

Monitor

API Reference

© 2017 All rights reserved by Metrological

This document contains information which is proprietary and confidential to Metrological. It is provided with the expressed understanding that the recipient will not divulge its content to other parties or otherwise misappropriate the information contained herein. This information is furnished for guidance; specifications and availability of goods mentioned in it are subject to change without notice. No part of this publication may be reproduced, stored in a database, retrieval system or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the written prior permission of Metrological.

History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| 0.1 | 07-01-2017 | P. Wielders | Initial version |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Scope 4

1.2 Case sensitivity 4

1.3 Acronyms, Abbreviations and Terms 4

1.4 Standards 4

1.5 References 4

1.6 Open Issues 5

1.7 Limitations 5

2. RemoteControl Plugin 6

2.1 Configuration 6

2.2 Key Mapping File **Error! Bookmark not defined.**

2.3 Application Programming Interface (API) 7

2.3.1 General information 7

2.3.2 Device actions **Error! Bookmark not defined.**

2.3.3 Key actions **Error! Bookmark not defined.**

2.3.4 Key mapping actions **Error! Bookmark not defined.**

2.4 Events 7

2.5 JSON definitions 8

2.5.1 General information 8

2.5.2 Codes for key actions **Error! Bookmark not defined.**

# Introduction

## Scope

This document describes the Plugin Monitor API interface. This plugin can be configured to be loaded and executed in the WPEFramework and offers functionality to monitor memory consumption and operability of other plugins on the platform. The plugin can also be configured to deactivate plugins that are exceeding configured memory values, or activate plugins again that where deactivated due to memory exceeding or failure. For details on the WPEFramework API, refer to: [WPEF]

## Case sensitivity

All identifiers on the interface described here are case-sensitive. E.g. an id known in the plugin as 'C0FFEE' is not the same as 'c0ffee'.

All keywords, entities, properties, relations and actions should be treated as case-sensitive.

## Acronyms, Abbreviations and Terms

The next list provides an overview of acronyms and abbreviations used in this document and their definitions.

|  |  |
| --- | --- |
| **Acronym** | **Definitions** |
| API | Application Programming Interface |
| JSON | JavaScript Object Notation |

Below terms are listed with their definitions, as used in this document.

|  |  |
| --- | --- |
| **Term** | **Definitions** |
| Callsign | The callsign is the name given to an instance of a plugin. One plugin can be instantiated multiple times, but each instance the instance name, callsign, must be unique. |

## Standards

Date time formats between the systems shall be in UTC time and W3C (ISO 8601 profile) formatting [ISO 8601], e.g.: 2004-11-05T13:15:30Z. This way time discontinuities can be avoided due to daylight savings. Note that all interfacing systems must decode/encode the date time to the correct local time.

Languages used in the WPEFramework will be conform [ISO 639-1] using two letter language codes. If WPEFramework encounters a language code it does not recognize, it will use ‘xx’ instead. For a list of available two letter ISO language codes, please visit:  
<http://www.loc.gov/standards/iso639-2/php/code_list.php>

## References

This section lists the references made in this document:

|  |  |
| --- | --- |
| [WPEF] | WPEFramework API Reference  <https://github.com/WebPlatformForEmbedded/WPEFramework> |
| [HTTP] | Hypertext Transfer Protocol  <http://www.w3.org/Protocols> |
| [ISO 8601] | Date and time format  http://www.iso.org/iso/date\_and\_time\_format |
| [ISO-3166] | Country code specification  <http://www.iso.org/iso/country_codes.htm> |
| [ISO-639-1] | Language code specification (Alpha-2 code)  <http://www.loc.gov/standards/iso639-2/php/code_list.php> |
| [JSON] | JavaScript Object Notation  <http://www.json.org> |
| [URLENC] | URL Encoding  <http://www.w3schools.com/tags/ref_urlencode.asp> |

## Open Issues

This is a list of open issues that needs to be resolved:

* This document is still a work in progress.

## Limitations

The information described in this document is preliminary and subject to change in the future.

Legend:

****

**Be aware of:** implementation choice is needed or side-effect needs to be handled.



**Implementation advice:** Guide line for implementation mostly related to performance.

# Monitor Plugin

## Configuration

|  |  |
| --- | --- |
| callsign | [string] the instance name for the plugin. Default: Monitor. |
| classname | [string] Monitor. |
| locator | [string] libWPEFrameworkMonitor.so |
| autostart | [bool] should the monitor plugin be instantiated at the moment the WPEFramework is starts up. |
| configuration | [JSON] JSON object specifying the exact configuration for this plugin. See the next paragraph for details. |

Configuration of the Monitor:

|  |  |
| --- | --- |
| observables | [array] array of configuration objects for each plugin to be monitored. |

Observable settings:

|  |  |
| --- | --- |
| callsign | [string] the name of specific plugin (call sign) that should be monitored. |
| memory | [uint32] periodicity, in seconds, when a memory measurement is taken. |
| operational | [sint32] periodicity, in seconds, when a check for the operational state is taken. If the time is negative, the operational state is monitored but there is no action taken if the plugin is not operational. |
| memorylimit | [uint32] the maximum value that the memory is allowed to use. If this value is exceeded, the plugin is deactivated. If memory limit is set to 0, the memory is only measured. No action is taken if the plugin exceeds this limit. |

A plugin can only be monitored if the plugin implements the IExchange::IMemory interface, to be queried via the IPlugin interface. See: <https://github.com/WebPlatformForEmbedded/WPEFramework/blob/master/Source/interfaces/IMemory.h>

for more details.

Plugins supporting this interface are:

* WebKitBrowser
* Netflix
* WebServer

****If the operational parameter is < 0 the monitor will also ***not*** start the plugin in case the plugin is deactivated with the reason of Memory Exceeded or Failure. If the operational parameter == 0 the monitor will ***not*** stop the plugin in case it becomes in-operable, but it will start the plugin again if it is deactivated with the reason of Memory Exceeded or Failure!!

## Application Programming Interface (API)

### General information

Using this method, all monitored objects can be retrieved and the associated information.

|  |  |
| --- | --- |
| Request: | GET /Service/Monitor |
| Success: | HTTP/1.1 200 List of monitored plugins  { observer\_list } |
| Failure |  |

|  |  |
| --- | --- |
| Request: | GET /Service/Monitor/<callsign> |
| Success: | HTTP/1.1 200 Monitored plugin statistics  { measurement } |
| Failure | HTTP/1.1 400 Bad request |

### Resetting the statistics

|  |  |
| --- | --- |
| Request: | PUT /Service/Monitor/<callsign> |
| Success: | HTTP/1.1 200 OK  { measurement } |
| Failure | HTTP/1.1 400 Bad request |

All collected information is cleared. The response contains the information just prior to the clear.

## Events

Whenever a plugin that is monitored is deactivated due to an in-operable state or due to exceeding memory usage, it is notified via an event. Also if the monitor is configures to activate a failing plugin, which it monitors, send out an activation event.

|  |  |
| --- | --- |
| callsign | [string] callsign of the plugin applicable for this event. |
| action | [enum] [“Activate”, “Deactivate”] |
| reason | [enum] activate: [“Automatic”]  deactivate: [“MemoryExceeded”, “Failure”] |

## JSON definitions

### General information

All memory measurement are in bytes.

observer\_list:

|  |  |
| --- | --- |
| - | [array] JSON array which consists of observe\_item elements. |

observe\_item:

|  |  |
| --- | --- |
| name | [string] callsign of the plugin being observed. |
| measurement | [JSON] metadata on memory consumption and operational state of the plugin. |

measurement:

|  |  |
| --- | --- |
| allocated {info} | [JSON] virtual memory information as gathered during the last measurement. |
| resident {info} | [JSON] RSS memory information as gathered during the last measurement. |
| shared {info} | [JSON] shared memory information as gathered during the last measurement. |
| process {info} | [JSON] processes allocated to the observed plugin |
| operational | [bool] Was the plugin operational during the last measurement. |
| count | [uint32] number of measurements taken since the plugin is monitored. |

info:

|  |  |
| --- | --- |
| min | [uint64] lowest number measured since the plugin is monitored. |
| max | [uint64] highest number measured since the plugin is monitored. |
| average | [uint64] average number measured since the plugin is monitored. |
| last | [uint64] number measured during the last measurement. |