

MeshAgent / modules / win-info.js

krayon007 Updated internal documentation f93610e · 2 years ago History

```
1  /*
2  Copyright 2019-2020 Intel Corporation
3
4  Licensed under the Apache License, Version 2.0 (the "License");
5  you may not use this file except in compliance with the License.
6  You may obtain a copy of the License at
7
8      http://www.apache.org/licenses/LICENSE-2.0
9
10 Unless required by applicable law or agreed to in writing, software
11 distributed under the License is distributed on an "AS IS" BASIS,
12 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13 See the License for the specific language governing permissions and
14 limitations under the License.
15 */
16
17
18 //
19 // This module fetches various Windows System information, such as pending reboot statu
20 // volume defrag state, installed applications, windows update status, etc
21 //
22
23
24 var promise = require('promise');
25
26 //
27 // This function queries WMI to fetch Windows Update Status
28 //
29 function qfe()
30 {
31     var child = require('child_process').execFile(process.env['windir'] + '\\System32\\
32     child.stdout.str = ''; child.stdout.on('data', function (c) { this.str += c.toStrin
33     child.stderr.str = ''; child.stderr.on('data', function (c) { this.str += c.toStrin
34     child.waitForExit();
35
36     var lines = child.stdout.str.trim().split('\r\n');
```

MeshAgent / modules / win-info.js

Top

Code Blame 260 lines (236 loc) · 9.16 KB

Raw Copy Download Compare



































```
29     function qfe()
42         for (i = 1; i < lines.length; ++i)
43         {
44             var obj = {};
45             tokens = lines[i].split(',');
46             for (key = 0; key < keys.length; ++key)
47             {
48                 if (tokens[key]) { obj[keys[key]] = tokens[key]; }
49             }
50             result.push(obj);
51         }
52         return (result);
53     }
54
55     // This function uses Windows Powershell to fetch metadata about the currently configur
56     function av()
57     {
```

Files

52cf129

Go to file

- .github
- docs
- lib-jpeg-turbo
- meshconsole
- meshcore
- meshreset
- meshservice
- microscript

- >  microstack
- ▼  modules
- >  utils
-  AgentHashTool.js
-  CSP.js
-  DeviceManager.js
-  MSH_Installer.js
-  PE_Parser.js
-  PostBuild.js
-  RecoveryCore.js
-  _agentNodeId.js
-  _agentStatus.js
-  agent-installer.js
-  agent-selftest.js
-  amt-lme.js
-  amt-mei.js
-  amt-scanner.js
-  amt-script.js
-  amt-wsman-duk.js
-  amt-wsman.js
-  amt-xml.js
-  amt.js
-  amt_heci.js
-  awk-helper.js
-  child-container.js
-  clipboard.js
-  code-utils.js
-  crc32-stream.js
-  daemon.js
-  dbTool.js
-  default_route.js
-  desktop-lock.js
-  dhcp.js
-  duktape-debugger.js

```
37 1
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58     var child = require('child_process').execFile(process.env['windir'] + '\\System32\\
59     if (child == null) { return ([]); }
60
61
62     child.descriptorMetadata = 'process-manager';
63     child.stdout.str = ''; child.stdout.on('data', function (c) { this.str += c.toStrin
64     child.stderr.str = ''; child.stderr.on('data', function (c) { this.str += c.toStrin
65
66     child.stdin.write(['reflection.Assembly']::LoadWithPartialName("system.core")\r\n');
67     child.stdin.write('Get-WmiObject -Namespace "root/SecurityCenter2" -Class AntiVirus
68     child.stdin.write('ForEach-Object -Process { ');
69     child.stdin.write('$Bytes = [System.Text.Encoding]::UTF8.GetBytes($_.displayName);
70     child.stdin.write('$EncodedText =[Convert]::ToBase64String($Bytes); ');
71     child.stdin.write('Write-Host ("{0},{1}" -f $_.productState,$EncodedText); }\r\n');
72     child.stdin.write('exit\r\n');
73     child.waitForExit();
74
75
76     if (child.stdout.str == '') { return ([]); }
77
78     var lines = child.stdout.str.trim().split('\r\n');
79     var result = [];
80     for (i = 0; i < lines.length; ++i)
81     {
82         var keys = lines[i].split(',');
83         if(keys.length == 2)
84         {
85             var status = {};
86             status.product = Buffer.from(keys[1], 'base64').toString();
87             status.updated = (parseInt(keys[0]) & 0x10) == 0;
88             status.enabled = (parseInt(keys[0]) & 0x1000) == 0x1000;
89             result.push(status);
90         }
91     }
92     return (result);
93 }
94
95 //
96 // This function uses the defrag system utility to query defrag state of the specified
97 //
98 // Note: options.volume must be specified
99
100 function defrag(options)
101 {
102     var ret = new promise(function (res, rej) { this._res = res; this._rej = rej; });
103     var path = '';
104
105     switch(require('os').arch())
106     {
107         case 'x64':
108             if (require('_GenericMarshal').PointerSize == 4)
109             {
110                 // 32 Bit App on 64 Bit Windows
111                 ret._rej('Cannot defrag volume on 64 bit Windows from 32 bit applicatio
112                 return (ret);
113             }
114             else
115             {
116                 // 64 Bit App
117                 path = process.env['windir'] + '\\System32\\defrag.exe';
118             }
119             break;
120         case 'ia32':
121             // 32 Bit App on 32 Bit Windows
122             path = process.env['windir'] + '\\System32\\defrag.exe';
123             break;
124         default:
125             ret._rej(require('os').arch() + ' not supported');
126             return (ret);
127             break;
128     }
129
130     ret.child = require('child_process').execFile(process.env['windir'] + '\\System32\\
131     ret.child.promise = ret;
132     ret.child.promise.options = options;
133     ret.child.stdout.str = ''; ret.child.stdout.on('data', function (c) { this.str += c
134     ret.child.stderr.str = ''; ret.child.stderr.on('data', function (c) { this.str += c
```

```
132     ret.child.on('exit', function (code)
133     {
134         var lines = this.stdout.str.trim().split('\r\n');
135         var obj = { volume: this.promise.options.volume };
136         for (var i in lines)
137         {
138             var token = lines[i].split('=');
139             if(token.length == 2)
140             {
141                 switch(token[0].trim().toLowerCase())
142                 {
143                     case 'volume size':
144                         obj['size'] = token[1];
145                         break;
146                     case 'free space':
147                         obj['free'] = token[1];
148                         break;
149                     case 'total fragmented space':
150                         obj['fragmented'] = token[1];
151                         break;
152                     case 'largest free space size':
153                         obj['largestFragment'] = token[1];
154                         break;
155                 }
156             }
157         }
158         this.promise._res(obj);
159     });
160     return (ret);
161 }
162
163 // Helper/Shortcut for registry query
164 ✓ function regQuery(H, Path, Key)
165 {
166     try
167     {
168         return(require('win-registry').QueryKey(H, Path, Key));
169     }
170     catch(e)
171     {
172         return (null);
173     }
174 }
175
176 // Returns a promise that fetches the list of installed applications
177
178 {
179     ret = 'Windows Update';
180 }
181 else if ((tmp=regQuery(HKEY.LocalMachine, 'SYSTEM\\CurrentControlSet\\Control\\Sess
182 {
183     ret = 'File Rename';
184 }
185 else if (regQuery(HKEY.LocalMachine, 'SYSTEM\\CurrentControlSet\\Control\\ComputerN
186 {
187     ret = 'System Rename';
188 }
189 return (ret);
190 }
191
192 //
193 // Returns a promise that fetches the list of installed applications
194 //
195 ✓ function installedApps()
196 {
197     var promise = require('promise');
```

```
207     var ret = new promise(function (a, r) { this._resolve = a; this._reject = r; });
208
209     var code = "\
210     var reg = require('win-registry');\
211     var result = [];\
212     var val, tmp;\
213     var items = reg.QueryKey(reg.HKEY.LocalMachine, 'SOFTWARE\\\\\\\\Microsoft\\\\\\\\Windows\\\\\
214     for (var key in items.subkeys)\
215     {\
216         val = {};\
217         try\
218         {\
219             val.name = reg.QueryKey(reg.HKEY.LocalMachine, 'SOFTWARE\\\\\\\\Microsoft\\\\\\\\Wi
220         }\
221         catch(e)\
222         {\
223             continue;\
224         }\
225         try\
226         {\
227             val.version = reg.QueryKey(reg.HKEY.LocalMachine, 'SOFTWARE\\\\\\\\Microsoft\\\\\
228             if (val.version == '') { delete val.version; }\
229         }\
230         catch(e)\
231         {\
232         }\
233         try\
234         {\
235             val.location = reg.QueryKey(reg.HKEY.LocalMachine, 'SOFTWARE\\\\\\\\Microsoft\\\\\
236             if (val.location == '') { delete val.location; }\
237         }\
238         catch(e)\
239         {\
240         }\
241         result.push(val);\
242     }\
243     console.log(JSON.stringify(result, '', 1));process.exit();"
244
245     ret.child = require('child_process').execFile(process.execPath, [process.execPath.s
246     ret.child.promise = ret;
247     ret.child.stdout.str = ''; ret.child.stdout.on('data', function (c) { this.str += c
248     ret.child.on('exit', function (c) { this.promise._resolve(JSON.parse(this.stdout.st
249     return (ret);
250 }
251
252 if (process.platform == 'win32')
253 {
254     module.exports = { qfe: qfe, av: av, defrag: defrag, pendingReboot: pendingReboot,
255 }
256 else
257 {
258     var not_supported = function () { throw (process.platform + ' not supported'); };
259     module.exports = { qfe: not_supported, av: not_supported, defrag: not_supported, pe
260 }
```