OK CONVERSIONS completed
```

For conversions from all image formats to all text formats, the client can issue the following command:

```
C: d CONVERSIONS "image/*" "text/*"
```

5.2. CONVERSION Response

Contents: source MIME type
target MIME type
optional list of supported conversion parameters

As a result of executing a CONVERSIONS command, the server can return one or more CONVERSION responses. Each CONVERSION response specifies which source MIME type can be converted to the target MIME type, and also lists supported conversion parameters.

6. CONVERT and UID CONVERT Commands

Arguments: sequence set
conversion parameters
CONVERT data item names

Responses: untagged responses: CONVERTED

Result: OK - convert completed
NO - convert error: can't fetch and/or convert that data
BAD - unrecognized syntax of an argument, unexpected extra argument, missing argument, etc.

The CONVERT extension defines CONVERT and UID CONVERT commands that are used to transcode the media type of a MIME part into another media type, and/or the same media type with different encoding parameters. These commands are structured and behave similarly to FETCH/UID FETCH commands as extended by [RFC3516]:

- A successful CONVERT/UID CONVERT command results in one or more untagged CONVERTED responses (one per message). They are similar to the untagged FETCH responses. Note that a single CONVERT/ UID CONVERT command can only perform a single type of conversion as defined by the conversion parameters. A client that needs to perform multiple different conversions needs to issue multiple CONVERT/UID CONVERT commands. Such a client MAY pipeline them.

- BINARY[...] data item requests conversion of a body part or of the whole message according to conversion parameters and requests that the converted message/body part be returned as binary.

- BINARY.SIZE data item is similar to RFC822.SIZE, but it requests size of a converted body part/message.

- BODYPARTSTRUCTURE data item is similar to BODYSTRUCTURE FETCH data item, but it returns the MIME structure of the converted body part.

- BODY[...HEADER] encoded words in the requested headers are converted to the specified charset. The CHARSET parameter is REQUIRED for this conversion.

- BODY[...MIME] encoded words in the requested headers are converted to the specified charset. The CHARSET parameter is REQUIRED for this conversion.

- AVAILABLECONVERSIONS data item requests the list of target MIME types the specified body part (or the whole message) can be converted to.

The CONVERT extension also adds one new response code. See Section 9 for more details.

Typically clients will request conversion of leaf body parts. In addition to support of leaf body part conversion, servers MAY offer conversion of non-leaf body parts (e.g., conversion from multipart/related).

Instead of specifying the exact target MIME media type the client wants to convert to, the client MAY use a special marker NIL (also known as "default conversion") to request the server to pick a suitable target media type. This document doesn't describe how exactly the server makes such a choice; however, some basic guidelines are described in this paragraph. If the server knows characteristics of the device using an in-band (such as device type specified in a conversion parameter) or an out-of-band mechanism, then it should convert the request body part to a media type the device is likely to support and display/play successfully. Unless specifically overridden by a conversion parameter, the server MAY also remove any unnecessary detail that exceeds the capabilities of the device (e.g., scaling images to just fit on the device's screen). In the absence of any in-band or out-of-band mechanism for determining device characteristics, the server should convert the request body part to the most standard or widely deployed media type available in that media category, for example, to convert to text/plain, image/jpeg. In such case, the server should minimize quality loss. Servers are REQUIRED to support "default conversion" requests. Server implementations that support conversions to multiple target MIME types SHOULD make the default conversion configurable. Clients SHOULD avoid using the default conversion unless they provided a way (in-band or out-band) to signal their capabilities to the server, as there is no guaranty that the server would guess their capability correctly. Client implementors should consider using AVAILABLECONVERSIONS CONVERT data item or CONVERSIONS command instead of the default conversion.

CONVERT's command syntax is modeled after the FETCH command syntax in [RFC3501], as extended by [RFC3516]. CONVERT data items are generally structured as:

BINARY[section-part]<partial>

BINARY.SIZE[section-part]

BODYPARTSTRUCTURE[section-part]

BODY[HEADER]

BODY[section-part.HEADER]

BODY[section-part.MIME]

AVAILABLECONVERSIONS[section-part]

The semantics of a partial CONVERT BINARY[...] command is the same as for a partial FETCH BODY[...] command, with the exception that the <partial> arguments refer to the TRANSCODED and DECODED section data.

Note that unlike the FETCH command, the CONVERT command never sets the \Seen flag on converted messages. A client wishing to mark a message with the \Seen flag would need to issue a STORE command (possibly pipelined with the CONVERT request) to do that.

The UID CONVERT command is different from the CONVERT command in the same way as the UID FETCH command is different from the FETCH command:

- UID CONVERT takes as a parameter a sequence of UIDs instead of a sequence of message numbers.
- UID CONVERT command MUST result in the UID data item in a corresponding CONVERTED response.
- An EXPUNGE response MUST NOT be sent while responding to a CONVERT command. This rule is necessary to prevent a loss of synchronization of message sequence numbers between client and server. Note that an EXPUNGE response MAY be sent during a UID CONVERT command.

Example: The client fetches body part section 3 in the message with the message sequence number of 2 and asks to have that attachment converted to pdf format.

```
C: a001 CONVERT 2 ("APPLICATION/PDF") BINARY[3]
S: * 2 CONVERTED (TAG "a001") (BINARY[3] {2135}
   <the document in .pdf format>
   )
S: a001 OK CONVERT COMPLETED
```

Example: The client requests for conversion of a text/html body part to text/plain and asks for a charset of us-ascii. The server cannot respect the charset conversion request because there are non-us-ascii characters in the text/html body part, so it fails the request by returning an ERROR phrase in place of the converted data (see Section 9).

```
C: b001 CONVERT 2 ("text/plain" ("charset" "us-ascii")) BINARY[3]
S: * 2 CONVERTED (tag "b001") (BINARY[3]
    (ERROR "Source text has non us-ascii" BADPARAMETERS
    "text/html" "text/plain" ("charset" "us-ascii")))
S: b001 NO All conversions failed
```

If the client also specified the "unknown-character-replacement" conversion parameter (see Section 12.1), the same example can look like this:

```
C: b001 CONVERT 2 ("text/plain" ("charset" "us-ascii"
    "unknown-character-replacement" "?")) BINARY[3]
S: * 2 CONVERTED (TAG "b001") (BINARY[3] {2135}
    <the document in text/plain format with us-ascii
     charset>
   )
S: b001 OK CONVERT COMPLETED
```

The server replaced non-us-ascii characters with a us-ascii character such as "?".

Example: The client first requests the converted size of a text/html body part converted to text/plain:

```
C: c000 CONVERT 2 ("TEXT/PLAIN" ("CHARSET" "us-ascii"))
    BINARY.SIZE[4]
S: * 2 CONVERTED (TAG "c000") (BINARY.SIZE[4] 3135)
S: c000 OK CONVERT COMPLETED
```

Later on, the client requests 1000 bytes from the converted body part, starting from byte 2001:

```
C: c001 CONVERT 2 ("TEXT/PLAIN" ("CHARSET" "us-ascii"))
    BINARY[4]<2001.1000>
S: * 2 CONVERTED (TAG "c001") (BINARY[4]<2001> {135}
     <bytes 2001 - 2135 of the document in text/plain format>
     )
S: c001 OK CONVERT COMPLETED
```

The server MUST respect the target MIME type and conversion parameters specified by the client in the transcoding request. Note that some conversion parameters can restrict what kind of conversion is possible, while others can remove some restrictions.

It is legal for a client to request conversion of a non-leaf body part, for example, to request conversion of a multipart/* into a PDF document. However, servers implementing this extension are not required to support such conversions. Servers that support such conversions MUST return one or more CONVERSION responses in response to a 'CONVERSIONS "multipart/*" "*"' command. See Section 5.1 for more details.

The client can request header conversions using the BODY[...HEADER] CONVERT request, for example

```
C: D001 FETCH 2 BODY[HEADER]
S: * 2 FETCH (BODY[HEADER] {158}
S: Date: Mon, 20 Apr 2007 20:05:43 +0200
S: From: Peter <peter@siroe.example.com>
S: To: Alexey <alexey@siroe.example.com>
S: Subject: =?KOI8-R?Q?why encode this?=
S:
S: )
S: D001 OK
C: D002 CONVERT 2 (NIL ("CHARSET" "utf-8")) BODY[HEADER]
S: * 2 CONVERTED (TAG "d002") (BODY[HEADER] {157}
S: Date: Mon, 20 Apr 2007 20:05:43 +0200
S: From: Peter <peter@siroe.example.com>
S: To: Alexey <alexey@siroe.example.com>
S: Subject: =?UTF-8?Q?why encode this?=
S:
S: )
S: D002 OK
```

Any such request MUST include the CHARSET parameter. Upon receipt of the request, the server MUST decode any encoded words (as described in [RFC2047]) in headers and return them re-encoded in the specified

charset. (Note that encoded-words might not be needed if the result can be represented entirely in US-ASCII, so the server MAY replace the resulting encoded-words with their pure US-ASCII representation.) If the server can't decode any particular encoded word, for example, if the charset or encoding is not recognized, it MUST leave them as is. Servers SHOULD also support decoding of any parameters as described in [RFC2231]. Support for RFC 2231 parameters might require reformatting of header fields during conversion. Consider the following

```
C: D011 FETCH 3 BODY[1.MIME]
S: * 3 FETCH (BODY[1.MIME] {118}
S: Content-Type: text/plain; charset=utf-8;
S:  foo*0*=utf-8'fr'tr%c0;
S:  foo*1*(very)=%03s_m%c0;
S:  foo*2*=(nasty)%09chant
S:
S: D011 OK
```

The server should preserve the headers during the conversion as much as possible. In case the characters are split (legally!) between fragments of an encoded parameter, the server MUST consolidate the parameter fragments, and convert, emit, and re-fragment them as necessary in order to keep the line length less than 78. Comments embedded like this SHOULD be preserved during conversion, but clients MUST gracefully handle the situation where comments are removed entirely. If the comments are preserved, they MAY be moved after the parameter. For example (continuing the previous example):

```
C: D012 CONVERT 3 (NIL) BODY[1.MIME]
S: * 3 CONVERTED (TAG "D012") (BODY[1.MIME] {109}
S: Content-Type: text/plain; charset=utf-8;
S:  foo*0*=utf-8'fr'tr%c0%03s_;
S:  foo*1*=%m%c0%09chant (very)(nasty)
S:
S: D012 OK
```

No destination MIME type MUST be specified with BODY[HEADER], BODY[section.HEADER], or BODY[section.MIME]. That is, BODY[HEADER], BODY[section.HEADER], or BODY[section.MIME] can only be used with the "default conversion". When performing these conversions, the server SHOULD leave encoded words as encoded words. A failure to do so may alter the semantics of structured headers.

7. CONVERT Conversion Parameters

The registry established by [MEDIAFEAT-REG] defines names of conversion parameters that can be used in the CONVERT command. Support for some conversion parameters is mandatory, as described in Section 7.1.

According to [MEDIAFEAT-REG], conversion parameter names are case-insensitive.

The following example illustrates how target picture dimensions can be specified in a CONVERT request using the PIX-X and PIX-Y parameters defined in [DISP-FEATURES].

```
C: e001 UID CONVERT 100 ("IMAGE/JPEG" ("PIX-X" "128"
    "PIX-Y" "96")) BINARY[2]
S: * 2 CONVERTED (TAG "e001") (UID 100 BINARY[2] ~{4182}
   <this part of a document is a rescaled image in
    JPEG format with width=128, height=96.>
   )
S: e001 OK UID CONVERT COMPLETED
```

7.1. Mandatory-to-Implement Conversions and Conversion Parameters

A server implementing CONVERT MUST support charset conversions for the text/plain MIME type, and MUST support charset conversions from iso-8859-1, iso-8859-2, iso-8859-3, iso-8859-4, iso-8859-5, iso-8859-6, iso-8859-7, iso-8859-8, and iso-8859-15 to utf-8.

The server MUST list "text/plain" as an allowed destination conversion from "text/plain" MIME type (see Section 5.1). A command 'CONVERSIONS "text/plain" "text/plain"' MUST also return "charset" and "unknown-character-replacement" (see Section 12.1) as supported conversion parameters in the corresponding CONVERSION response.

IMAP servers implementing the CONVERT extension MUST support recognition of the "charset" [CHARSET-REG] parameter for text/plain, text/html, text/css, text/csv, text/enriched, and text/xml MIME types. Note, a server implementation is not required to support any conversion from the text MIME subtypes specified above, except for the mandatory-to-implement conversion described above. That is, a server implementation MUST support the "charset" parameter for text/csv, only if it supports any conversion from text/csv.

The server MUST support decoding of [RFC2047] headers and their conversion to UTF-8 as long as the encoded words are in one of the supported charsets.

Servers SHOULD offer additional character encoding conversions where they make sense, as character conversion libraries are generally available on many platforms.

If the server cannot carry out the charset conversion while preserving all the characters (i.e., a source character can't be represented in the target charset), and the "unknown-character-replacement" conversion parameter is not specified, then the server MUST fail the conversion and MUST return the untagged ERROR BADPARAMETERS response (see Section 9). If the value specified in the "unknown-character-replacement" conversion itself can't be represented in the target charset, then the server MUST also fail the conversion and MUST return the untagged ERROR BADPARAMETERS response (see Section 9).

7.2. Additional Features for Mobile Usage

This section is informative.

Based on the expected usage of CONVERT in mobile environments, server implementors should consider support for the following conversions:

- Conversion of HTML and XHTML documents to text/plain in ways that preserve at the minimum the document structure and tables.
- Image conversions among the types image/gif, image/jpeg, and image/png for at least the following parameters:
  - size limit (i.e., reduce quality)
  - width ("pix-x" parameter)
  - height ("pix-y" parameter)
  - resize directive (crop, stretch, aspect ratio)

  The support for "depth" may also be of interest.

Audio conversion is also of interest but the relevant formats depend significantly on the usage context.

8. Request/Response Data Items to CONVERT/UID CONVERT Commands

8.1. CONVERTED Untagged Response

Contents: convert correlator
CONVERTED return data items

The CONVERTED response may be sent as a result of a successful, partially successful, or unsuccessful CONVERT or UID CONVERT command specified in Section 6.

The CONVERTED response starts with a message number, followed by the "CONVERTED" label. The label is followed by a convert correlator, which contains the tag of the command that caused the response to be returned. This can be used by a client to match a CONVERTED response against a corresponding CONVERT/UID CONVERT command.

The convert correlator is followed by a list of one or more CONVERT return data items. If the UID data item is returned, it MUST be returned as the first data item in the CONVERTED response. This requirement is to simplify client implementations. See Section 10 and the remainder of Section 8 for more details.

8.2. BODYPARTSTRUCTURE CONVERT Request and Response Item

BODYPARTSTRUCTURE[section-part]

The CONVERT extension defines the BODYPARTSTRUCTURE CONVERT data item. Data contained in the BODYPARTSTRUCTURE return data item follows the exact syntax specified in the [RFC3501] BODYSTRUCTURE data item, but only contains information for the converted part. All information contained in BODYPARTSTRUCTURE pertains to the state of the part after it is converted, such as the converted MIME type, sub-type, size, or charset. Note that the client can expect the returned MIME type to match the one it requested (as the server is required to obey the requested MIME type) and can treat mismatch as an error.

The returned BODYPARTSTRUCTURE data MUST match the BINARY data returned for exactly the same conversion in the same IMAP "session". This requirement allows a client to request BODYPARTSTRUCTURE and BINARY data in separate commands in the same IMAP session.

If the client lists a BODYPARTSTRUCTURE data item for a section-part before a BINARY data item for the same section-part, then, in the CONVERTED response, the server MUST return the BODYPARTSTRUCTURE data prior to the corresponding BINARY data. Also, any BODYSTRUCTURE data

items MUST be after the UID data item if the UID data item is present. Both requirements are to simplify handling of converted data in clients.

Example:

```
C: e002 CONVERT 2 (NIL ("PIX-X" "128" "PIX-Y" "96")) (BINARY[2]
    BODYPARTSTRUCTURE[2])
S: * 2 CONVERTED (TAG "e002") (BODYPARTSTRUCTURE[2] ("IMAGE"
    "JPEG" () NIL NIL "8bit" 4182 NIL NIL NIL) BINARY[2]
    ~{4182}
   <this part of a document is a rescaled image in
    JPEG format with width=128, height=96.>
   )
S: e002 OK CONVERT COMPLETED
```

8.3. BINARY.SIZE CONVERT Request and Response Item

BINARY.SIZE[section-part]

This item requests the converted size of the section (i.e., the size to expect in a response to the corresponding CONVERT BINARY request). The returned value MUST be exact and MUST NOT change during a duration of an IMAP "session", unless the message is expunged in another session (see below). This allows a client to download a converted part in chunks (using "<partial>"). This requirement means that in most cases the server needs to perform conversion of the requested body part before returning its size.

If the message is expunged in another session, then the server MAY return the value 0 in response to the BINARY.SIZE request item later in the same session.

In order to allow for upgrade of server transcoding components, clients MUST NOT assume that repeating a particular body part conversion in another IMAP "session" would yield the same result as a previous conversion of the very same body part -- any characteristics of the converted body part might be different (format, size, etc.). In particular, clients MUST NOT cache sizes of converted messages/ body parts beyond duration of any IMAP "session", or use sizes obtained in one connection in another IMAP connection to the same server.

Historical note: Previous experience with IMAP servers that returned estimated RFC822.SIZE value shows that this caused interoperability problems. If the server returns a value that is smaller than the actual size, this will result in data truncation if <partial>

download is used. If the server returns a value that is bigger than the actual size, this might mislead a client to believe that it doesn't have enough storage to download a body part.

Note for client implementors: client authors are cautioned that this might be an expensive operation for some server implementations. Requesting BINARY.SIZE for a large number of converted body parts or for multiple conversions of the same body part can result in slow performance and/or excessive server load and is discouraged. Client implementors should consider implementation approaches that limit this request to only the most necessary cases and are encouraged to test the performance impact of BINARY.SIZE with multiple server implementations.

8.4. AVAILABLECONVERSIONS CONVERT Request and Response Item

AVAILABLECONVERSIONS[section-part] allows the client to request the list of target MIME types the specified body part of a message or the whole message can be converted to. This data item is only useful when the default conversion (see Section 6) is requested.

This data item MUST return a list of target MIME types that is a subset of the list returned by the CONVERSIONS command for the same source and target MIME type pairs. If specific conversion is requested, it MUST return the target MIME type as requested in the CONVERT command, or the ERROR phrase.

For both specific or default conversion requests, if conversion parameters are specified, then the server must take them into consideration when generating the list of target MIME types. For example, if one or more of the conversion parameters doesn't apply to a potential target MIME type, then such MIME type MUST be omitted from the resulting list. If the server only had a single target MIME type candidate and it was discarded due to the list of conversion parameters, then the server SHOULD return the ERROR phrase instead of the empty list of the target MIME types.

The AVAILABLECONVERSIONS request SHOULD be processed quickly if specified by itself. Note that if a MIME type is returned in response to the AVAILABLECONVERSIONS, there is no guaranty that the corresponding BINARY/BINARY.SIZE/BODYPARTSTRUCTURE CONVERT request will not fail.

Example:

```
C: f001 CONVERT 2 (NIL) (AVAILABLECONVERSIONS[2])
S: * 2 CONVERTED (TAG "f001") (AVAILABLECONVERSIONS[2]
            (("IMAGE/JPEG" "application/PostScript"))
S: f001 OK CONVERT COMPLETED
```

8.5. Implementation Considerations

Note that this section is normative.

Servers MAY refuse to execute conversion requests that convert multiple messages and/or body parts at once, e.g., a conversion request that specifies multiple message numbers/UIDs. If the server refuses a conversion because the request lists too many messages, the server MUST return the MAXCONVERTMESSAGES response code (see Section 9). For example:

```
C: g001 CONVERT 1:* ("text/plain" ("charset" "us-ascii"))
    BINARY[3]
S: g001 NO [MAXCONVERTMESSAGES 1]
```

If the server refuses a conversion because the request lists too many body parts, the server MUST return the MAXCONVERTPARTS response code (see Section 9). For example:

```
C: h001 CONVERT 1 ("text/plain" ("charset" "us-ascii"))
    (BINARY[1] BINARY[2])

S: g001 NO [MAXCONVERTPARTS 1] You can only request 1 body part at
    any given time
```

Note for server implementors: In order to improve performance, implementations SHOULD cache converted body parts. For example, the server may perform a body part conversion when it receives the first BINARY.SIZE[...], BODYPARTSTRUCTURE[...], or BINARY[...] request and cache it until the client requests conversion/download of another body part, a different conversion of the same body part, or until the mailbox is closed. In order to mitigate denial-of-service attacks from misbehaving or badly-written clients, a server SHOULD limit the number of converted body parts it can cache. Servers SHOULD be able to cache at least 2 conversions at any given time.

9. Status Responses and Response Code Extensions

A syntactically invalid MIME media type SHOULD generate a BAD tagged response from the server. An unrecognized MIME media type generates a NO tagged response.

Some transcodings may require parameters. If a transcoding request with no parameters is sent for a format which requires parameters, the server will return an ERROR MISSINGPARAMETERS phrase in place of the data associated with the data items requested. This is analogous to the NIL response in FETCH, but with structured data associated with the failure.

If the server is unable to perform the requested conversion because a resource is temporary unavailable (e.g., lack of disk space, temporary internal error, transcoding service down), then the server MUST return a tagged NO response that SHOULD contain the TEMPFAIL response code (see below), or an ERROR TEMPFAIL phrase.

If the requested conversion cannot be performed because of a permanent error, for example, if a proprietary document format has no existing transcoding implementation, the server MUST return a CONVERTED response containing a ERROR BADPARAMETERS or ERROR MISSINGPARAMETERS phrase.

The server MAY choose to return one ERROR phrase for a single conversion if several related data items are requested. For instance:

```
C: b002 CONVERT 2 ("text/plain" ("charset" "us-ascii"))
    (BINARY[3] BODYPARTSTRUCTURE[3])
S: * 2 CONVERTED (tag "b002") (BODYPARTSTRUCTURE[3]
    (ERROR "Source text has non us-ascii" BADPARAMETERS
    "text/html" "text/plain" ("charset" "us-ascii")))
S: b002 NO All conversions failed
```

If at least one conversion succeeds, the server MUST return an OK response. If all conversions fail, the server MAY return OK or NO. For instance:

```
C: b002 CONVERT 2 ("text/plain" ("charset" "us-ascii"))
    (BINARY[3] BODYPARTSTRUCTURE[3] BINARY[4]
    BODYPARTSTRUCTURE[4])
S: * 2 CONVERTED (tag "b002") (BODYPARTSTRUCTURE[3]
    (ERROR "Source text has non us-ascii" BADPARAMETERS
    "text/html" "text/plain" ("charset" "us-ascii"))
    BODYSTRUCTURE[4] ("TEXT" "PLAIN" (CHARSET US-ASCII)
    NIL NIL "8bit" 4182 NIL NIL NIL) BINARY[4] {4182}
      <body in text plain>
   )
S: b002 OK Some conversions failed
```

In general, the client can tell from the BODYPARTSTRUCTURE response whether or not its request was honored exactly, but may not know the reasons why.

This document defines the following response codes that can be returned in the tagged NO response code.

TEMPFAIL - The transcoding request failed temporarily. It might succeed later, so the client MAY retry.

MAXCONVERTMESSAGES <number> - The server is unable or unwilling to convert more than <number> messages in any given CONVERT/UID CONVERT request.

MAXCONVERTPARTS <number> - The server is unable or unwilling to convert more than <number> body parts of a message at once in any given CONVERT/UID CONVERT request.

The word ERROR is always followed by an informal human-readable descriptive text, which is followed by the convert-error-code. The convert-error-code MUST be one of the following:

TEMPFAIL mm - The transcoding request failed temporarily. It might succeed later, so the client MAY retry. The client SHOULD wait for at least mm minutes before retrying.

BADPARAMETERS from-concrete-mime-type to-mime-type "(" transcoding-params ")" -
The listed parameters were not understood, not valid for the source/destination MIME type pair, had invalid values or could not be honored for another reason noted in the human-readable text that was specified after the ERROR label. The transcoding-params can be omitted, in which case, it means that the conversion from the from-concrete-mime-type to the to-mime-type is not possible. If the from-concrete-mime-type is NIL, this means that the specified body part doesn't exist. All unrecognized or irrelevant parameters MUST be listed in the transcoding-params. It is not legal behavior to ignore irrelevant parameters.

Note that if the client requested the "default conversion" (see Section 6), the to-mime-type contains the destination MIME type chosen by the server.

MISSINGPARAMETERS from-concrete-mime-type to-mime-type "(" transcoding-params ")" -
The listed parameters are required for conversion of the specified source MIME type to the destination MIME type, but were not seen in the request. Note that if the client requested the "default conversion" (see Section 6), the to-mime-type contains the destination MIME type chosen by the server.

Examples:

```
C: b002 CONVERT 2 ("APPLICATION/PDF") BINARY[3]
S: b002 NO [TEMPFAIL] All conversions failed

C: b003 CONVERT 2 ("TEXT/PLAIN") BINARY[3]
S: * 2 CONVERTED (tag "b003") (BINARY[3]
    (ERROR "CHARSET must be specified for text conversions"
    MISSINGPARAMETERS (CHARSET)))
S: b003 NO All conversions failed

C: b005 CONVERT 2 ("TEXT/PLAIN" (CHARSET "US-ASCII"
          UNKNOWN-CHARACTER-REPLACEMENT "<badchar>")) BINARY[3]
S: * 2 CONVERTED (tag "b005") (BINARY[3]
    (ERROR "UNKNOWN-CHARACTER-REPLACEMENT limited to 4
    bytes" BADPARAMETERS (UNKNOWN-CHARACTER-REPLACEMENT
    "<badchar>")))
S: b005 NO All conversions failed
```

10. Formal Syntax

The following syntax specification uses the augmented Backus-Naur Form (ABNF) notation as used in [ABNF], and incorporates by reference the core rules defined in that document.

This syntax augments the grammar specified in [RFC3501] and [RFC3516]. Non-terminals not defined in this document can be found in [RFC3501], [RFC3516], [IMAPABNF], [MIME-MTSRP], and [MEDIAFEAT-REG].

```
command-select  =/ convert

uid             =/ "UID" SP convert
                ; Unique identifiers used instead of message
                ; sequence numbers

convert         = "CONVERT" SP sequence-set SP convert-params SP
                  ( convert-att /
                    "(" convert-att *(SP convert-att) ")" )

convert-att     = "UID" /
                  "BODYPARTSTRUCTURE" section-convert /
                  "BINARY" section-convert [partial] /
                  "BINARY.SIZE" section-convert /
                  "BODY[HEADER]" /
                  "BODY[" section-part ".HEADER]" /
                  "BODY[" section-part ".MIME]" /
                  "AVAILABLECONVERSIONS" section-convert
```

```
                ; <partial> is defined in [RFC3516].
                ; <section-part> is defined in [RFC3501].

      convert-params = "(" (quoted-to-mime-type / default-conversion)
                       [SP "(" transcoding-params ")"] ")"

      quoted-to-mime-type = DQUOTE to-mime-type DQUOTE

      transcoding-params  = transcoding-param
                            *(SP transcoding-param)

      transcoding-param-names  = transcoding-param-name
                            *(SP transcoding-param-name)

      transcoding-param  = transcoding-param-name SP
                           transcoding-param-value

      transcoding-param-name = astring
               ; <transcod-param-name-nq> represented as a quoted,
               ; literal or atom.  Note that
               ; <transcod-param-name-nq> allows for "%", which is
               ; not allowed in atoms.  Such values must be
               ; represented as quoted or literal.

      transcod-param-name-nq = Feature-tag
               ; <Feature-tag> is defined in [MEDIAFEAT-REG].

      transcoding-param-value = astring

      default-conversion = "NIL"

      message-data   =/ nz-number SP "CONVERTED" SP convert-correlator
                        SP convert-msg-attrs

      convert-correlator = "(" "TAG" SP tag-string ")"

      tag-string = string
                ; tag of the command that caused
                ; the CONVERTED response, sent as
                ; a string.

      convert-msg-attrs = "(" convert-msg-att *(SP convert-msg-att) ")"
                ; "UID" MUST be the first data item, if present.

      convert-msg-att = msg-att-semistat / msg-att-conv-static

      msg-att-conv-static  = "UID" SP uniqueid
                ; MUST NOT change for a message
```

```
      msg-att-semistat =
                  ( "BINARY" section-convert ["<" number ">"] SP
                     (nstring / literal8 / converterror-phrase) ) /
                  ( "BINARY.SIZE" section-convert SP
                     (number / converterror-phrase) ) /
                  ( "BODYPARTSTRUCTURE" section-convert SP
                     (body / converterror-phrase) ) /
                  ( "AVAILABLECONVERSIONS" section-convert SP
                     (mimetype-list / converterror-phrase) )
                   ; MUST NOT change during an IMAP "session",
                   ; but not necessarily static in the long term.

      section-convert = section-binary
                    ; <section-binary> is defined in [RFC3516].
                    ;
                    ; Note that unlike [RFC3516], conversion
                    ; of a top level multipart/* is allowed.

      resp-text-code =/ "TEMPFAIL" /
                        "MAXCONVERTMESSAGES" SP nz-number /
                        "MAXCONVERTPARTS" SP nz-number
          ; <resp-text-code> is defined in [RFC3501].

      mimetype-and-params = quoted-to-mime-type
          [SP "(" transcoding-params ")"]
          ; always includes a specific MIME type

      mimetype-list = "(" "(" [quoted-to-mime-type
                               *(SP quoted-to-mime-type)] ")" ")"
          ; Unordered list of MIME types.  It can be empty.
          ;
          ; Two levels of parenthesis is needed to distinguish this
          ; data from <converterror-phrase>.

      converterror-phrase = "(" "ERROR" SP
           convert-err-descript SP convert-error-code ")"

      convert-error-code = "TEMPFAIL" [SP nz-number]
                         / bad-params
                         / missing-params

      convert-err-descript = string
           ; Human-readable text explaining the conversion error.
                    ; The default charset is US-ASCII, unless
                    ; the LANGUAGE command [IMAP-I18N] is called, when
                    ; the charset changes to UTF-8.

      quoted-from-mime-type = DQUOTE from-concrete-mime-type DQUOTE
```

```
      bad-params = "BADPARAMETERS"
             1*(SP (quoted-from-mime-type / nil)
                SP mimetype-and-params)
          ; nil is only returned when the body part doesn't exist.

      missing-params = "MISSINGPARAMETERS"
             1*(SP quoted-from-mime-type SP
                   mimetype-and-missing-params)

      mimetype-and-missing-params = quoted-to-mime-type
          "(" transcoding-param-names ")"
          ; always includes a specific MIME type

      concrete-mime-type = type-name "/" subtype-name
                         ; i.e., "type/subtype".
                         ; type-name and subtype-name
                         ; are defined in [MIME-MTSRP].

      from-concrete-mime-type = concrete-mime-type

      to-mime-type = concrete-mime-type

      command-auth =/ conversions-cmd

      conversions-cmd = "CONVERSIONS" SP from-mime-type-req SP
                        to-mime-type-req

      from-mime-type-req = astring
        ; "mime-type-req" represented as IMAP <atom>,
        ; <quoted> or <literal>

      to-mime-type-req = astring
        ; <mime-type-req> represented as IMAP <atom>,
        ; <quoted> or <literal>.
        ; Note that <mime-type-req> allows for "*",
        ; which is not allowed in <atom>.  Such values must
        ; be represented as <quoted> or <literal>.

      any-mime-type  = "*"

      mime-type-req = any-mime-type /
                      (type-name "/" any-mime-type) /
                      concrete-mime-type
        ; '*', 'type/*' or 'type/subtype'.
        ; type-name is defined in [MIME-MTSRP].

      response-payload =/ conversion-data
```

```
conversion-data = "CONVERSION" SP quoted-from-mime-type SP
                  quoted-to-mime-type
                  [SP "(" transcoding-param-name
                   *(SP transcoding-param-name) ")"]
```

11. Manageability Considerations

The monitoring of CONVERT operation is similar to monitoring of the IMAP FETCH operation.

At the time of writing this document, there is no standard IMAP MIB defined. Similarly, a standard MIB for monitoring CONVERT operations and their failures does not exist. However, the authors believe that in the absence of such a MIB, server implementations SHOULD provide operators with tools to report the following information:

- which conversions (source and target MIME types and possibly conversion parameters used) are invoked more frequently and how long they take,
- information about conversion errors and which error condition caused them (see Section 9), and
- information about users which invoke conversion operation.

This information can help operators to detect client abuse of this extension and scalability issues that might arise from its use.

Standardizing these tools may be the subject of future work.

12. IANA Considerations

IMAP4 capabilities are registered by publishing a Standards Track or IESG-approved Experimental RFC. This document defines the CONVERT IMAP capability. IANA has added this extension to the IANA IMAP Capability registry.

IANA has performed registrations as defined in the following subsections.

12.1. Registration of unknown-character-replacement Media Type Parameter

IANA has added the following registration to the registry established by RFC 2506.

To: "Media feature tags mailing list"
<media-feature-tags@apps.ietf.org>

Subject: Registration of media feature tag
unknown-character-replacement

Media feature tag name:
unknown-character-replacement

ASN.1 identifier associated with feature tag:
1.3.6.1.8.1.33

Summary of the media feature indicated by this feature tag:
Allows servers that can perform charset conversion for text/plain text/html, text/css, text/csv, text/enriched, and text/xml MIME types to replace characters not supported by the target charset with a fixed string, such as "?".
This feature tag is also applicable to other conversions to text, e.g., conversion of images using OCR (optical character recognition).

Values appropriate for use with this feature tag:
The feature tag contains a UTF-8 string used to replace any characters from the source media type that can't be represented in the target media type.

The feature tag is intended primarily for use in the following applications, protocols, services, or negotiation mechanisms:
IMAP CONVERT extension [RFC5259]

Examples of typical use:
C: b001 CONVERT 2 BINARY[3 ("text/plain" ("charset"
"us-ascii" "unknown-character-replacement" "?"))]

Related standards or documents:
[RFC5259]
[CHARSET-REG]

Considerations particular to use in individual applications, protocols, services, or negotiation mechanisms:
None

Interoperability considerations: None

Security considerations: None

Additional information:
This media feature only make sense for MIME types that also support the "charset" media type parameter [CHARSET-REG].

Name(s) & email address(es) of person(s) to contact for further information:
Alexey Melnikov <alexey.melnikov@isode.com>

Intended usage:
COMMON

Author/Change controller:
IETF

Requested IANA publication delay:
None

Other information:
None

13. Security Considerations

It is to be noted that some conversions may present security threats (e.g., converting a document to a damaging executable, exploiting a buffer overflow in a media codec/parser, or a denial-of-service attack against a client or a server such as requesting an image be scaled to extremely large dimensions). Server SHOULD refuse to execute CPU-expensive conversions. Servers should avoid dangerous conversions if possible. Whenever possible, servers should perform verification of the converted attachments before returning them to the client. Clients should be careful when requesting conversions or processing transformed attachments. Clients SHOULD use mutual Simple Authentication and Security Layer (SASL) authentication and the SASL/TLS integrity layer, to make sure they are talking to trusted servers.

When the client requests a server-side conversion of a signed body part (e.g., a part inside multipart/signed), there is no way for the client to verify that the converted content is authentic. A client not trusting the server to perform conversion of a signed body part can download the signed object, verify the signature, and perform the conversion itself.

A client can create a carefully crafted bad message with the APPEND command followed by the CONVERT command to attack the server. If the server's conversion function or library has a security problem (such as vulnerability to a buffer overflow), this could result in privilege escalation or denial of service. In order to mitigate such attacks, servers SHOULD log the client authentication identity on APPEND and/or CONVERT operations in order to facilitate tracking of abusive clients. Also server implementors SHOULD isolate the conversion function or library from the privileged mailstore, perhaps by running it within a distinct process.

Deployments in which the actual transcoding is done outside the IMAP server in a separate server are recommended to keep the servers in the same trusted domain (e.g., subnet).

14. Acknowledgments

Stephane H. Maes and Ray Cromwell from Oracle edited several earlier versions of this document. Their contribution is gratefully acknowledged.

The authors want to specifically acknowledge the excellent criticism and comments received from Randall Gellens (Qualcomm), Arnt Gulbrandsen (Oryx), Zoltan Ordogh (Nokia), Ben Last (Émccsoft), Dan Karp (Zimbra), Pete Resnick (Qualcomm), Chris Newman (Sun), Ted Hardie (Qualcomm), Larry Masinter (Adobe), Philip Guenther (Sendmail), Greg Vaudreuil (Alcatel-Lucent), David Harrington (Comcast), Dave Cridland (Isode), Pasi Eronen (Nokia), Magnus Westerlund (Ericsson), and Jari Arkko (Ericsson), which improved the quality of this specification considerably.

The authors would also like to specially thank Dave Cridland for the MEDIACAPS command proposal and Dan Karp for the CONVERSIONS command proposal.

The authors also want to thank all who have contributed key insight and extensively reviewed and discussed the concepts of CONVERT and its predecessor P-IMAP. In particular, this includes the authors of the LCONVERT document: Rafiul Ahad (Oracle Corporation), Eugene Chiu (Oracle Corporation), Ray Cromwell (Oracle Corporation), Jia-der Day (Oracle Corporation), Vi Ha (Oracle Corporation), Wook-Hyun Jeong (Samsung Electronics Co. LTF), Chang Kuang (Oracle Corporation), Rodrigo Lima (Oracle Corporation), Stephane H. Maes (Oracle Corporation), Gustaf Rosell (Sony Ericsson), Jean Sini (Symbol Technologies), Sung-Mu Son (LG Electronics), Fan Xiaohui (China Mobile Communications Corporation (CMCC)), and Zhao Lijun (China Mobile Communications Corporation (CMCC)).

15. References

15.1. Normative References

[ABNF] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[CHARSET-REG] Hoffman, P., "Registration of Charset and Languages Media Features Tags", RFC 2987, November 2000.

[IMAPABNF] Melnikov, A. and C. Daboo, "Collected Extensions to IMAP4 ABNF", RFC 4466, April 2006.

[MEDIAFEAT-REG] Holtman, K., Mutz, A., and T. Hardie, "Media Feature Tag Registration Procedure", BCP 31, RFC 2506, March 1999.

[MIME-MTSRP] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", BCP 13, RFC 4288, December 2005.

[RFC2047] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2231] Freed, N. and K. Moore, "MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations", RFC 2231, November 1997.

[RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, March 2003.

[RFC3516] Nerenberg, L., "IMAP4 Binary Content Extension", RFC 3516, April 2003.

15.2. Informative References

[DISP-FEATURES] Masinter, L., Wing, D., Mutz, A., and K. Holtman, "Media Features for Display, Print, and Fax", RFC 2534, March 1999.

[IMAP-I18N] Newman, C., Gulbrandsen, A., and A. Melnikov, "Internet Message Access Protocol Internationalization", RFC 5255, June 2008.

[LEM-STREAMING] Cook, N., "Streaming Internet Messaging Attachments", Work in Progress, June 2008.

[OMA-ME-RD] OMA, "Open Mobile Alliance Mobile Email Requirement Document", OMA 55.919 3.0.0, December 2007.

[OMA-STI] OMA, "Open Mobile Alliance, Standard Transcoding Interface Specification", OMA OMA-STI-V1_0, December 2005.

Authors' Addresses

Alexey Melnikov (editor)
Isode Ltd
5 Castle Business Village
36 Station Road
Hampton, Middlesex TW12 2BX
UK

EMail: Alexey.Melnikov@isode.com

Peter Coates (editor)
Sun Microsystems
185 Falcon Drive
Whitehorse, YT Y1A 6T2
Canada

EMail: peter.coates@Sun.COM

Full Copyright Statement

Intellectual Property