tinyMonitor API Specification

Version 0.1

Table of Contents

General Information

The tinyMonitor API Specification defines methods and data formats to send and receive monitoring information about a software or a hardware source.

The current API version number is ***1***.

Technology

The API is based on ***HTTP*** requests and ***JSON*** for data exchange.

Requesting Information from a Monitor Source

To request information from a source, the client must send a GET request to the source URL with no parameters at all.

E.g.: ***GET*** *http://www.myserver.com/mymonitor*

Upon the request, the monitor may send its data.

Sending Data to a Client

Upon the request of a client, the monitor may send its data as a JSON object string, following the rules below:

- the JSON object must have the following properties: ***title, version, items, actions***;

- the ***title*** property defines the name of the monitor to be displayed on the client;

- the ***version*** property defines the version number of the API being used to send the data;

- the ***items*** property is an array of items to be displayed on the client (see ***Monitor Items***);

- the ***actions*** property is an array of actions that can be performed on the monitor (see ***Monitor Actions***);

- the monitor may send empty arrays for ***items*** and ***actions*** when no information is provided and/or no actions can be executed on the monitor;

E.g.:

```
{
  "title": “My Monitor”,
  “version”: 1,
  “items”: [...],
  “actions”: [...]
}
```

Monitor Items

The monitor items define the information that must be shown on the requesting client (how this information is shown is up to the client).

Each item on the items array must be an object, have a ***type*** property and, depending on its type, a number of other properties to be considered a valid item.

The ***type*** property must be an integer, identifying the type of the item that must be displayed. The table below shows the supported types of items and its required properties.

Type	Name / Description	Required Properties
1	***Message*** A simple message.	***message:*** string - the message;
2	***Text*** A text value.	***title:*** string - the item title; ***text:*** string - the text value;
3	***Status*** A status item.	***title:*** string - the item title; ***text:*** string - the status text; ***status:*** integer - the status code, as defined bellow: - 0: ***unknown***; - 1: ***OK***; - 2: ***warning***; - 3: ***error***;
4	***Progress / Usage*** A progress/usage bar.	***title:*** string - the item title; ***text:*** string - the progress/usage information text; ***percentage:*** float - the progress/usage percentage;

E.g.:

```
{
  [...]
  "items": [
        "0": { "type": 1, "message": "This is my awesome server!" },
        "1": { "type": 2, "title": "Server Version", "text": "0.awesome!" },
        "2": { "type": 3, "title": "Server Status", "text": "All good!", "status": 1 },
        "3": { "type": 4, "title": "Disk Usage", "text": "Looooooow!", "percentage": 5 },
  ],
  [...]
}
```

Monitor Actions

The monitor actions define what actions can be performed on the monitor.

Each item on the actions array must be an object and have a ***name*** and an ***action*** property, where the ***name*** is the one that's displayed on the client and the ***action*** is the identifier to be sent to the monitor.

E.g.:

```
{
  [...]
  “actions”: [
        "0": { "name": "Reboot!", "action": "reboot" }
        "1": { "name": "Shutdown!", "action": "shutdown" }
  ]
}
```

To perform actions on the monitor, the client must send a GET request to the monitor URL including the command as the name of the first URL parameter:

E.g.: *GET http://www.myserver.com/mymonitor/**?shutdown***

Action Response

When an action is requested, the monitor must respond normally with at least a message item, telling the user what's happening.

E.g.:

Action Request: GET http://www.myserver.com/mymonitor/?shutdown

Response:

```
{
  "title": “My Monitor”,
  “version”: 1,
  “items”: [
        "0": { "type": 1, "message": "The server is shutting down, please wait." },
  ],
  “actions”: [],
}
```