XEP-0449: Stickers

Marvin Wißfeld
mailto:xmpp@larma.de
xmpp:jabber@larma.de

2022-02-10
Version 0.1.1

Status	Type	Short Name
Experimental	Standards Track	stickers

This specification provides a protocol to send stickers and to create and share sticker packs.

Legal

Copyright

This XMPP Extension Protocol is copyright © 1999 – 2024 by the XMPP Standards Foundation (XSF).

Permissions

Permission is hereby granted, free of charge, to any person obtaining a copy of this specification (the "Specification"), to make use of the Specification without restriction, including without limitation the rights to implement the Specification in a software program, deploy the Specification in a network service, and copy, modify, merge, publish, translate, distribute, sublicense, or sell copies of the Specification, and to permit persons to whom the Specification is furnished to do so, subject to the condition that the foregoing copyright notice and this permission notice shall be included in all copies or substantial portions of the Specification. Unless separate permission is granted, modified works that are redistributed shall not contain misleading information regarding the authors, title, number, or publisher of the Specification, and shall not claim endorsement of the modified works by the authors, any organization or project to which the authors belong, or the XMPP Standards Foundation.

Warranty

NOTE WELL: This Specification is provided on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE.

Liability

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall the XMPP Standards Foundation or any author of this Specification be liable for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising from, out of, or in connection with the Specification or the implementation, deployment, or other use of the Specification (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if the XMPP Standards Foundation or such author has been advised of the possibility of such damages.

Conformance

This XMPP Extension Protocol has been contributed in full conformance with the XSF's Intellectual Property Rights Policy (a copy of which can be found at <https://xmpp.org/about/xsf/ipr-policy> or obtained by writing to XMPP Standards Foundation, P.O. Box 787, Parker, CO 80134 USA).

Contents

1 Introduction

Stickers[1] are images or short animations that often represent an emoticon or an action. They are increasingly used in private messaging in the last years, especially on mobile phone apps. Stickers can be more nuanced and expressive than emoticons. They can be used to convey emotions or tell stories in a fun and easy way.

Recently, demand to implement stickers, including their social feature of sharing sticker packs arised within the XMPP community. This specification is to fulfill this demand.

2 Requirements

- Do not require additional server components for easier deployment

- Compatibility with non-graphical user interfaces (terminals, etc)

3 Glossary

Sticker Emoticon-like picture

Sticker pack Collection of stickers, typically devoted to a specific theme or with all stickers created by the same author

4 Use Cases

4.1 Creating a sticker pack

Sticker packs are stored on pubsub nodes. They SHOULD be located at a personal eventing (PEP) node named "urn:xmpp:stickers:0" but MAY be located at a generic publish-subscribe node that is not attached to a user's IM account. The pubsub node's access model SHOULD be set to 'open', so that other users can fetch sticker packs.

4.1.1 Sticker pack pubsub item

Each sticker pack is one item on the pubsub node consisting of a <pack/> element.

The <pack/> element includes elements <name/> and <summary/> to announce a name and summary of the sticker pack that can be displayed to users, respectively. The summary MAY contain copyright and license information in user-readable format. Multiple <name/> and <summary/> elements MAY be included given that no two elements of same name specify the same xml:lang.

The <pack/> element MAY include a <restricted/> element to indicate that this sticker pack

---

[1]*Sticker (messaging)* on Wikipedia <https://en.wikipedia.org/wiki/Sticker_(messaging)>.

1

may not be used or imported by other users.

The <pack/> element contains one or more <item/> elements that include details on the stickers in the pack. Each <item/> element has <file/> element as described in File metadata element (XEP-0446) [2]. The metadata MUST include one or multiple <hash/> elements as described in Use of Cryptographic Hash Functions in XMPP (XEP-0300) [3]. All stickers in a pack MUST use the same algorithm for their <hash/> elements. The metadata MUST include one <desc/> element that is not qualified by a xml:lang attribute and describes the textual fallback (often an emoji or a sequence of emojis) of the sticker. Each <item/> element further has a <sources/> element as described in Stateless file sharing (XEP-0447) [4] describing how to retrieve the image file of the sticker. Each <item/> element MAY include one or more <suggest/> elements that include a text that clients may replace with the sticker. <suggest/> elements MAY have an xml:lang attribute.

Finally, the <pack/> element includes a single <hash/> element as described in Use of Cryptographic Hash Functions in XMPP (XEP-0300) [5], specifying the hash of the sticker pack. See next section on how such hash is generated.

4.1.2 Sticker pack hash calculation

Each sticker pack has a unique ID, which is used as the pubsub item's id. This ID is derived from a hash of significant information from a <pack/> element as outlined in this section.

1. Processing <name/> and <summary/> elements:

    a) For each <name/> or <summary/> element:

        i. Encode the character data of element name (name or summary), xml:lang attribute value and content.

        ii. Append an octet of value 0x1f (ASCII Unit Separator) to each resulting octet string.

        iii. Join the resulting octet strings together, in the order of element name, xml:lang attribute and content, resulting in a single octet string for the <name/> or <summary/> element.

        iv. Append an octet of value 0x1e (ASCII Record Separator).

    b) Join the resulting octet strings together, ordered from lesser to greater.

    c) Append an octet of value 0x1c (ASCII File Separator).

---

[2]XEP-0446: File metadata element <https://xmpp.org/extensions/xep-0446.html>.

[3]XEP-0300: Use of Cryptographic Hash Functions in XMPP <https://xmpp.org/extensions/xep-0300.html>.

[4]XEP-0447: Stateless file sharing <https://xmpp.org/extensions/xep-0447.html>.

[5]XEP-0300: Use of Cryptographic Hash Functions in XMPP <https://xmpp.org/extensions/xep-0300.html>.

The result of this step is referenced as *Meta string* later.

2. Processing <item/> elements:

   a) For each <item/> element:

      i. Encode the character data of <file/>'s <desc/> content.
      ii. Append an octet of value 0x1e (ASCII Record Separator) to the resulting octet string.
      iii. For each of <hash/> element in <file/>:

         A. Encode the character data of algo attribute and content.
         B. Append an octet of value 0x1f (ASCII Unit Separator) to the resulting octet strings.
         C. Join the resulting octet strings together, in the order of algo attribute and content, resulting in a single octet string for the <hash/> element.
         D. Append an octet of value 0x1e (ASCII Record Separator).

      iv. Join the resulting octet strings together, in the order of <file/>'s <desc/> content first and then all <hash/> elements ordered from lesser to greater.
      v. Append an octet of value 0x1d (ASCII Group Separator).

   b) Join the resulting octet strings together, ordered from lesser to greater.
   c) Append an octet of value 0x1c (ASCII File Separator).

   The result of this step is referenced as *Stickers string* later.

3. Join the *Meta String* and *Stickers String* together, in this order.

4. Hash the resulting string using the algorithm choosen for the sticker pack.

The ID of the sticker pack is dervied by taking the first 144 bit of the hash and encode it using Base64. This is equivalent to the first 24 characters of the Base64 encoded content of the <hash/> element.

Listing 1: Publish a new sticker pack

```
<iq from='romeo@montague.lit/pda'
    id='pub1'
    to='romeo@montague.lit'
    type='set'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <publish node='urn:xmpp:stickers:0'>
```

```
        <item id='EpRv28DHHzFrE4zd+xaNpVb4'>
          <pack xmlns='urn:xmpp:stickers:0'>
            <name>Marsey the Cat</name>
            <summary>Be cute or be cynical, this little kitten works
                both ways.</summary>
            <item>
              <file xmlns='urn:xmpp:file:metadata:0'>
                <media-type>image/png</media-type>
                <desc> </desc>
                <size>71045</size>
                <dimensions>512x512</dimensions>
                <hash xmlns='urn:xmpp:hashes:2' algo='sha-256'>0
                    AdP8lJOWJrugSKOIAqfEKqFatIpG5JBCjjxY253ojQ=</hash>
              </file>
              <sources xmlns='urn:xmpp:sfs:0'>
                <url-data xmlns='http://jabber.org/protocol/url-data'
                    target='https://download.montague.lit/51078299-d071
                    -46e1-b6d3-3de4a8ab67d6/sticker_marsey_thumbs_up.png
                    ' />
              </sources>
              <suggest>+1</suggest>
            </item>
            <item>
              <file xmlns='urn:xmpp:file:metadata:0'>
                <media-type>image/png</media-type>
                <desc> </desc>
                <size>67016</size>
                <dimensions>512x512</dimensions>
                <hash xmlns='urn:xmpp:hashes:2' algo='sha-256'>gw+6
                    xdCgOcvCYSKuQNrXH33lV9NMzuDf/s0huByCDsY=</hash>
              </file>
              <sources xmlns='urn:xmpp:sfs:0'>
                <url-data xmlns='http://jabber.org/protocol/url-data'
                    target='https://download.montague.lit/51078299-d071
                    -46e1-b6d3-3de4a8ab67d6/sticker_marsey_kiss.png' />
              </sources>
            </item>
            <!-{}- ... -{}->
            <hash xmlns='urn:xmpp:hashes:2' algo='sha-256'>
                EpRv28DHHzFrE4zd+xaNpVb4jbu4s74XtioExNjQzZ0=</hash>
          </pack>
        </item>
      </publish>
    </pubsub>
</iq>
```

4.2 Sending a sticker

The sending entity uses Stateless file sharing (XEP-0447) [6] to send the image file from the sticker pack. The <item/>'s <file/> and <sources/> elements from the sticker pack are used as a template. The sending entity MAY add additional <hash/> elements or sources.

Additionally, the sending entity adds an element <sticker/> to the message. This element carries an attribute pack referring to the id of the pubsub item carrying the sticker pack. If the sticker pack resides on a pubsub item other than the senders personal eventing (PEP) node named "urn:xmpp:stickers:0", the sending entity must add additional attributes jid and node, referring to the jid of the pubsub node and the name of the node, respectively.

The sending entity SHOULD use the <file/>'s <desc/> content as the message body, except if the sticker was selected based on a suggestion, in which case the content of the corresponding <suggest/> SHOULD be used.

Listing 2: Sending a sticker

```
<message to='juliet@shakespeare.lit' from='romeo@montague.lit/pda' id=
    'sharing-a-file'>
  <body> </body>
  <sticker xmlns='urn:xmpp:stickers:0' pack='EpRv28DHHzFrE4zd+xaNpVb4'
      />
  <file-sharing xmlns='urn:xmpp:sfs:0'>
    <file xmlns='urn:xmpp:file:metadata:0'>
      <media-type>image/png</media-type>
      <desc> </desc>
      <size>67016</size>
      <dimensions>512x512</dimensions>
      <hash xmlns='urn:xmpp:hashes:2' algo='sha-256'>gw+6
          xdCgOcvCYSKuQNrXH33lV9NMzuDf/s0huByCDsY=</hash>
    </file>
    <sources>
      <url-data xmlns='http://jabber.org/protocol/url-data' target='
          https://download.montague.lit/51078299-d071-46e1-b6d3-3
          de4a8ab67d6/sticker_marsey_kiss.png' />
    </sources>
  </file-sharing>
</message>
```

4.3 Receiving a sticker

When receiving a message with a <sticker/> element, the receiving entity SHOULD consider the message and the <file-sharing/> element as a sticker.

When receiving a sticker, the receiving entity may display the stickers image file inline or display the textual representation in the <desc/> element. The receiving entity SHOULD NOT

---

[6]XEP-0447: Stateless file sharing <https://xmpp.org/extensions/xep-0447.html>.

display the received sticker only as a downloadable file.

4.4 Importing a sticker pack

When receiving a sticker or any other reference to a sticker pack, the receiving entity may allow the user to import such sticker pack. The receiving entity SHOULD NOT offer to import a sticker pack that has a <restricted/> element.

When importing a sticker pack, the entity SHOULD make all files that are part of the sticker pack locally available. The entity SHOULD duplicate the pubsub item from its original source to the users personal eventing (PEP) node named "urn:xmpp:stickers:0", preserving the pack id. The entity MAY remove any of the sources provided, and MAY make an effort to make the files available at additional sources, e.g. by using HTTP File Upload (XEP-0363) [7].

4.5 Sharing a sticker pack

After creating or importing a sticker pack, it can be shared using an xmpp:-uri as described in Publish-Subscribe (XEP-0060) [8].

When handling this URI, the entity SHOULD display a preview of the sticker pack and allow importing it as described above.

Listing 3: Sticker pack URI

```
xmpp:romeo@montague.lit?pubsub;action=retrieve;node=
    urn:xmpp:stickers:0;item=EpRv28DHHzFrE4zd%2BxaNpVb4
```

5 Accessibility Considerations

Flickering, flashing or blinking stickers can induce seizures or other incapacitating physical reactions. It is RECOMMENDED that clients that support stickers have an option to disable animations and/or stickers entirely.

6 Security Considerations

The security considerations of Stateless file sharing (XEP-0447) [9] apply.

---

[7]XEP-0363: HTTP File Upload <https://xmpp.org/extensions/xep-0363.html>.
[8]XEP-0060: Publish-Subscribe <https://xmpp.org/extensions/xep-0060.html>.
[9]XEP-0447: Stateless file sharing <https://xmpp.org/extensions/xep-0447.html>.

7 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [10].

8 XMPP Registrar Considerations

8.1 Protocol Namespaces

The XMPP Registrar [11] includes 'urn:xmpp:stickers:0' in its registry of protocol namespaces (see <`https://xmpp.org/registrar/namespaces.html`>).

- urn:xmpp:stickers:0

---

[10]The Internet Assigned Numbers Authority (IANA) is the central coordinator for the assignment of unique parameter values for Internet protocols, such as port numbers and URI schemes. For further information, see <`http://www.iana.org/`>.

[11]The XMPP Registrar maintains a list of reserved protocol namespaces as well as registries of parameters used in the context of XMPP extension protocols approved by the XMPP Standards Foundation. For further information, see <`https://xmpp.org/registrar/`>.