

Magma JSON Interface

Description: This document summarizes the JSON interface for the Magma email platform.

Version: 6.0.1

Last Updated: Tue, 23 Dec 2014 04:45:41 -0600

Assumptions:

- User is identified with cookie a or optional sessionID in all requests.
- Responses for tables send all fields.
- No assumption is made about the order of items in a response unless specified otherwise.

Document MO:

- Some parameters with a designated set of options are listed as an ordered list.
- Optional parameters are indicated by surrounding param with square brackets.
 - e.g. ["type"]
- All parameters are required unless specified as optional.
- The default request value (*def*) is assumed by server if that parameter is not provided in the request.
- Multiple example requests map to response successes unless specified otherwise.
- Methods with no request params simply leave out the parameter summary and example requests.

Method: ad

Gets advertisement and fun facts for loading screen and ad banner for messages.

Request Params

["context"] - *where the ad will be displayed*

1. "loading"
2. "messages" (*def*)

Example Request

```
{
  "id": 1,
  "method": "ad",
  "params": {
    "context": "loading"
  }
}
```

Response Params

"ad"

- "href" - *href of advertisement*
 - string
- ["title"] - *link title*
 - string
- ["img"] - *if image ad*
 - "src"
 - string
 - "alt"
 - string
- ["text"] - *if text ad*
 - string

["fact"]

- "img"
 - "src"
 - string
 - "alt"
 - string
- "text"
 - array of strings

Example Response

```
[ {
  "result": {
    "ad": {
```

```

        "href": "/advertisement.html",
        "title": "Loading Ad",
        "img": {
            "src": "http://placeholder.it/300x250",
            "alt": "Loading Ad"
        }
    },
    "fact": {
        "img": {
            "src": "http://placeholder.it/300x50",
            "alt": "Fun picture"
        },
        "text": [
            "Most elephants weigh less than the tongue of a blue whale.",
            "A kangaroo can only jump if its tail is touching the ground.",
            "In a study of 200,000 ostriches over a period of 80 years, no one reported a single case where an ostrich buried
        ]
    }
}, {
    "result": {
        "ad": {
            "href": "/advertisement.html",
            "title": "Wide Advertisement",
            "img": {
                "src": "http://placeholder.it/728x90",
                "alt": "Wide Advertisement"
            }
        }
    }
}, {
    "result": {
        "ad": {
            "href": "/text-ad.html",
            "title": "Text Ad",
            "text": "This is a text ad"
        }
    }
}, {
    "error": true
}]

```

Method: alert.acknowledge

Javascript will pull server for alerts at an interval.

Request Params

If alertIDs are included in request it is an awk.

["alertIDs"] - *ack alerts*

- array of ints

Example Request

```

{
  "id": 2,
  "method": "alert.acknowledge",
  "params": {
    "alertID": [1,2]
  }
}

```

Example Response

```

{
  "jsonrpc": "2.0",
  "result": {
    "alert.acknowledge": "success"
  },
  "id": 3
}

```

Method: alert.list

Javascript will pull server for alerts at an interval.

Response Params

"alertID" - *allows for ack*

- integer

"type"

1. "alert" - (*def*)
2. "warning"

"message"

- string

"utc"

- unix time

Example Response

```
[{
  "result": [{
    "alertID": 1,
    "type": "alert",
    "message": "It's a trap!",
    "date": "1029381092830"
  }, {
    "alertID": 2,
    "type": "warning",
    "message": "Your subscription will expire soon.",
    "date": "12093810293801"
  }]
}, {
  "result": true
}, {
  "error": true
}]
```

Method: aliases

Fills in composing from field dropdown.

Response Params

"email"

- string

"name"

- string

["def"] - indicates which to display by default (default keyword reserved by javascript)

- bool

Example Response

```
[{
  "result": [{
    "email": "peter@lavabit.com",
    "name": "Peter"
  }, {
    "email": "work@lavabit.com",
    "name": "Work",
    "def": true
  }, {
    "email": "peter@pan.com",
    "name": "Website"
  }, {
    "email": "email@example.com",
    "name": "Test Email"
  }]
}, {
  "error": true
}]
```

Method: attachments.add

Sent when the user chooses a file to attach.

Request Params

"composeID" - provided by server when user starts composing a message

- integer

"filename"

- string

Example Request

```
{
  "id": 5,
  "method": "attachments.add",
  "params": {
    "composeID": 45,
    "filename": "script.js"
  }
}
```

Response Params

"attachmentID"

- integer

Example Response

```
[{
  "result": {
    "attachmentID": 34
  }, {
    "error": true
  }]
```

Method: attachments.progress

Pulls server for progress of file upload.

Request Params

"attachmentID"

- integer

Example Request

```
{
  "id": 6,
  "method": "attachments.progress",
  "params": {
    "attachmentID": 34
  }
}
```

Response Params

"progress" - 0 to 100 indicating percent complete

- integer

["size"] - in bytes

- integer

["rate"]

- string

Example Response

```
[{
  "result": {
    "progress": 54
  }, {
    "error": true
  }]
```

Method: attachments.remove

Removes an attachment from a message.

Request Params

"attachmentID"

- integer

Example Request

```
{
  "id": 7,
  "method": "attachments.remove",
  "params": {
    "attachmentID": 34
  }
}
```

Example Response

```
[{
  "result": true
}, {
  "error": true
}]
```

Method: auth

Sent when user submits login form. In this case, the example requests correspond to the errors respectively.

Request Params

"username"

- string

"password"

- string

Example Request

```
{
  "id": 8,
  "method": "auth",
  "params": {
    "username": "magma",
    "password": "test"
  }
}
```

Response Params

"auth" - string 1. "success" 1. "failed" 1. "locked"

- *sessionID* sent when successfully authenticated

- string

- *error message*

- string

Example Response

```
[{
  "result": {
    "auth": "success",
    "session": ""
  }
}, {
  "result": {
    "auth": "failed",
    "message": "The username and password provided are incorrect, please try again."
  }
}, {
  "result": {
    "auth": "locked",
    "message": "Your account is locked due to abuse."
  }
}]
```

Method: contacts.add

Gets a contactID from server for subsequent communication.

Response Params

"contactID" - *key from server*

- integer

Example Response

```
[{
  "result": {
    "contactID": 88
  }, {
    "error": true
  }]
```

Method: contacts.edit

Change a contact's info. This method has several parameters that are optional since it only sends the updated fields. The response sends back all the fields that are set on the server.

Request Params

"contactID"

- integer

["name"] - *full name of contact*

- string

["email"]

- ["primary"]
 - string
- ["alternate-1"]
 - string
- ["alternate-2"]
 - string

["chat"] - *IM/XMPP/IRC handles*

- ["yahoo"]
 - string
- ["live"]
 - string
- ["aim"]
 - string
- ["google"]
 - string
- ["icq"]
 - string

["phone"]

- ["home"]
 - string
- ["work"]
 - string
- ["mobile"]
 - string
- ["pager"]
 - string
- ["fax"]
 - string

["address"]

- ["home"]
 - string
- ["work"]
 - string

Example Request

```
{
  "id": 8,
  "method": "contacts.edit",
  "params": {}
}
```

Response Params

"contactID"

- integer

["name"] - *full name of contact*

- string

["email"]

- ["primary"]
 - string
- ["alternate-1"]
 - string
- ["alternate-2"]
 - string

["chat"] - *IM/XMPP/IRC handles*

- ["yahoo"]
 - string
- ["live"]
 - string
- ["aim"]
 - string
- ["google"]
 - string
- ["icq"]
 - string

["phone"]

- ["home"]
 - string
- ["work"]
 - string
- ["mobile"]
 - string
- ["pager"]
 - string
- ["fax"]
 - string

["address"]

- ["home"]
 - string
- ["work"]
 - string

Example Response

```
[{
  "result": {
    "name": "Peter Pan",
    "email": {
      "primary": "peter@lavabit.com",
      "alternate-1": "peter@pan.com"
    },
    "chat": {
      "yahoo": "pp@yahoo.com",
      "live": "pp@hotmail.com",
      "aim": "peanutbutter",
      "google": "pp@gmail.com"
    },
    "phone": {
      "home": "123-456-7890",
      "work": "123-456-7890",
      "mobile": "123-456-7890",
      "address": ["1234 Lost Way", "Neverland Isl"]
    }
  }, {}
}, {
  "error": true
}]
```

Method: contacts.list

Summarize contact info in a table.

Request Params

"folderID"

- integer

Example Request

```
{
  "id": 9,
  "method": "contacts.list",
  "params": {
    "folderID": 3
  }
}
```

Response Params

"contactID"

- integer

"name" - *contact's full name*

- string

"email"

- string

["company"]

- string

["img"] - *avatar*

- "src"
 - href
- "alt"
 - string

Example Response

```
[{
  "result": [{
    "contactID": 45,
    "name": "John Resig",
    "email": "jr@mozilla.org",
    "company": "Mozilla",
    "img": {
      "src": "http://www.gravatar.com/avatar/00000000000000000000000000000000?s=32&d=mm&f=y",
      "alt": "Avatar"
    }
  }], {
    "contactID": 48,
    "name": "Linus Torvalds",
    "email": "lt@kernel.org",
    "company": "Linux"
  }, {
    "contactID": 53,
    "name": "Douglas Crockford",
    "email": "dc@yahoo.com",
    "company": "Yahoo",
    "img": {
      "src": "http://www.gravatar.com/avatar/00000000000000000000000000000000?s=32&d=mm&f=y",
      "alt": "Avatar"
    }
  }], {
    "contactID": 58,
    "name": "Paul Graham",
    "email": "pg@ycombinator.com",
    "company": "Y Combinator",
    "img": {
      "src": "http://www.gravatar.com/avatar/00000000000000000000000000000000?s=32&d=mm&f=y",
      "alt": "Avatar"
    }
  }], {
    "contactID": 64,
    "name": "John Paul",
    "email": "jpm@morgan.com"
  }
}], {
  "result": [{
    "contactID": 45,
    "name": "John Resig",
    "email": "jr@mozilla.org",
    "company": "Mozilla",
    "img": {
      "src": "http://www.gravatar.com/avatar/00000000000000000000000000000000?s=32&d=mm&f=y",
      "alt": "Avatar"
    }
  }], {
    "contactID": 48,
    "name": "Linus Torvalds",
    "email": "lt@kernel.org",
    "company": "Linux"
  }, {
    "contactID": 53,
    "name": "Douglas Crockford",
    "email": "dc@yahoo.com",
    "company": "Yahoo",
    "img": {
      "src": "http://www.gravatar.com/avatar/00000000000000000000000000000000?s=32&d=mm&f=y",
      "alt": "Avatar"
    }
  }], {
    "contactID": 58,
    "name": "Paul Graham",
    "email": "pg@ycombinator.com",
    "company": "Y Combinator",
    "img": {
      "src": "http://www.gravatar.com/avatar/00000000000000000000000000000000?s=32&d=mm&f=y",
      "alt": "Avatar"
    }
  }], {
    "contactID": 64,
    "name": "John Paul",
    "email": "jpm@morgan.com"
  }
}]
```



```

        "contactID": 48,
        "name": "Linus Torvalds",
        "email": "lt@kernel.org",
        "company": "Linux"
    }, {
        "contactID": 53,
        "name": "Douglas Crockford",
        "email": "dc@yahoo.com",
        "company": "Yahoo",
    }, {
        "contactID": 58,
        "name": "Paul Graham",
        "email": "pg@ycombinator.com",
        "company": "Y Combinator",
    }, {
        "contactID": 64,
        "name": "John Paul",
        "email": "jp@morgan.com"
    }
  ]
}, {
  "error": true
}]

```

Method: contacts.load

Load the contact info for a particular contact.

Request Params

"contactID"

- integer

Example Request

```

{
  "id": 10,
  "method": "contacts.load",
  "params": {
    "contactID": 88
  }
}

```

Response Params

"name"

- string

"email"

- "primary"
 - string
- ["alternate1"]
 - string
- ["alternate2"]
 - string

["chat"]

- ["yahoo"]
 - string
- ["live"]
 - string
- ["aim"]
 - string
- ["google"]
 - string

["contact"]

- ["home"]
 - string
- ["work"]
 - string
- ["mobile"]
 - string
- ["address"] - *include for new lines*

- strings

Example Response

```
[{
  "result": {
    "name": "Douglas Crockford",
    "email": {
      "primary": "dougrock@lavabit.com",
      "alternatel": "dc@crockford.com"
    },
    "chat": {
      "yahoo": "crockford@yahoo.com",
      "aim": "thejsguy",
      "google": "crockford@gmail.com"
    },
    "contact": {
      "home": "123-456-7890",
      "work": "123-456-7890",
      "mobile": "123-456-7890",
      "address": "1234 Yahoo<br \\/>Sunnyvale, CA 12345 "
    }
  }, {
    "error": true
  }
}]
```

Method: contacts.remove

Removes the specified contact.

Request Params

"contactID"

- integer

Example Request

```
{
  "id": 12,
  "method": "contacts.remove",
  "params": {
    "contactID": 88
  }
}
```

Example Response

```
[{
  "result": true
}, {
  "error": true
}]
```

Method: display-options

Not implemented yet

Example Response

```
{
  "result": {
    "reading-pane": 2,
    "fields": ["from", "to", "subject", "date", "size"],
    "sort": {
      "field": "from",
      "order": 1
    }
  },
  "error": true
}
```

Method: folders.add

Adds a folder with given name and returns folderID.

Request Params

["parentID"] - *given if added folder is a child*

- integer

"name" - *name of the new folder*

- string

Example Request

```
{
  "id": 13,
  "method": "folders.add",
  "params": {
    "parentID": 9,
    "name": "Bitbucket"
  }
}
```

Response Params

"folderID" - *global folderID assigned by server*

- integer

Example Response

```
[{
  "result": {
    "folderID": 41
  }
}, {
  "error": true
}]
```

Method: folders.list

Lists the folders to be displayed in view menu.

Request Params

"context" - *lets server know which folders to list*

1. "mail" - *(def)*
2. "contacts"
3. "settings"
4. "help"

Example Request

```
{
  "id": 14,
  "method": "folders.list",
  "params": {
    "context": "mail"
  }
}
```

Response Params

"folderID"

- integer

["parentID"] - *only given if folder is a child*

- integer

"name"

- string

Example Response

```
[{
  "result": [{
    "folderID": 1,
    "name": "Inbox"
  }, {
    "folderID": 5,
    "name": "Projects"
  }, {
    "folderID": 8,
    "parentID": 12,
    "name": "jQuery Plugins"
  }
}]
```

```

    }, {
      "folderID": 10,
      "parentID": 12,
      "name": "Tests"
    }, {
      "folderID": 11,
      "parentID": 10,
      "name": "Webmail"
    }, {
      "folderID": 15,
      "parentID": 12,
      "name": "Templates"
    }, {
      "folderID": 12,
      "name": "Javascript"
    }, {
      "folderID": 9,
      "parentID": 8,
      "name": "Querios"
    }
  ]
}, {
  "result": [{
    "folderID": 200,
    "name": "All"
  }, {
    "folderID": 201,
    "name": "People"
  }, {
    "folderID": 202,
    "name": "Business"
  }, {
    "folderID": 203,
    "name": "Collected"
  }, {
    "folderID": 204,
    "name": "Work"
  }, {
    "folderID": 205,
    "name": "Personal"
  }, {
    "folderID": 206,
    "parentID": 205,
    "name": "Family"
  }, {
    "folderID": 207,
    "parentID": 205,
    "name": "Friends"
  }, {
    "folderID": 999,
    "name": "No Gravatars"
  }
  ]
}, {
  "result": [{
    "folderID": 100,
    "name": "Identity"
  }, {
    "folderID": 101,
    "name": "Mail Settings"
  }, {
    "folderID": 102,
    "name": "Portal Settings"
  }, {
    "folderID": 103,
    "name": "Account Upgrades"
  }, {
    "folderID": 104,
    "name": "Password"
  }
  ]
}, {
  "result": [{
    "folderID": 400,
    "name": "Statistics"
  }, {
    "folderID": 401,
    "name": "Security"
  }, {
    "folderID": 402,
    "name": "Contacts"
  }, {
    "folderID": 403,
    "name": "Mail"
  }
  ]
}, {
  "result": [{
    "folderID": 500,
    "name": "Help Category"
  }, {
    "folderID": 501,
    "name": "Help Category"
  }, {
    "folderID": 502,
    "name": "Help Category"
  }
  ]
}, {

```

```
    "error": true
  }]
```

Method: folders.remove

Removes the folder.

Request Params

"folderID"

- integer

Example Request

```
{
  "id": 16,
  "method": "folders.remove",
  "params": {
    "folderID": 30
  }
}
```

Example Response

```
[{
  "result": true
}, {
  "error": true
}]
```

Method: folders.rename

Rename the specified folder.

Request Params

"folderID"

- integer

"name"

- string

Example Request

```
{
  "id": 17,
  "method": "folders.rename",
  "params": {
    "folderID": 30,
    "name": "New Name"
  }
}
```

Example Response

```
[{
  "result": true
}, {
  "error": true
}]
```

Method: help.page

Get html for wiki style help pages.

Request Params

"topicID"

- integer

Example Request

Response Params

- html string

Example Response

```
[{
  "jsonrpc": "2.0",
  "result": "<h2>Help</h2><p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque ipsum orci, dictum quis vestibulum
  "id": 100
}, {
  "error": true
}]
```

Method: help.topics

List help topics for a category.

Request Params

"categoryID"

- integer

Example Request

Response Params

"topicID"

- topicID

"name" - *topic display name*

- string

Example Response

```
[{
  "jsonrpc": "2.0",
  "result": [{
    "topicID": 600,
    "name": "Help Topic"
  }, {
    "topicID": 601,
    "name": "Help Topic"
  }, {
    "topicID": 602,
    "name": "Help Topic"
  }],
  "id": 100
}, {
  "error": true
}]
```

Method: logs.mail

Retrieve mail logs

Response Params

"name" - *full name*

- string

"first" - *first name*

- string

"last" - *last name*

- string

"website"

- string

Example Response

```
[{
  "result": [{
    "messageID": 100,
    "queue": 1011,

```

```
    "type": "Incoming",
    "from": "Bill Shakespeare",
    "to": "Bill Gates",
    "outcome": "Delivered",
    "utc": 102032012310,
    "bytes": 1024
  }, {
    "messageID": 101,
    "queue": 1012,
    "type": "Outgoing",
    "from": "Bob Dole",
    "to": "Bob Dylan",
    "outcome": "Delayed",
    "utc": 102033422369,
    "bytes": 1232
  }, {
    "messageID": 102,
    "queue": 1013,
    "type": "Incoming",
    "from": "Bill Shakespeare",
    "to": "Bill Gates",
    "outcome": "Delivered",
    "utc": 102032012310,
    "bytes": 123145
  }, {
    "messageID": 103,
    "queue": 1014,
    "type": "Outgoing",
    "from": "Bob Dole",
    "to": "Bob Dylan",
    "outcome": "Delayed",
    "utc": 102033422369,
    "bytes": 10
  }, {
    "messageID": 104,
    "queue": 1015,
    "type": "Incoming",
    "from": "Bill Shakespeare",
    "to": "Bill Gates",
    "outcome": "Delivered",
    "utc": 102032012310,
    "bytes": 999
  }, {
    "messageID": 105,
    "queue": 1016,
    "type": "Outgoing",
    "from": "Bob Dole",
    "to": "Bob Dylan",
    "outcome": "Delayed",
    "utc": 102033422369,
    "bytes": 123
  }
}, {
  "error": true
}]
```

Method: logs.security

Retrieve security logs

Response Params

"utc" - *unix time*

- string

"type" - *login / logout*

- string

"severity"

- string

"ip"

- string

"protocol"

- string

Example Response

```
[{
  "result": [{
    "utc": 10239834048,
    "type": "Login",
    "severity": "Zero",
```

```
        "ip": "0.0.0.0",
        "protocol": "Web"
    }, {
        "utc": 10238439074,
        "type": "Logout",
        "severity": "Zero",
        "ip": "0.0.0.0",
        "protocol": "Web"
    }, {
        "utc": 10239834157,
        "type": "Login",
        "severity": "Zero",
        "ip": "0.0.0.0",
        "protocol": "IMAP"
    }
  ], {
    "error": true
  }
}
```

Method: logs.statistics

Gets account statistics.

Response Params

"account"

"username"

- string

"name"

- string

"number"

- integer

"reputation"

- string

"plan"

- string

"date"

- string

"storage"

"space"

- integer

"folders"

- integer

"stored"

- integer

"archived"

- integer

"encrypted"

- integer

"logins"

"smtp" - *number of times accessed*

- integer

"pop" - *number of times accessed*

- integer

"imap" - *number of times accessed*

- integer

"web" - *number of times accessed*

- integer

"messages"

"received" - *total received*

- integer

"sent" - *total sent*

- integer

"email"

"address"

- string

"transfer"

"sent"

- integer

"received"

- integer

"blocked"

"spam"

- integer

"bounced"

- integer

Example Response

```
[{
  "result": {
    "account": {
      "username": "Mike",
      "name": "Mike",
      "number": 1,
      "reputation": "100%",
      "plan": "Free",
      "date": "Nov 11th, 2011"
    },
    "storage": {
      "space": 10203192038,
      "folders": 12,
      "stored": 1000303,
      "archived": 12030123213,
      "encrypted": 1200310320
    },
    "logins": {
      "smtp": 1000,
      "pop": 1000,
      "imap": 1000,
      "web": 1000
    },
    "messages": {
      "received": 1000,
      "sent": 1000
    },
    "email": {
      "address": "mike@lavabit.com"
    },
    "transfer": {
      "sent": 2,
      "received": 3
    },
    "blocked": {
      "spam": 0,
      "bounced": 0
    }
  }, {
    "error": true
  }
}]
```

Method: messages.compose

Sent when the user starts to compose a message. Gets the composeID from the server.

Response Params

"contactID"

- integer

Example Response

```
[{
  "result": { "composeID": 1 }
}, {
  "error": true
}]
```

Method: messages.copy

Copy messages from one folder to another or the same folder.

Request Params

"messageIDs"

- array of ints

"sourceFolderID"

- int

"targetFolderID"

- int

Example Request

```
{
  "id": 19,
  "method": "messages.copy",
  "params": {
    "messageIDs": [100, 101, 102],
    "sourceFolderID": 1,
    "targetFolderID": 10
  }
}
```

Response Params

Each object in the response array is associated with a messageID in the request.

"sourceMessageID"

- integer

"targetMessageID"

- integer

Example Response

```
[{
  "result": [{
    "sourceMessageID": 1021,
    "targetMessageID": 1035
  }, {
    "sourceMessageID": 1022,
    "targetMessageID": 1036
  }]
}, {
  "error": true
}]
```

Method: messages.flag

Assign various flags to a message. The flag property indicates what flag is being assigned.

Request Params

Each object in the example is a separate request.

"action"

- string
 1. add
 2. replace
 3. remove

"flags"

- array of strings
 - read
 - unread
 - star
 - unstar

"messageIDs"

- array

"folderID"

- integer

Example Request

```
[{
  "id": 20,
  "method": "messages.flag",
  "params": {
    "action": "add",
    "flags": ["junk"],
    "messageIDs": [100, 101, 102],
    "folderID": 1
  }
}, {
  "id": 20,
  "method": "messages.flag",
  "params": {
    "action": "replace",
    "flags": [],
    "messageIDs": [100, 101, 102],
    "folderID": 1
  }
}, {
  "id": 20,
  "method": "messages.flag",
  "params": {
    "action": "remove",
    "flags": ["read"],
    "messageIDs": [100, 101, 102],
    "folderID": 1
  }
}, {
  "id": 20,
  "method": "messages.flag",
  "params": {
    "action": "list",
    "messageIDs": [100, 101, 102],
    "folderID": 1
  }
}]
```

Example Response

```
[{
  "jsonrpc": "2.0",
  "result": {
    "messages.flag": "success"
  },
  "id": 20
}, {
  "jsonrpc": "2.0",
  "result": {
    "messageID": 1,
    "flags": ["flagged"]
  },
  "id": 20
}]
```

Method: messages.list

List messages of the given folder.

Request Params

"folderID"

- integer

Example Request

```
{
  "id": 21,
  "method": "messages.list",
  "params": {
```

```
    "folderID": 1
  }
}
```

Response Params

"messageID"

- integer

["attachment"]

- boolean

["flags"]

- array of strings

"from"

- string

"to"

- string

"addressedTo"

- string

"replyTo"

- string

"returnPath"

- string

["carbon"]

- string

"subject"

- string

"utc" - *server utc time in seconds (unix time)*

- integer

"arrivalUtc"

- integer

"snippet"

- string

"bytes" - *size in bytes*

- integer

["tags"]

- array of strings

Example Response

```
[{
  "result": []
}, {
  "result": [{
    "messageID": 1024,
    "flags": ["seen", "flagged"],
    "from": "William Adama",
    "to": "Apallo",
    "addressedTo": "apallo@lavabit.com",
    "replyTo": "bill@lavabit.com",
    "returnPath": "bill@lavabit.com",
    "subject": "Pegasus",
    "utc": 1109749011,
    "arrivalUtc": 1109749011,
    "snippet": "Go the the gym softy",
    "bytes": 487479772
  }, {
    "messageID": 1023,
    "attachment": true,
    "from": "Douglas Crockford",
    "to": "John Resig",
    "addressedTo": "jresig@lavabit.com",
    "replyTo": "dcrock@lavabit.com",
    "returnPath": "dcrock@lavabit.com",
    "carbon": "mozilla@lavabit.com",
    "subject": "jQuery cruft",
    "utc": 1266415038183,
```

```

    "arrivalUtc": 1020030,
    "snippet": "jQuery is too crufty. YUI is much leaner.",
    "bytes": 895625721,
    "tags": ["Javascript"]
  }, {
    "messageID": 1022,
    "from": "Paul Grahm",
    "to": "Douglas Crockford",
    "addressedTo": "dcrock@lavabit.com",
    "replyTo": "paulg@lavabit.com",
    "returnPath": "paulg@lavabit.com",
    "subject": "Reply: Y Combinator cruft",
    "utc": 1071564744567,
    "arrivalUtc": 1023012302,
    "snippet": "I have a startup idea. It's a startup starter...",
    "bytes": 1014127863,
    "tags": ["Yahoo", "Y Combinator", "cruft"]
  }, {
    "messageID": 1021,
    "from": "Douglas Crockford",
    "to": "David Flanagan",
    "addressedTo": "dflan@lavabit.com",
    "replyTo": "dcrock@lavabit.com",
    "returnPath": "dcrock@lavabit.com",
    "subject": "Javascript: The Definitive Guide cruft",
    "utc": 1054397333677,
    "arrivalUtc": 10023100321,
    "snippet": "You should only talk about the good parts.",
    "bytes": 839535031,
    "tags": ["Javascript"]
  }
], {
  "error": true
}]

```

Method: messages.load

Loads message details to be viewed in reading screen or info accordian. Sections of the message can be requested in batches with an array.

Request Params

"messageID"

- integer

"section" - *multiple can be requested*

- array of strings
 1. "header"
 2. "body"
 3. "attachments"
 4. "info"

Example Request

```

{
  "id": 22,
  "method": "messages.load",
  "params": {
    "messageID": 1,
    "folderID": 1,
    "sections": ["header", "body", "attachments"]
  }
}

```

Response Params

["header"] - *header summary*

- "from"
 - string
- "subject"
 - string
- "utc" - *email client utc*
 - integer
- "to"
 - string
- ["cc"]
 - string
- ["bcc"]
 - string
- ["sender"]

- string
- "replyto"
 - string

["body"]

- ["html"] - *if message is to displayed as html*
 - string
- ["text"] - *if message is ascii text*
 - string

["attachments"]

- "attachmentID"
 - integer
- "name"
 - string
- "bytes" - *size in bytes*
 - integer

["info"] - *extra message information*

- "source"
 - "ip"
 - string
 - "dns"
 - string
 - "reputation"
 - string
- "security"
 - "secure" - *displays red or green status indicator*
 - bool
 - "spf" - *displays red or green status indicator*
 - bool
 - "dkim" - *displays red or green status indicator*
 - bool
- "server"
 - "utc" - *server time*
 - integer
 - "images" - *displays red or green status indicator*
 - bool
 - "warnings"
 - bool

Example Response

```
[
  {
    "result": {
      "meta": {
        "messageID": 1,
        "folderID": 1,
        "flags": [

        ],
        "tags": [
          "thefuture"
        ]
      },
      "header": {
        "sender": "forwarder@example.org",
        "from": "John Resig",
        "to": "Mozilla",
        "cc": "carbon@example.org",
        "bcc": "blind@example.org",
        "returnpath": "bounces@example.org",
        "subject": "jQuery",
        "date": 1003939300349,
        "replyto": "John Resig",
        "size": 87348
      },
      "body": {
        "html": "<html>Hi There!</html><p>jQuery 1.5 just came out. You should use it!</p><p>John Resig</p>"
      },
      "attachments": [
        {
          "attachmentID": 59,
          "name": "Photo1.jpg",
          "size": 11039,
```

```
{
  "type": "image/jpeg"
},
{
  "attachmentID": 60,
  "name": "Photo2.jpg",
  "size": 12093,
  "type": "image/jpeg"
},
{
  "attachmentID": 60,
  "name": "profits.xlsx",
  "size": 59120,
  "type": "application/octet-stream"
}
]
}, {
  "result": {
    "info": {
      "source": {
        "ip": "1.23.45.678",
        "dns": "dns1.email.com",
        "reputation": "Good"
      },
      "security": {
        "secure": true,
        "spf": true,
        "dkim": false
      },
      "server": {
        "utc": 1003029300349,
        "images": true,
        "warnings": "The message is spoofed! The signature it carries did not validate! "
      }
    }
  }
}
}]
```

Method: messages.move

Move messages from one folder to another. Cannot be the same folder.

Request Params

"messageIDs"

- array of ints

"folderID"

- int

Example Request

```
{
  "id": 23,
  "method": "messages.move",
  "params": {
    "messageIDs": [100, 101, 102],
    "sourceFolderID": 1,
    "targetFolderID": 10
  }
}
```

Example Response

```
[{
  "result": true
}, {
  "error": true
}]
```

Method: messages.remove

Move messages from one folder to another. Cannot be the same folder.

Request Params

"messageIDs"

- array of ints

Example Request

```
{
  "id": 24,
  "method": "messages.remove",
  "params": {
    "folderID": 1,
    "messageIDs": [100, 110, 120]
  }
}
```

Example Response

```
[{
  "result": true
}, {
  "error": true
}]
```

Method: messages.send

Sends serialized message data to server

Request Params

"composeID"

- integer

"from"

- string

"to"

- string

["subject"] - *not sent if empty - user warned before sent*

- string

["priority"]

1. "low"
2. "normal" - (*def*)
3. "high"

["body"] - *not sent if empty - user warned before sent*

1. "html"
 - string
2. "text"
 - string

Example Request

```
{
  "id": 25,
  "method": "messages.send",
  "params": {
    "composeID": 0,
    "from": "Douglas Crockford <dc@yahoo.com>",
    "to": "Linus Torvalds <lt@kernel.org>",
    "subject": "Kernel cruft",
    "priority": "normal",
    "attachments": [0, 1, 3],
    "body": {
      "html": "<h1>Linux Cruft</h1><p>Split Linux into specialized kernels to clean up some cruft!</p><p>DC</p>"
    }
  }
}
```

Example Response

```
[{
  "result": true
}, {
  "error": true
}]
```

Method: messages.tag

Assign tags to messages.

Request Params

"messageIDs"

- array of ints

"tags"

- array of strings

Example Request

```
[{
  "id": 20,
  "method": "messages.tag",
  "params": {
    "action": "add",
    "tags": ["javascript", "magma"],
    "messageIDs": [100, 101, 102],
    "folderID": 1
  }
}, {
  "id": 20,
  "method": "messages.tag",
  "params": {
    "action": "replace",
    "tags": [],
    "messageIDs": [100, 101, 102],
    "folderID": 1
  }
}, {
  "id": 20,
  "method": "messages.tag",
  "params": {
    "action": "remove",
    "tags": ["almost", "done"],
    "messageIDs": [100, 101, 102],
    "folderID": 1
  }
}, {
  "id": 20,
  "method": "messages.tag",
  "params": {
    "action": "list",
    "messageIDs": [100, 101, 102],
    "folderID": 1
  }
}]
```

Example Response

```
[{
  "result": true
}, {
  "error": true
}]
```

Method: messages.tags

Grabs tags from the server

Response Params

Success just sends array of tags.

- array of tags

Example Response

```
[{
  "result": ["Important", "Work", "Personal", "To Do"]
}, {
  "error": true
}]
```

Method: meta

Displays meta info in the main page. Browser, cookies, stylesheets, and connection security are detected by client.

Response Params

"userID"

- integer

"clientIP"

- string

"location"

- string

"reputation" - 0 to 100

- integer

"plan"

1. "Free"
2. "Basic"
3. "Personal"
4. "Enhanced"
5. "Premium"

"version" - *webmail version*

- string

"quota" - 0 to 100

- integer

Example Response

```
[{
  "result": {
    "userID": 32,
    "clientIP": "127.0.0.1",
    "location": "Dallas, TX",
    "timezone": "Central",
    "reputation": 88,
    "plan": "Basic",
    "version": "1.02",
    "quota": 45
  }, {
    "error": true
  }
}]
```

Method: scrape.add

Add a scraped contact from the scraped contacts page. Any optional request params not present were empty submissions.

Request Params

"messageID"

- integer

"id" - *temp id assigned by server from contacts.scrape method*

- integer

"name"

- string

"email"

- string

["phone"]

- string

["work"]

- string

["cell"]

- string

Example Request

```
{
  "id": 27,
  "method": "scrape.add",
  "params": {
    "messageID": 203,
```

```
      "id": 59,
      "name": "John Doe",
      "email": "jdoe@example.com"
    }
  }
}
```

Example Response

```
[{
  "result": true
}, {
  "error": true
}]
```

Method: scrape

Grab contacts from a message to display in the scraping screen.

Request Params

"messageID" - *refers to message that contacts should be scraped from*

- integer

Example Request

```
{
  "id": 28,
  "method": "scrape",
  "params": {
    "messageID": 203
  }
}
```

Response Params

"id" - *temporary id assigned by server*

- integer

"name"

- string

"email"

- string

["phone"]

- string

["work"]

- string

["cell"]

- string

Example Response

```
[{
  "result": [{
    "contactID": 45,
    "name": "John Resig",
    "email": "jrmozilla.org",
  }, {
    "contactID": 48,
    "name": "Linus Torvalds",
    "email": "lt@kernel.org",
  }, {
    "contactID": 53,
    "name": "Douglas Crockford",
    "email": "dc@yahoo.com",
  }, {
    "contactID": 58,
    "name": "Paul Graham",
    "email": "pg@ycombinator.com"
  }, {
    "contactID": 64,
    "name": "John Paul",
    "email": "jp@morgan.com"
  }
]}, {
  "error": true
}]
```

Method: search

Sent when searching with advanced search.

Request Params

"searchin"

- "folderID" - *will need some way to specify all*
 - integer

"queries" - *array of objects containing the following params*

- "field"
 1. "from"
 2. "to"
 3. "subject"
 4. "date" - *changes filter to date range*
 5. "size" - *changes filter to greater than, less than*
- ["filter"]
 1. "contains"
 2. "does not contain"
 3. "greater" - *for searching size*
 4. "less" - *for searching size*
- ["query"] - *sent only for filtered items*
 1. string
 2. integer - *size in bytes when searching size*
- ["range"]
 - "from"
 - unix time
 - "to"
 - unix time

Example Request

```
{
  "id": 29,
  "method": "search",
  "params": {
    "searchin": 0,
    "queries": [{
      "field": "from",
      "filter": "contains",
      "query": "Paul Graham"
    }, {
      "field": "date",
      "range": {
        "from": 23112342342,
        "to": 2342342343
      }
    }
  ]
}
```

Example Response

```
[{
  "result": [{
    "messageID": 1024,
    "flags": ["seen", "flagged"],
    "from": "William Adama",
    "to": "Apollo",
    "addressedTo": "apollo@lavabit.com",
    "replyTo": "bill@lavabit.com",
    "returnPath": "bill@lavabit.com",
    "subject": "Pegasus",
    "utc": 1109749011,
    "arrivalUtc": 1109749011,
    "snippet": "Go the the gym softy",
    "bytes": 487479772
  }, {
    "messageID": 1023,
    "attachment": true,
    "from": "Douglas Crockford",
    "to": "John Resig",
    "addressedTo": "jresig@lavabit.com",
    "replyTo": "dcrock@lavabit.com",
    "returnPath": "dcrock@lavabit.com",
    "carbon": "mozilla@lavabit.com",
    "subject": "jQuery cruft",
    "utc": 1266415038183,
    "arrivalUtc": 1020030,
```

```
    "snippet": "jQuery is too crufty. YUI is much leaner.",
    "bytes": 895625721,
    "tags": ["Javascript"]
  }
}, {
  "error": true
}]
```

Method: settings.identity

Retrieve identity settings

Response Params

Returns settings that have been set.

"name" - *full name*

- string

"first" - *first name*

- string

"last" - *last name*

- string

"website"

- string

Example Response

```
[{
  "result": {
    "name": "Lava Bit",
    "first": "Lava",
    "last": "Bit",
    "website": "lavabit.com"
  }
}, {
  "error": true
}]
```