

Snippets: An IMAP Extension

IETF 102 Montreal: extra

Michael Slusarz – Product Manager, Dovecot
19 July 2018

Stay Open. **OX**

The Goal

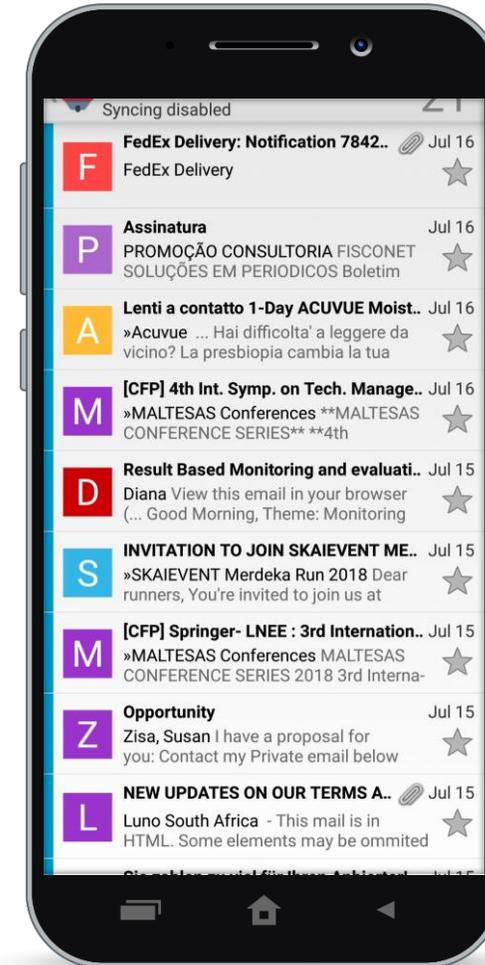
Text previews of messages displayed in mailbox-level UI view

Displaying a small preview of message contents (a **snippet**) is a common UI element of MUAs.

Snippets are generally loaded and displayed when scrolling through the list of messages in a mailbox.

Snippets are used as an aid to the end user about the contents of the message. It can act as a “trigger” to convince a user to open (or not open) the message for full display in situations where a user may not have opened the message otherwise (i.e. commercial messages; potential spam).

Example (at right): K-9 mail (Android)



The Issue(s)

Snippet generation can be expensive and inconsistent

Expensive

Generation requires **multiple protocol round-trips**.

1. FETCH ENVELOPE (to build message list) + FETCH BODYSTRUCTURE
2. [MUA: parse BODYSTRUCTURE to determine “interesting” part]
3. FETCH BODY[interesting_part]<<partial>> (how much should partial be?)

Body retrieval may be **I/O expensive** (e.g. message stored in object storage).

- To prevent UI blocking, BODYSTRUCTURE and/or BODY fetches might need to be issued separately for each message.

Generated snippet is **not globally cached**.

Inconsistent

Snippet may appear differently on user’s clients based on the proprietary way the MUA creates the preview.

The Solution

Server-side snippet generation is more efficient and promotes UI consistency

Efficient

Generation requires **single protocol round-trip**.

1. FETCH ENVELOPE (to build message list) + FETCH SNIPPET
 - If lazy loading snippets, additional FETCH SNIPPETS may be needed, but not blocking on message list generation.

Snippet generation can be done on delivery, so **message body retrieval not needed**.

Generated snippet can be **globally cached**.

Consistent

Snippet will appear the same on all clients (using the same SNIPPET algorithm).

Snippet IMAP Implementation

Technical Details

- Extends IMAP FETCH command
- *[Optional]* Can specify snippet generation **algorithm**
 - 1 algorithm defined in draft: **FUZZY**
 - FUZZY: UTF-8 text/plain representation; no markup; length limitation
 - Absent explicit algorithm selection in FETCH, server decides algorithm to use
- *[Optional]* Can specify snippet generation **modifier**
 - 1 modifier defined in draft: **LAZY**
 - LAZY: Return snippet ONLY if data can be returned “without undue delay to the client”
 - Usage: intended for initial mailbox listing when UI display should not be blocked due to snippet retrieval

Snippet Example

```
C: D1 CAPABILITY
S: * CAPABILITY IMAP4rev1 SNIPPET=FUZZY
S: D1 OK Capability command completed.
[...a mailbox is SELECTed...]
C: D2 FETCH 1:3 (ENVELOPE SNIPPET (LAZY=FUZZY))
S: * 1 FETCH (ENVELOPE ("Wed, 25 Oct 2017 15:03:11 +0000" [...])
  SNIPPET (FUZZY {61})
S: This is the first line of text from the first text part.
S: ))
S: * 2 FETCH (SNIPPET (FUZZY "") ENVELOPE
  ("Thu, 26 Oct 2017 12:17:23 +0000" [...]))
S: * 3 FETCH (ENVELOPE ("Fri, 27 Oct 2017 22:19:21 +0000" [...])
  SNIPPET (FUZZY NIL))
S: D2 OK FETCH completed.
[...Client knows that message 2 has a snippet that is empty;
  therefore, client only needs to request message 3 snippet again
  (e.g. in background)...]
C: D3 FETCH 3 (SNIPPET (FUZZY))
S: * 3 FETCH (SNIPPET (FUZZY {25})
S: First sentence of mail 3.
S: ))
S: D3 OK Fetch completed.
```

Scenario:

Use explicit algorithm priority selection, with LAZY modifier, to obtain snippets during initial mailbox listing if readily available.

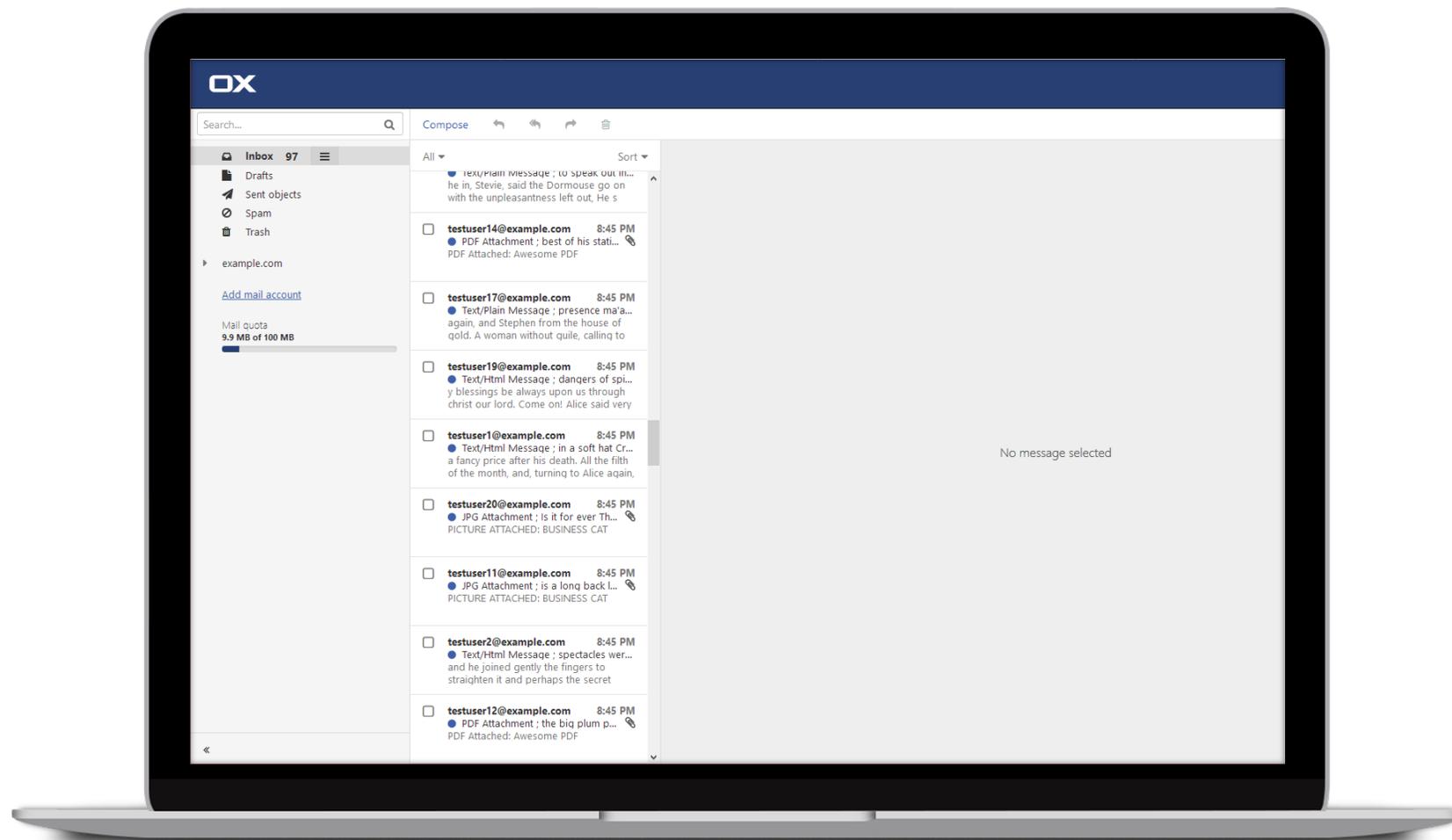
Otherwise, load snippets in background.

Implementation Status

As of July 2018

Snippet retrieval, implemented as per the RFC draft, was added to **Dovecot v2.2.34** (February 2018).

Client-side support was added to **Open-Xchange App Suite v7.10.0** (July 2018).



Open-Xchange GmbH

Olper Huette 5F
D-57462 Olpe

Phone: +49 2761-8385-0
Fax: +49 2761-8385-30

info@open-xchange.com
www.open-xchange.com



Stay Open. **OX**