API Reference

This document is a reference for Comapping API. Please refer to the information below to interact with the Comapping service programmatically.

This document consists of the following parts:

- *Comapping XML format* defines the data format used to keep Comapping data in the file system
- *Comapping Server API* defines the ways to make Comapping server perform certain operations
- *Comapping Client API* defines the ways to make Comapping client pull data from certain external data source

Comapping XML format

Comapping XML format is used to keep Comapping maps in the file system. It can be obtained using a variety of ways:

- using Comapping Export functionality
- using Comapping AIR version Save As Desktop functionality

This section defines the structure of Comapping XML format. The document is wrapped into the root `<mindmap>` node:

```
<mindmap>
   (contents here)
</mindmap>
```

The root code contains the following subnodes:

- `<metadata>`
- `<presentation>`
- data section

Metadata

This subnode is used to keep general information about the map. A typical structure is presented below:

```
<metadata>
  <id><![CDATA[15411]]></id>
  <name><![CDATA[Example map]]></name>
  <owner>
    <id><![CDATA[49]]></id>
    <name><![CDATA[Mike Pliskin]]></name>
    <email><![CDATA[mike@comapping.com]]></email>
  </owner>
  <access><![CDATA[Shared]]></access>
  <source><![CDATA[Server]]></source>
  <collaborators><![CDATA[info@comapping.com]]></collaborators>
  <watchers><![CDATA[]]></watchers>
  <isTemplate><![CDATA[false]]></isTemplate>
  <params/>
</metadata>
```

- `id` is map ID in the internal Comapping database
- `name` is map name
- `owner` is map owner with its own ID, name and email
- `access` defines the access level map has for the user that exported the map
- `source` defines the source of map. Possible values include `Server` and `Desktop`
- `collaborators` and `watchers` are comma-separated lists of map collaborators and watchers respectively
- `isTemplate` is a boolean defining whether map is a template or not

Note most of these parameters are only informational and represent map status in the Comapping service. They are mostly ignored when importing the map back.

Presentation

This section keeps the presentation a map might have. Presentation is a sequence of slides, while each slide is focus topic and a sequence of manually collapsed nodes. An self-descriptive example of this structure is presented below:

```
<presentation>
  <slide>
    <focus><![CDATA[1058528]]></focus>
    <collapseStatus>
      <topic><![CDATA[1058529]]></topic>
      <topic><![CDATA[1058574]]></topic>
    </collapseStatus>
  </slide>
</presentation>
```

Data section

This section defines the most important aspect of the map: its tree of topics. Each topic is represented by a <node> tag with nested tags used to represent subtopics. A typical <node> structure is represented below:

```
<node id="1058529" priority="1">
  <note><![CDATA[This is a note]]></note>
  <text><![CDATA[Use this comap to learn the basics]]></text>
  <node arrow="blue" flag="caution" id="1058530" smiley="happy"
star="yellow" taskCompletion="50">
    <icon name="clock"/>
    <task deadline="tomorrow" estimate="1 day" responsible="You"
start="today"/>
    <text><![CDATA[Don&apos;t worry - you can&apos;t break anything no
matter how hard you try!]]></text>
  </node>
</node>
```

You can see a topic with one subtopic there. Notice a number of attributes and nested tags – they are used to represent various aspects of a topic. Please find the complete reference below.

Topic attributes

Name	Values	Comments
id	Positive integer Arbitrary string	Topic ID, integer or non-integer depending on whether it is client or server
priority	0-9	Specifies the priority icon
taskCompletion	0,25,50,75,100,-1	Specifies the task completion icon
smiley	happy neutral neutral sad furious	Specifies the smiley icon
flag	go for_discussion possibility risk progress careful caution	Specifies the flag icon
star	red orange	Specifies the star icon

	yellow green blue purple black	
arrow	red orange yellow green blue purple black	Specifies the arrow icon
bgColor	Hex value	Background color for the topic
externalId	ID of external data item	See External data explained document
locked	true	If present, indicates that the topic is locked

Nested tags

Name	Occurrence	Contents	Comments
text	Once	CDATA	Topic text with CDATA
node	Any		Subtopics of the topic
task	Once or zero	See Task description below	Task of a topic
icon	Any	Name of an icon as an attribute	See below for the list of valid names
Attachment	Once or zero	See Attachment description below	File attachment of a topic

Task

The task tag describes the task for a topic. It has the following attributes:

Name	Values	Comments
responsible	String	Task responsible
deadline	String	Task deadline
start	String	Task start date
estimate	String	Task estimate

Although `deadline` and `start` fields are strings and can contain any data (like '`tomorrow`' or '`next week`'), specifying dates in `YYYY-MM-DD` format (like `2009-09-05`) will make the calendar pop-up display this date when editing a task.

Attachment

The attachment tag describes the file attachment for a topic. It has the following attributes:

Name	Values	Comments
key	String	Unique attachment ID
filename	String	Filename of the attachment
size	Integer	Attachment size in bytes

Icon

The icon tag describes possible individual icons for a topic. The following icons are available:

- `question_mark`
- `exclamation_mark`

- bomb
- thumbs_up
- thumbs_down
- magnifier
- tea_time
- puzzle
- dollar
- heart
- clock
- idea
- lock
- test
- homework
- needs_feedback
- reminder
- needs_chat

Comapping Server API

Comapping Server API is HTTP-based and designed in a RESTful manner. All operations are performed as HTTP GET or POST requests with GET ones used whenever possible.

All API requests are to be sent to a single API handler URL:

`http://go.comapping.com/cgi-bin/comapping.n`

The following parameters are required:

- `action` - defines the API action to be performed

The API calls respond either with action-specific result indicating success, or with a standard XML indicating error. This XML looks like the following:

```
<error/>
```

with some details inside the error element.

API actions are the highest-level concepts in API. They define the top-level operations you can perform with a Comapping service. The following actions are available now.

Login

This action performs user authentication. The login procedure is as follows:

1. Client sends a request with the following parameters:

Name	Value
`action`	`login`
`login`	e-mail
`loginMethod`	`simple`

2. If user exists, server responds with the XML like the following:

   ```
   <success salt="some_random_value_here"/>
   ```

   If there is no such a user or other error occurs, server responds with the standard error XML.

3. Client computes the password hash by appending the salt from server response to password and calculating an MD5 hash code of the combined string.
4. Client sends a GET request with the following parameters

Name	Value
`action`	`login`
`login`	e-mail
`password`	hash computed on step 3
`loginMethod`	`withSalt`

5. If everything is correct, server responds with the following XML:

```
<success clientID="ABHWESADH200908151156"/>
```

In case of a problem, the standard error XML is returned.

Client has to remember the `clientID` it was given and use it in all subsequent interaction with the service.

Download

The download action is used to download a map specified in one of possible export formats. The client sends a GET request with the following parameters:

Name	Value
`action`	`download`
`clientID`	client ID
`format`	One of possible export formats, see below
`mapid`	numeric map ID `meta` to download a map of maps for the user

The following export formats are available:

`comap`	Comapping own XML-based format
`mindmanager`	Mindjet MindManager file
`freemind`	Freemind file
`html_bullets`	HTML file with map as multi-level bullet list
`html_layout`	HTML file with map laid out close to Comapping application
`rtf`	Rich Text Format (RTF) file, Microsoft Word compatible
`mpx`	MPX file for Microsoft Project
`opml`	OPML file
`csv_semicolon` **`csv_comma`** **`csv_tab`** **`csv_space`**	CSV (Comma Separated File) with the separator specified

The server responds with the map exported in the format specified or with the standard error XML if an error occurs.

Upload

The upload action is used to upload/import data into Comapping. To do that, you need to upload the file to the URL with the following GET parameters:

Name	Value
`action`	`Upload`
`clientID`	client ID

The server will automatically try to recognize the format of the file being uploaded by its extension. The following formats are supported:

Extension	Format
`.comap`	Comapping own XML-based format
`.mmap`	Mindjet MindManager file
`.mm`	Freemind file
`.opml`	OPML file

The server will reply with either the following XML indicating success and the id of the uploaded map:

```
<success mapid="12345"/>
```

or with a standard error XML in case of error.

Broadcast Commands

This action allows you to change a map. All your changes are broadcasted to the online users editing the same map live if any. The server sends a POST request with the following query parameters:

Name	Value
action	broadcastCommands
clientID	client ID

and the POST body with the XML specified below:

```
<commands mapid="1214">
  (commands here)
</commands>
```

The content of commands tag is a sequence of commands. Each command is represented by an XML structure depending on its type (see below). Server response with a standard error package if an error occurs or with the following XML for success:

```
<success>
  <idmap client="myid" server="12345" />
  ...
</success>
```

The <idmap> entries represent a topic ID translation table. When you add a topic, server assigns a unique ID to it, and this table is a way for you to know this ID for further topic operations. If you specified some other ID for a new topic, this ID will be listed as "client" allowing you to add multiple topics in one go and still get the proper IDs back.

Change Topic command

This command changes a topic contents replacing it with a new one. Contents including everything in topic except children and parents, i.e. text, icons, tasks, notes etc. The XML structure is as follows:

```
<changeTopic id="12346">
  (new node spec here)
</changeTopic>
```

Add Topic command

This command adds a child to the existing topic. The XML structure is as follows:

```
<addTopic parentId="12345">
   (new topic spec here)
</changeTopic>
```

Remove Topic command

This command removes a topic. The XML structure is as follows:

```
<removeTopic childid="12345" />
```

Move Topic command

This command moves topic from one position to another. The XML structure is as follows:

```
<moveTopic id="12345" newparentid="45678" index="5" />
```

Position to insert is specified using the `index` attribute.

Change Permissions command

This command allows you to change the map permissions. The XML structure is as follows:

```
<changePermissions message="Message to be sent in the sharing email">
    <user name="New User" email="newuser@comapping.com" access="edit" />
    <user name="Too Active User" email="activeuser@comapping.com"
access="view" />
    <user name="To be removed" email="toberemoved@comapping.com"
access="none" />
</changePermissions>
```

The `access` parameters in `<user>` tag can be one of: `edit`, `view`, `none`. Specifying `none` removes user from the sharing list. Omitting a user leaves her/his permissions for this map unchanged.

Comapping Client API

Comapping Client API allows using Comapping client to synchronize data from an external data source. The algorithm is as follows:

- A map can be associated with an external URL
- Comapping client can the do one of the following:
  - `Get Data` from the external service and update the map
  - `Send Data` to the external service and make the service aware of the changes in the map
- The way map is updated links individual topics in the map with the data items in the external service
- The protocol is described below

Configuring the map

To associate a map with an external service, do the following:

1. Open a map in Comapping
2. Click Advanced tab
3. Click Settings button in the External Data group
4. Edit the Service URL and the secret key

Synchronizing data

To synchronize your map with an online service, do the following:

1. Open a map in Comapping
2. Click Advanced tab
3. Click Get Data button in the External Data group to get data
4. Click Send Data button in the External Data group to send data

Get Data

To pull data from an external service, Comapping makes a GET request with the following parameters to the service URL:

Name	Value
command	`list`
time	Current time in seconds
sign	request signature, see below

The service should respond with the XML like the following:

```
<response>
    <success>
        <nodes>
            <node id="86" priority="1">
                <text><![CDATA[Main course]]></text>
            </node>
            <node id="87" priority="2">
                <text><![CDATA[Dessert]]></text>
            </node>
            <node id="99" priority="3">
                <text><![CDATA[<a
        href="http://www.google.com">Google</a> for drinks]]></text>
```

```
                </node>
                <node id="100" priority="4">
                      <text><![CDATA[Pay the bill]]></text>
                </node>
          </nodes>
     </success>
</response>
```

The <nodes> collection contains a sequence of <node> items. Each <node> element is similar to the <node> element in *Comapping XML format* section above but with the following notes:

- `id` attribute means External ID (i.e. ID in the external database the service is for), it will be kept in corresponding Comapping topics as an `externalId` attribute
- only `priority` attribute is supported at the moment
- only `<text>` nested element is supported at the moment

Having got the service response, Comapping client tries to update the map with the data received. To do that, the following algorithm is applied:

- For each received <node>
  - try to find a corresponding topic in the map by looking
    - at the `externalId` attribute at first
    - at the topic text then
  - if found, update the topic found with the text and priority
  - if not found, create new topic with the text and priority received as a new child of root
- For each topic with `externalId` attribute in a map
  - If there is no corresponding <node> in the data received, grey the topic out

Send Data

When sending data to the external service, a POST request is made to the external service URL with the following extra GET parameters:

Name	Value
time	Current time in seconds
sign	request signature, see below

The POST body is an XML like the following:

```
<request>
     <command name=”send”>
          <nodes>
                <node id="86" priority="1">
                      <text><![CDATA[Main course]]></text>
                </node>
                <node id="87" priority="2">
                      <text><![CDATA[Dessert]]></text>
                </node>
                <node id="99" priority="3">
                      <text><![CDATA[<a
          href="http://www.google.com">Google</a> for drinks]]></text>
                </node>
                <node id="100" priority="4">
```

```
                    <text><![CDATA[Pay the bill]]></text>
                </node>
            </nodes>
        </command>
    </request>
```

The <nodes> collection will have an item for each node with a non-empty `externalId` attribute.

Calculating signature

All requests will be done with a `sign` parameter containing request signature. The signature is an MD5 hash computed of the following string:

```
Command name (either from GET parameter or from <command name="…"> tag concatenated with time (contents of GET parameter) concatenated with the service secret key.
```