

Programming with **XMPP** and **JavaScript**

An Introduction to **Strophe.js**

Jack Moffitt



XMPP

What is **Strophe**?

XMPP client library

JavaScript

Real time Web applications

Fully documented

Highly optimized

Well tested

Built for Chesspark

StanzIQ and Speeqe

also used by
Seismic
Yammer
Neuros OSD

First steps

Managing **connections**

Connecting

```
var connection =  
new Strophe.Connection(URL);
```

```
connection.connect(  
    jid,  
    password,  
    callback  
);
```

user@domain

Strophe lets server
assign resource

user@domain/resource

Strophe requests a specific
resource

domain or domain/resource

Strophe will try
SASL ANONYMOUS

The connection **callback**

Strophe reports status

connecting
authenticating
authentication failed
connected
disconnecting
disconnected

```
function on_status(status) {  
    if (status == Strophe.Status.CONNECTED) {  
        // send initial presence  
        // query for the roster  
    }  
}
```

Sending data

```
connection.send(xml);
```

Disconnecting

```
connection.disconnect();
```

All about **events**

Event driven

Interaction events

Timed events

Stanza events

Examples

User clicks send button

```
$('#send').click(function () {  
    // build message stanza  
    // send message  
});
```

Display **incoming messages**

Add a handler

```
connection.addHandler(  
    on_message,  
    null,  
    "message",  
    "chat"  
);
```

Respond to matched stanzas

```
function on_message(msg) {  
    // extract message body  
    // display text  
  
    return true;  
}
```

Dealing with **IQ stanzas**

Answering incoming IQs

```
connection.addHandler(  
    on_iq_version,  
    “jabber:iq:version”,  
    “iq”,  
    “get”  
);
```

Getting responses

```
connection.addHandler(  
    on_version,  
    null,  
    “iq”,  
    null,  
    “disco-1”  
);
```

Timed handlers

```
connection.addTimedHandler(  
    100,  
    send_flood  
);
```

Building stanzas

Strophe.**Builder**

Almost always returns
Strophe.Builder

Allows function chaining
just like jQuery

```
var stanza = new Strophe.Builder(  
    "message",  
    {"to": "someone@jabber.org",  
     "type": "chat"}  
);
```

Chainable methods

Adding a child

`c(name, attrs)`

Adding text content

```
t("some text")
```

Adding pre-made children

`cnode(element)`

Modifying attributes

```
attrs(new_attrs)
```

Traversing the tree

up()

Examples

```
new Strophe.Builder(  
    "message",  
    {"to": "someone@jabber.org",  
     "type": "chat"}  
).c("body").t("Hello, World!");
```

```
new Strophe.Builder(  
    "message",  
    {"to": "...", "type": "chat"}  
).c("body").t("Hi")  
.up()  
.c("html",  
    {"xmlns": ".../xhtml-im"})  
.c("body", ...)  
.c("p").t("Hi")
```

Convenience functions

\$pres(attrs)

\$msg(attrs)

\$iq(attrs)

Send available presence

`$pres()`

Build a message

```
$msg({"to": "someone@jabber.org",  
      "type": "chat"})  
.c("body").t("XMPP rocks!")
```

Unchainable methods

calling `stanza.toString()` produces

```
“<message to='someone@jabber.org'  
  type='chat'/>”
```

`stanza.tree()` produces a DOM tree

Hello, Server!

an application

The **Future** of Strophe

XPath matching

Sub-protocol support

Plugin system

<http://code.stanziq.com/strophe>

<http://metajack.im>

jack@metajack.im