XEP-0481: Content Types in Messages

Peter Waher
mailto:peterwaher@hotmail.com
xmpp:peter.waher@jabber.org
http://www.linkedin.com/in/peterwaher

2023-05-04
Version 0.1.0

Status	**Type**	**Short Name**
Experimental	Standards Track	content

This specification describes a generic method whereby content in messages can be tagged as having a specific Internet Content Type. It also provides a method for sending the same content using different content types, as a fall-back mechanism when communicating between clients having different content type support.

Contents

1 Introduction **1**

2 Use Cases **1**

2.1 Content type hint 1
2.2 Alternate encoding 1
2.3 Alternate encodings 2

3 Determining Support **3**

4 Implementation Notes **3**

4.1 Content Types 3
4.2 Custom Content Types 4
4.3 Stanza size 4

5 IANA Considerations **4**

6 XMPP Registrar Considerations **4**

7 XML Schema **4**

1 Introduction

Sometimes it is desirable for clients to communicate, or to send messages between each other, using a well defined Internet Content Type. Reasons can vary between the purely esthetic to funcional data-exchange. While there are XEPs, such as XHTML-IM (XEP-0071) [1] that provides means for sending richer content using a specific type, this extensions provides a similar mechanism, but for the general case of any content having a defined Internet Content Type.
Note: While the examples in this extensions uses Markdown as an example, any other text-based content type can be used.

2 Use Cases

2.1 Content type hint

The simplest use case is hinting at the content type of the textual content presented in the message body. This is done by aggregating a **content** element of namespace **urn:xmpp:content** to the message, with the attribute **type** specifying the content type. If the element does not provide a value, it is understood that the body contains the textual body of the content. This method should only be used if there's no risk of misunderstanding the message if the content type is not understood by the receiver, and the textual representation is readable. Example:

Listing 1: Hinting at a content type

```
<message
        from='person1@example.org/34892374'
        to='person2@example.org/938089023'
        type='chat'>
       <body>**Note:** This message is very important.</body>
       <content type='text/markdown' xmlns='urn:xmpp:content'/>
    </message>
```

2.2 Alternate encoding

If there is a risk of misunderstanding the message if it's content type is not recognized, or the presentation of the message is done in an undesireable fashion, you can provide an alternate encoding of the message in the **content** element itself. If the **content** element contains a message, and the content type is recognized, the message should be taken from the **content** element instead of the **body** element. The **body** element in turn, should contain the **plain text** version of the same message. Example:

Listing 2: Alternate encoding

[1] XEP-0071: XHTML-IM <https://xmpp.org/extensions/xep-0071.html>.

```
<message
        from='person1@example.org/34892374'
        to='person2@example.org/938089023'
        type='chat'>
       <body>Note: Go to Google and search for it.</body>
       <content type='text/markdown' xmlns='urn:xmpp:content'>
          **Note:** Go to [Google](http://www.google.com/) and search
             for it.
       </content>
    </message>
```

2.3 Alternate encodings

By providing multiple **content** elements in the same message, you can allow the receiver to choose the encoding best suited for its purpose. It also makes it possible to interchange messages that are understood by both humans and machines in the same message. If an empty **content** element is found, it is interpreted as above, i.e. providing a hint as to the content type of the message in the **body** element. Example:

Listing 3: Alternate encodings

```
<message
        from='person1@example.org/34892374'
        to='person2@example.org/938089023'
        type='chat'>
       <body>
          Your energy consumption this month is 5000 kWh.
          That is very much. It will cost you 200 USD.
          You can find current tariffs at our web page.
       </body>
       <content type='text/markdown' xmlns='urn:xmpp:content'>
          Your energy consumption this month is **5000 kWh**.
          That is *very much*. It will cost you **200 USD**.
          You can find current tariffs at our [web page](http://www.
             example.com/Energy).
       </content>
       <content type='text/xml' xmlns='urn:xmpp:content'>
          &lt;Quote xmlns='somenamespace'&gt;
             &lt;Consumption unit='kWh'&gt5000&lt;/Consumption&gt;
             &lt;Cost unit='USD'&gt200&lt;/Cost&gt;
          &lt;/Quote&gt;
       </content>
    </message>
```

3 Determining Support

If an entity supports content types as specified herein, it MUST advertise that fact by returning a feature of "urn:xmpp:content" in response to Service Discovery (XEP-0030) [2] information requests.

Listing 4: Service discovery information request

```
<iq type='get'
        from='example.org'
        to='device@example.org'
        id='disco1'>
      <query xmlns='http://jabber.org/protocol/disco#info'/>
    </iq>
```

Listing 5: Service discovery information response

```
<iq type='result'
        from='device@example.org'
        to='example.org'
        id='disco1'>
      <query xmlns='http://jabber.org/protocol/disco#info'>
        ...
        <feature var='urn:xmpp:content'/>
        ...
      </query>
    </iq>
```

In order for an application to determine whether an entity supports this protocol, where possible it SHOULD use the dynamic, presence-based profile of service discovery defined in Entity Capabilities (XEP-0115) [3]. However, if an application has not received entity capabilities information from an entity, it SHOULD use explicit service discovery instead.

4 Implementation Notes

4.1 Content Types

This document does not specify how content types are to be interpreted, or if content types are valid or well defined. It does not specify which content types are to be understood, or when. It only provides a means to hint or include different encodings in the same message.

[2]XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.
[3]XEP-0115: Entity Capabilities <https://xmpp.org/extensions/xep-0115.html>.

4.2 Custom Content Types

It is possible to use custom or vendor-specific content types. These types are marked by prefixing the subtype with **x.** for custom unregistered types, and with **vnd.** for registered vendor specific types.

4.3 Stanza size

Care has to be taken when sending multiple encodings of the same message, as to not reach the smallest allowed maximum stanza size used by client and server software.

5 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [4].

6 XMPP Registrar Considerations

The protocol schema needs to be added to the list of XMPP protocol schemas.

7 XML Schema

```
<?xml version='1.0' encoding='UTF-8'?>
<!--
© XMPP Standards Foundation, 2016
Author: Peter Waher
-->
<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:xmpp:content'
    xmlns='urn:xmpp:content'
    elementFormDefault='qualified'>

    <xs:element name='content'>
        <xs:complexType mixed='true'>
            <xs:attribute name='type' use='required'/>
        </xs:complexType>
```

[4] 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/>.

```
    </xs:element>

</xs:schema>
```