

DSHENG4 GEM 通信エンジン  
ソフトウェア・パッケージ

DSHEng4Class

## クラス・ライブラリ説明書

Vol-1 エンジン起動と管理情報クラス 編

Part-2 第14章 ～ 21章

2014年5月 (改-6)

株式会社データマップ

**[取り扱い注意]**

- ・ この資料ならびにソフトウェアの一部または全部を無断で使用、複製することはできません。
- ・ 本説明書に記述されている内容は予告なしで変更される可能性があります。
- ・ Windows は米国 Microsoft Corporation の登録商標です。
- ・ ユーザーが本ソフトウェアの使用によって生じた遺失履歴、(株) データマップの予見の有無を問わず発生した特別損害、付随的損害、間接損害およびその他の拡大損害に対して責任を負いません。

**【改訂履歴】**

| 番号 | 改訂日付       | 項目  | 概略   |
|----|------------|---|--|
| 1. | 2010年3月    | 初版  |  |
| 2. | 2010年6月    | 20.2 を追加                                  | DshPtrToData クラスの追加<br>IntPtr に保存されているデータをフォーマットに対応するプロパティにコピーするクラスを追加した。                            |
| 3  | 2011/02/07 | クラスに Dispose メソッドとクラスのトレースモニターと表示機能を実装した。 | クラスに Dispose メソッドを追加した。<br>また、デバッグ機能強化のためユーザーがクラスの生成、消滅の回数を管理し、それらのタイミングをトレース表示できるようにした。             |
| 4  | 2011/02/07 | DshDebug クラスに class trace 表示機能を追加         | 21. クラスライブラリの DshDebug クラスにクラスのトレース機能を追加した。。   |
| 5  | 2012/06/05 | DshFPP クラスを追加した。                          | 22. フォーマット付きプロセス・プログラムクラス DshFPP を追加した。  |
| 6  | 2014/05/30 | 文書番号の変更                                   | クラスライブラリプログラミングガイドの<br>文書番号を DSEng4-09-30305-03 から<br>DSEng4-09-30365-03 に変更した。<br>(他のドキュメントと重複していた) |

|  |           |
|--|-----------|
| <b>14. コントロール・ジョブ関連クラス</b> .....           | <b>1</b>  |
| <b>14. 1 DshCj クラス</b> .....               | <b>2</b>  |
| 14. 1. 1 コンストラクタ .....                     | 2         |
| 14. 1. 2 プロパティ .....                       | 2         |
| 14. 1. 3 メソッド .....                        | 3         |
| 14. 1. 3. 1 set_id() .....                 | 4         |
| 14. 1. 3. 2 init_set() .....               | 4         |
| 14. 1. 3. 3 add_attr() .....               | 5         |
| 14. 1. 3. 4 alloc_id() .....               | 5         |
| 14. 1. 3. 5 set() .....                    | 6         |
| 14. 1. 3. 6 get() .....                    | 6         |
| 14. 1. 3. 7 delete() .....                 | 7         |
| 14. 1. 3. 8 copy() .....                   | 7         |
| 14. 1. 3. 9 decode() .....                 | 8         |
| 14. 1. 3. 10 get_id_count() .....          | 9         |
| 14. 1. 3. 11 get_id_list() .....           | 9         |
| 14. 1. 3. 12 clear() .....                 | 10        |
| 14. 1. 3. 13 Dispose() .....               | 10        |
| <b>14. 2 DshCjCmd クラス</b> .....            | <b>11</b> |
| 14. 2. 1 コンストラクタ .....                     | 11        |
| 14. 2. 2 プロパティ .....                       | 12        |
| 14. 2. 3 メソッド .....                        | 13        |
| 14. 2. 3. 1 set_cmd_info() .....           | 13        |
| 14. 2. 3. 2 decode() .....                 | 15        |
| <b>14. 3 DshCjAttr クラス</b> .....           | <b>16</b> |
| 14. 3. 1 コンストラクタ .....                     | 16        |
| 14. 3. 2 プロパティ .....                       | 17        |
| 14. 3. 3 メソッド .....                        | 18        |
| 14. 3. 3. 1 clear() .....                  | 19        |
| 14. 3. 3. 2 set_string() .....             | 19        |
| 14. 3. 3. 3 set_num() .....                | 20        |
| 14. 3. 3. 4 add_MaterialOutStatus() .....  | 20        |
| 14. 3. 3. 5 add_slot() .....               | 21        |
| 14. 3. 3. 6 add_MaterialOutSpec() .....    | 21        |
| 14. 3. 3. 7 add_src_slot() .....           | 22        |
| 14. 3. 3. 8 add_dst_slot() .....           | 22        |
| 14. 3. 3. 9 add_prj_proc() .....           | 23        |
| 14. 3. 3. 10 init_prj_proc() .....         | 23        |
| 14. 3. 3. 11 init_PrjStateList() .....     | 24        |
| 14. 3. 3. 12 init_PauseEvent() .....       | 24        |
| 14. 3. 3. 13 init_Text() .....             | 25        |
| 14. 3. 3. 14 set_string_attr() .....       | 25        |
| 14. 3. 3. 15 set_num_attr() .....          | 26        |
| 14. 3. 3. 16 get_attrid_from_index() ..... | 26        |
| 14. 3. 3. 17 copy() .....                  | 27        |
| <b>14. 4 DshCjAttrList クラス</b> .....       | <b>28</b> |

|               |                                       |           |
|---------------|---------------------------------------|-----------|
| 14. 4. 1      | コンストラクタ .....                         | 28        |
| 14. 4. 2      | プロパティ .....                           | 28        |
| 14. 4. 3      | メソッド .....                            | 28        |
| 14. 4. 3. 1   | clear() .....                         | 29        |
| 14. 4. 3. 2   | add_attr() .....                      | 29        |
| <b>14. 5</b>  | <b>DshStatusPara クラス .....</b>        | <b>30</b> |
| 14. 5. 1      | コンストラクタ .....                         | 30        |
| 14. 5. 2      | プロパティ .....                           | 30        |
| 14. 5. 3      | メソッド .....                            | 31        |
| 14. 5. 3. 1   | free() .....                          | 31        |
| 14. 5. 3. 2   | set_para() .....                      | 32        |
| 14. 5. 3. 3   | copy() .....                          | 32        |
| <b>14. 6</b>  | <b>DshStatusParaList クラス .....</b>    | <b>33</b> |
| 14. 6. 1      | コンストラクタ .....                         | 33        |
| 14. 6. 2      | プロパティ .....                           | 33        |
| 14. 6. 3      | メソッド .....                            | 34        |
| 14. 6. 3. 1   | clear() .....                         | 34        |
| 14. 6. 3. 2   | add_para() .....                      | 35        |
| 14. 6. 3. 3   | copy() .....                          | 35        |
| <b>14. 7</b>  | <b>DshMaterialOutStatus クラス .....</b> | <b>36</b> |
| 14. 7. 1      | コンストラクタ .....                         | 36        |
| 14. 7. 2      | プロパティ .....                           | 36        |
| 14. 7. 3      | メソッド .....                            | 36        |
| 14. 7. 3. 1   | set_status() .....                    | 37        |
| 14. 7. 3. 2   | copy() .....                          | 37        |
| <b>14. 8</b>  | <b>DshMaterialOutSpec クラス .....</b>   | <b>38</b> |
| 14. 8. 1      | コンストラクタ .....                         | 38        |
| 14. 8. 2      | プロパティ .....                           | 38        |
| 14. 8. 3      | メソッド .....                            | 39        |
| 14. 8. 3. 1   | copy() .....                          | 39        |
| <b>14. 9</b>  | <b>DshCjCarSlotList クラス .....</b>     | <b>40</b> |
| 14. 9. 1      | コンストラクタ .....                         | 40        |
| 14. 9. 2      | プロパティ .....                           | 40        |
| 14. 9. 3      | メソッド .....                            | 41        |
| 14. 9. 3. 1   | clear() .....                         | 41        |
| 14. 9. 3. 2   | set_carid() .....                     | 42        |
| 14. 9. 3. 3   | add_slotid() .....                    | 42        |
| 14. 9. 3. 4   | copy_slotid() .....                   | 43        |
| 14. 9. 3. 5   | copy() .....                          | 43        |
| <b>14. 10</b> | <b>DshProcControlSpec クラス .....</b>   | <b>44</b> |
| 14. 10. 1     | コンストラクタ .....                         | 44        |
| 14. 10. 2     | プロパティ .....                           | 44        |
| 14. 10. 3     | メソッド .....                            | 45        |
| 14. 10. 3. 1  | clear() .....                         | 45        |
| 14. 10. 3. 2  | set_prjobid() .....                   | 46        |
| 14. 10. 3. 3  | add_control_rule() .....              | 46        |
| 14. 10. 3. 4  | add_output_rule() .....               | 47        |
| 14. 10. 3. 5  | copy() .....                          | 47        |

|  |           |
|--|-----------|
| <b>14. 11 DshControlRule クラス</b> ..... | <b>48</b> |
| 14. 11. 1 コンストラクタ.....                 | 48        |
| 14. 11. 2 プロパティ.....                   | 48        |
| 14. 11. 3 メソッド.....                    | 49        |
| 14. 11. 3. 1 clear().....              | 49        |
| 14. 11. 3. 2 add_control_rule().....   | 50        |
| 14. 11. 3. 3 copy().....               | 51        |
| <b>14. 12 DshOutputRule クラス</b> .....  | <b>52</b> |
| 14. 12. 1 コンストラクタ.....                 | 52        |
| 14. 12. 2 プロパティ.....                   | 52        |
| 14. 12. 3 メソッド.....                    | 53        |
| 14. 12. 3. 1 clear().....              | 53        |
| 14. 12. 3. 2 add_output_rule().....    | 54        |
| 14. 12. 3. 3 copy().....               | 54        |
| <b>14. 13 DshTextList クラス</b> .....    | <b>55</b> |
| 14. 13. 1 コンストラクタ.....                 | 55        |
| 14. 13. 2 プロパティ.....                   | 55        |
| 14. 13. 3 メソッド.....                    | 56        |
| 14. 13. 3. 1 clear().....              | 56        |
| 14. 13. 3. 2 add_text().....           | 57        |
| 14. 13. 3. 3 copy().....               | 57        |
| <b>14. 14 DshS14Rsp クラス</b> .....      | <b>58</b> |
| 14. 14. 1 コンストラクタ.....                 | 58        |
| 14. 14. 2 プロパティ.....                   | 58        |
| 14. 14. 3 メソッド.....                    | 59        |
| 14. 14. 3. 1 clear().....              | 59        |
| 14. 14. 3. 2 void init_set().....      | 60        |
| 14. 14. 3. 3 set_count().....          | 61        |
| 14. 14. 3. 4 add_err().....            | 61        |
| <b>14. 15 DshS16F27Rsp クラス</b> .....   | <b>62</b> |
| 14. 15. 1 コンストラクタ.....                 | 62        |
| 14. 15. 2 プロパティ.....                   | 62        |
| 14. 15. 3 メソッド.....                    | 63        |
| 14. 15. 3. 1 set_obj_err ().....       | 63        |
| <b>15. スプール関連クラス</b> .....             | <b>64</b> |
| <b>15. 1 DshSpool クラス</b> .....        | <b>65</b> |
| 15. 1. 1 コンストラクタ.....                  | 65        |
| 15. 1. 2 プロパティ.....                    | 65        |
| 15. 1. 3 メソッド.....                     | 66        |
| 15. 1. 3. 1 clear_all().....           | 67        |
| 15. 1. 3. 2 clear ().....              | 67        |
| 15. 1. 3. 3 reset().....               | 68        |
| 15. 1. 3. 4 add_func ().....           | 68        |
| 15. 1. 3. 5 set ().....                | 69        |
| 15. 1. 3. 6 get().....                 | 70        |
| <b>15. 2 DshSpoolList クラス</b> .....    | <b>71</b> |
| 15. 2. 1 コンストラクタ.....                  | 71        |
| 15. 2. 2 プロパティ.....                    | 71        |

|              |  |           |
|--------------|--|-----------|
| 15. 2. 3     | メソッド.....                                | 72        |
| 15. 2. 3. 1  | clear() .....                            | 72        |
| 14. 2. 3. 2  | decode() .....                           | 73        |
| <b>15. 3</b> | <b>DshSpoolUnit クラス .....</b>            | <b>74</b> |
| 15. 3. 1     | コンストラクタ .....                            | 74        |
| 15. 3. 2     | プロパティ .....                              | 74        |
| 15. 3. 3     | メソッド.....                                | 75        |
| 15. 3. 3. 1  | clear() .....                            | 75        |
| 15. 3. 3. 2  | add_func () .....                        | 76        |
| <b>15. 4</b> | <b>DshSpoolRsp クラス .....</b>             | <b>77</b> |
| 15. 4. 1     | コンストラクタ .....                            | 77        |
| 15. 4. 2     | プロパティ .....                              | 77        |
| 15. 4. 3     | メソッド.....                                | 78        |
| 15. 4. 3. 1  | clear() .....                            | 78        |
| 15. 4. 3. 2  | add() .....                              | 79        |
| <b>15. 5</b> | <b>DshSpoolRspUnit クラス.....</b>          | <b>80</b> |
| 15. 5. 1     | コンストラクタ .....                            | 80        |
| 15. 5. 2     | プロパティ .....                              | 80        |
| 15. 5. 3     | メソッド.....                                | 80        |
| <b>16.</b>   | <b>ホストコマンド、キャリアアクションメッセージ関連クラス .....</b> | <b>81</b> |
| <b>16. 1</b> | <b>DshRCmd クラス .....</b>                 | <b>82</b> |
| 16. 1. 1     | コンストラクタ .....                            | 82        |
| 16. 1. 2     | プロパティ .....                              | 82        |
| 16. 1. 3     | メソッド.....                                | 83        |
| 16. 1. 3. 1  | clear() .....                            | 83        |
| 16. 1. 3. 2  | set_cmd() .....                          | 84        |
| 16. 1. 3. 3  | add_para() .....                         | 84        |
| 16. 1. 3. 4  | add_sub_para() .....                     | 85        |
| 16. 1. 3. 5  | copy() .....                             | 85        |
| 16. 1. 3. 6  | decode() .....                           | 86        |
| <b>16. 2</b> | <b>DshERCmd クラス .....</b>                | <b>87</b> |
| 16. 2. 1     | コンストラクタ .....                            | 87        |
| 16. 2. 2     | プロパティ .....                              | 88        |
| 16. 2. 3     | メソッド.....                                | 89        |
| 16. 2. 3. 1  | clear() .....                            | 89        |
| 16. 2. 3. 2  | set_cmd() .....                          | 90        |
| 16. 2. 3. 3  | add_para() .....                         | 91        |
| 16. 2. 3. 4  | add_sub_para() .....                     | 92        |
| 16. 2. 3. 5  | set_multi() .....                        | 93        |
| 16. 2. 3. 6  | add_multi_para() .....                   | 94        |
| 16. 2. 3. 7  | copy() .....                             | 95        |
| 16. 2. 3. 8  | decode() .....                           | 96        |
| <b>16. 3</b> | <b>DshCarAction クラス .....</b>            | <b>97</b> |
| 16. 3. 1     | コンストラクタ .....                            | 97        |
| 16. 3. 2     | プロパティ .....                              | 98        |
| 16. 3. 3     | メソッド.....                                | 99        |
| 16. 3. 3. 1  | clear() .....                            | 99        |
| 16. 3. 3. 2  | init_set() .....                         | 100       |

|              |                               |            |
|--------------|-------------------------------|------------|
| 16. 3. 3. 3  | add_para()                    | 101        |
| 16. 3. 3. 4  | get_attr_pos ()               | 102        |
| 16. 3. 3. 5  | copy()                        | 102        |
| 16. 3. 3. 6  | decode()                      | 103        |
| <b>16. 4</b> | <b>DshCarActionPara クラス</b>   | <b>104</b> |
| 16. 4. 1     | コンストラクタ                       | 104        |
| 16. 4. 2     | プロパティ                         | 105        |
| 16. 4. 3     | メソッド                          | 106        |
| 16. 4. 3. 1  | clear()                       | 106        |
| 16. 4. 3. 2  | set_info()                    | 107        |
| 16. 4. 3. 3  | add_content_map()             | 108        |
| 16. 4. 3. 4  | add_slot_map()                | 108        |
| 16. 4. 3. 5  | copy()                        | 109        |
| <b>16. 5</b> | <b>DshContentMap クラス</b>      | <b>110</b> |
| 16. 5. 1     | コンストラクタ                       | 110        |
| 16. 5. 2     | プロパティ                         | 110        |
| 16. 5. 3     | メソッド                          | 111        |
| 16. 5. 3. 1  | clear()                       | 111        |
| 16. 5. 3. 2  | add_info()                    | 112        |
| 16. 5. 3. 3  | copy()                        | 112        |
| <b>16. 6</b> | <b>DshSlotMap クラス</b>         | <b>113</b> |
| 16. 6. 1     | コンストラクタ                       | 113        |
| 16. 6. 2     | プロパティ                         | 113        |
| 16. 6. 3     | メソッド                          | 114        |
| 16. 6. 3. 1  | clear()                       | 114        |
| 16. 6. 3. 2  | add_info()                    | 115        |
| 16. 6. 3. 3  | copy()                        | 115        |
| <b>16. 7</b> | <b>DshPortGroupAction クラス</b> | <b>116</b> |
| 16. 7. 1     | コンストラクタ                       | 116        |
| 16. 7. 2     | プロパティ                         | 116        |
| 16. 7. 3     | メソッド                          | 117        |
| 16. 7. 3. 1  | clear()                       | 117        |
| 16. 7. 3. 2  | set_info()                    | 118        |
| 16. 7. 3. 3  | add_para()                    | 119        |
| 16. 7. 3. 4  | copy()                        | 119        |
| 16. 7. 3. 5  | decode()                      | 120        |
| <b>16. 8</b> | <b>DshPortAction クラス</b>      | <b>121</b> |
| 16. 8. 1     | コンストラクタ                       | 121        |
| 16. 8. 2     | プロパティ                         | 121        |
| 16. 8. 3     | メソッド                          | 122        |
| 16. 8. 3. 1  | clear()                       | 122        |
| 16. 8. 3. 2  | set_info()                    | 123        |
| 16. 8. 3. 3  | add_para()                    | 124        |
| 16. 8. 3. 4  | copy()                        | 124        |
| 16. 8. 3. 5  | decode()                      | 125        |
| <b>16. 9</b> | <b>DshPortAccess クラス</b>      | <b>126</b> |
| 16. 9. 1     | コンストラクタ                       | 126        |
| 16. 9. 2     | プロパティ                         | 126        |

|                                   |                                  |            |
|-----------------------------------|----------------------------------|------------|
| 16. 9. 3                          | メソッド.....                        | 127        |
| 16. 9. 3. 1                       | clear() .....                    | 127        |
| 16. 9. 3. 2                       | set_info() .....                 | 128        |
| 16. 9. 3. 3                       | add_port() .....                 | 128        |
| 16. 9. 3. 4                       | copy() .....                     | 129        |
| 16. 9. 3. 5                       | decode() .....                   | 130        |
| <b>16. 10</b>                     | <b>DshRCmdRsp クラス .....</b>      | <b>131</b> |
| 16. 10. 1                         | コンストラクタ .....                    | 131        |
| 16. 10. 2                         | プロパティ .....                      | 131        |
| 16. 10. 3                         | メソッド.....                        | 132        |
| 16. 10. 3. 1                      | clear() .....                    | 132        |
| 16. 10. 3. 2                      | add_ack() .....                  | 133        |
| 16. 10. 3. 3                      | copy() .....                     | 133        |
| <b>16. 11</b>                     | <b>DshCpAck クラス .....</b>        | <b>134</b> |
| 16. 11. 1                         | コンストラクタ .....                    | 134        |
| 16. 11. 2                         | プロパティ .....                      | 134        |
| 16. 11. 3                         | メソッド.....                        | 134        |
| <b>16. 12</b>                     | <b>DshCarActionRsp クラス.....</b>  | <b>135</b> |
| 16. 12. 1                         | コンストラクタ .....                    | 135        |
| 16. 12. 2                         | プロパティ .....                      | 135        |
| 16. 12. 3                         | メソッド.....                        | 136        |
| 16. 12. 3. 1                      | clear() .....                    | 136        |
| 16. 12. 3. 2                      | set_count() .....                | 137        |
| 16. 12. 3. 3                      | add_err() .....                  | 137        |
| <b>16. 13</b>                     | <b>DshPortActionRsp クラス.....</b> | <b>138</b> |
| 16. 13. 1                         | コンストラクタ .....                    | 138        |
| 16. 13. 2                         | プロパティ .....                      | 138        |
| 16. 13. 3                         | メソッド.....                        | 139        |
| 16. 13. 3. 1                      | clear() .....                    | 139        |
| 16. 13. 3. 2                      | set_count() .....                | 140        |
| 16. 13. 3. 3                      | add_err() .....                  | 140        |
| <b>16. 14</b>                     | <b>DshAccessError クラス.....</b>   | <b>141</b> |
| 16. 14. 1                         | コンストラクタ .....                    | 141        |
| 16. 14. 2                         | プロパティ .....                      | 141        |
| 16. 14. 3                         | メソッド.....                        | 141        |
| 16. 14. 3. 1                      | set_err () .....                 | 142        |
| <b>16. 15</b>                     | <b>DshPortAccessRsp クラス.....</b> | <b>143</b> |
| 16. 15. 1                         | コンストラクタ .....                    | 143        |
| 16. 15. 2                         | プロパティ .....                      | 143        |
| 16. 15. 3                         | メソッド.....                        | 144        |
| 16. 15. 3. 1                      | clear() .....                    | 144        |
| 16. 15. 3. 2                      | add_err() .....                  | 145        |
| <b>17. 端末表示メッセージ関連クラス .....</b>   |                                  | <b>146</b> |
| <b>17. 1 DshTermMsg クラス .....</b> |                                  | <b>147</b> |
| 17. 1. 1                          | コンストラクタ .....                    | 147        |
| 17. 1. 2                          | プロパティ .....                      | 147        |
| 17. 1. 3                          | メソッド.....                        | 148        |
| 17. 1. 3. 1                       | set_info() .....                 | 148        |



|              |                              |            |
|--------------|------------------------------|------------|
| 17. 1. 3. 2  | decode()                     | 149        |
| <b>17. 2</b> | <b>DshTermMultiMsg クラス</b>   | <b>150</b> |
| 17. 2. 1     | コンストラクタ                      | 150        |
| 17. 2. 2     | プロパティ                        | 150        |
| 17. 2. 3     | メソッド                         | 151        |
| 17. 2. 3. 1  | set_tid()                    | 151        |
| 17. 2. 3. 2  | clear()                      | 152        |
| 17. 2. 3. 3  | add()                        | 152        |
| 17. 2. 3. 4  | copy()                       | 153        |
| 17. 2. 3. 5  | decode()                     | 154        |
| <b>18.</b>   | <b>オブジェクト属性、パラメータ情報関連クラス</b> | <b>155</b> |
| <b>18. 1</b> | <b>DshObjPara クラス</b>        | <b>156</b> |
| 18. 1. 1     | コンストラクタ                      | 156        |
| 18. 1. 2     | プロパティ                        | 156        |
| 18. 1. 3     | メソッド                         | 157        |
| 18. 1. 3. 1  | free()                       | 157        |
| 18. 1. 3. 2  | set_para ()                  | 158        |
| 18. 1. 3. 3  | copy()                       | 158        |
| <b>18. 2</b> | <b>DshObjSubPara クラス</b>     | <b>159</b> |
| 18. 2. 1     | コンストラクタ                      | 159        |
| 18. 2. 2     | プロパティ                        | 159        |
| 18. 2. 3     | メソッド                         | 159        |
| 18. 2. 3. 1  | clear()                      | 160        |
| 18. 2. 3. 2  | add()                        | 160        |
| 18. 2. 3. 3  | copy()                       | 161        |
| <b>18. 3</b> | <b>DshDataItem クラス</b>       | <b>162</b> |
| 18. 3. 1     | コンストラクタ                      | 162        |
| 18. 3. 2     | プロパティ                        | 162        |
| 18. 3. 3     | メソッド                         | 163        |
| 18. 3. 3. 1  | free()                       | 163        |
| 18. 3. 3. 2  | set_data()                   | 164        |
| 18. 3. 3. 3  | copy()                       | 165        |
| <b>18. 4</b> | <b>DshDataItemList クラス</b>   | <b>166</b> |
| 18. 4. 1     | コンストラクタ                      | 166        |
| 18. 4. 2     | プロパティ                        | 166        |
| 18. 4. 3     | メソッド                         | 167        |
| 18. 4. 3. 1  | clear()                      | 167        |
| 18. 4. 3. 2  | add_data()                   | 168        |
| <b>18. 5</b> | <b>DshStrList クラス</b>        | <b>169</b> |
| 18. 5. 1     | コンストラクタ                      | 169        |
| 18. 5. 2     | プロパティ                        | 169        |
| 18. 5. 3     | メソッド                         | 169        |
| 18. 5. 3. 1  | clear()                      | 170        |
| 18. 5. 3. 2  | add()                        | 170        |
| 18. 5. 3. 3  | copy()                       | 171        |
| <b>19.</b>   | <b>エラー情報関連クラス</b>            | <b>172</b> |
| <b>19. 1</b> | <b>DshObjError クラス</b>       | <b>173</b> |
| 19. 1. 1     | コンストラクタ                      | 173        |

|              |                                   |            |
|--------------|-----------------------------------|------------|
| 19. 1. 2     | プロパティ                             | 173        |
| 19. 1. 3     | メソッド                              | 173        |
| 19. 1. 3. 1  | set_obj_err ()                    | 174        |
| 19. 1. 3. 2  | copy()                            | 174        |
| 19. 1. 3. 3  | copy_err_list()                   | 175        |
| <b>20.</b>   | <b>データアイテム関連、エラー情報コピークラス</b>      | <b>176</b> |
| <b>20. 1</b> | <b>DshLib クラス</b>                 | <b>176</b> |
| 20. 1. 1     | コンストラクタ                           | 176        |
| 20. 1. 2     | プロパティ                             | 176        |
| 20. 1. 3     | メソッド                              | 177        |
| 20. 1. 3. 1  | DshPtrToFmtDataString()           | 178        |
| 20. 1. 3. 2  | StringToFormatData()              | 179        |
| 20. 1. 3. 3  | DshPtrFmtDataInt ()               | 180        |
| 20. 1. 3. 4  | PtrToFormatData()                 | 181        |
| 20. 1. 3. 5  | DataToPtr()                       | 182        |
| 20. 1. 3. 6  | StartPerformanceTimer()           | 183        |
| 20. 1. 3. 7  | StopPerformanceTimer()            | 183        |
| 20. 1. 3. 8  | beep()                            | 184        |
| 20. 1. 3. 9  | DshPtrToString ()                 | 185        |
| 20. 1. 3. 10 | ptr_to_string ()                  | 186        |
| <b>20. 2</b> | <b>DshPtrToData クラス</b>           | <b>187</b> |
| 20. 2. 1     | コンストラクタ                           | 187        |
| 20. 2. 2     | プロパティ                             | 187        |
| 20. 2. 3     | メソッド                              | 188        |
| 20. 2. 3. 1  | ptr_to_value()                    | 188        |
| <b>21.</b>   | <b>DSHDEBUG - クラス・トレースのためのクラス</b> | <b>189</b> |
| <b>21. 1</b> | <b>DshDebug クラス</b>               | <b>189</b> |
| 21. 1. 1     | コンストラクタ                           | 190        |
| 21. 1. 2     | プロパティ                             | 190        |
| 21. 1. 3     | メソッド                              | 190        |
| 21. 1. 3. 1  | init_DshDebug()                   | 191        |
| 21. 1. 3. 2  | set_trace_form_handle ()          | 192        |
| 21. 1. 3. 3  | set_trace_WM()                    | 192        |
| 21. 1. 3. 4  | enable_trace_log()                | 193        |
| 21. 1. 3. 5  | get_class_name()                  | 193        |
| 21. 1. 3. 6  | set_trace_flag()                  | 194        |
| 21. 1. 3. 7  | get_trace_flag()                  | 194        |
| 21. 1. 3. 8  | reset_trace_count()               | 195        |
| 21. 1. 3. 9  | get_trace_count()                 | 195        |
| <b>22.</b>   | <b>フォーマット付きプロセス・プログラム関連クラス</b>    | <b>196</b> |
| <b>22. 1</b> | <b>DshFPP クラス</b>                 | <b>197</b> |
| 22. 1. 1     | コンストラクタ                           | 197        |
| 22. 2.       | プロパティ                             | 197        |
| 22. 1. 3     | メソッド                              | 198        |
| 22. 1. 3. 1  | set_id()                          | 199        |
| 22. 1. 3. 2  | set_mdln()                        | 199        |
| 22. 1. 3. 3  | set_softrev()                     | 200        |
| 22. 1. 3. 4  | add_ccode()                       | 200        |

|              |                                    |            |
|--------------|------------------------------------|------------|
| 22. 1. 3. 5  | set()                              | 201        |
| 22. 1. 3. 6  | get()                              | 201        |
| 22. 1. 3. 7  | alloc_id()                         | 202        |
| 22. 1. 3. 8  | delete()                           | 204        |
| 22. 1. 3. 9  | copy()                             | 204        |
| 22. 1. 3. 10 | decode()                           | 205        |
| 22. 1. 3. 11 | get_id_count()                     | 206        |
| 22. 1. 3. 12 | get_id_list()                      | 206        |
| 22. 1. 3. 13 | clear()                            | 207        |
| <b>22. 2</b> | <b>DshCCode プロセスコマンド・クラス</b>       | <b>208</b> |
| 22. 2. 1     | コンストラクタ                            | 208        |
| 22. 2. 2     | プロパティ                              | 208        |
| 22. 2. 3     | メソッド                               | 209        |
| 22. 2. 3. 1  | clear()                            | 209        |
| 22. 2. 3. 2  | add_para()                         | 210        |
| 22. 2. 3. 3  | copy()                             | 211        |
| <b>22. 3</b> | <b>DshPPara プロセスコマンド・パラメータ・クラス</b> | <b>212</b> |
| 22. 3. 1     | コンストラクタ                            | 212        |
| 22. 3. 2     | プロパティ                              | 212        |
| 22. 3. 3     | メソッド                               | 213        |
| 22. 3. 3. 1  | clear()                            | 213        |
| 22. 3. 3. 2  | set_data()                         | 214        |
| 22. 3. 3. 3  | copy()                             | 214        |

## 14. コントロール・ジョブ関連クラス

コントロールジョブ(CJ)関連情報を保存するクラスです。

関連クラスの一覧を次表に示します。

|    | クラス名                 | 用途  |
|----|----------------------|---|
| 1  | DshCj                | コントロールジョブ 情報保存クラスです。<br>S14F9                       |
| 2  | DshCjCmd             | コントロールジョブコマンド 情報保存用クラスです。<br>S16F27                 |
| 3  | DshCjAttr            | オブジェクトの情報を保存するためのクラスです。<br>コントロールジョブクラス内では配列で使用します。 |
| 4  | DshCjAttrList        | DshCjAttr クラスの配列リストクラスです。                           |
| 5  | DshStatusPara        | ステータスパラメータ値を保存するためのクラスです                            |
| 6  | DshStatusParaList    | DshStatusPara クラスの配列クラスです。                          |
| 7  | DshMaterialOutStatus | MtrlOutByStatus 属性情報を保存するクラスです。                     |
| 8  | DshMaterialOutSpec   | MtrlOutSpec 属性情報を保存するためのクラスです。                      |
| 9  | DshCjCarSlotList     | キャリアのスポット ID を配列で保存するためのクラスです。                      |
| 10 | DshProcControlSpec   | ProcessingCtrlSpec 属性情報を保存するためのクラスです。               |
| 11 | DshControlRule       | ProcessingCtrlSpec 属性のコントロールルール情報を保存するためのクラスです      |
| 12 | DshOutputRule        | ProcessingCtrlSpec 属性のアウトプットルール情報を保存するためのクラスです。     |
| 13 | DshTextList          | コントロールジョブの属性で、テキスト配列情報を保存するためのクラスです。                |

CJの通信関連クラスとして、以下のものがあります。 これらについては、Vol-2 で説明します。

|                   |
|-------------------|
| DshS14F9Send      |
| DshS14F10Response |
| DshS14F11Send     |
| DshS14F12Response |
| DshS16F27Send     |
| DshS16F28Response |

## 14. 1 DshCj クラス

コントロールジョブ情報を保存するために使用されます。  
通常、S14F9 メッセージを通して装置、ホスト間でやり取りされる情報です。

### 14. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                        | 説明   |
|---|---------------------------|--|
| 1 | public DshCj()            | インスタンスを生成します。<br>(後で、set_id メソッドでコントロールジョブ ID(cjid)を指定します) |
| 2 | public DshCj(string cjid) | インスタンスを cjid を指定して生成します。                                   |

### 14. 1. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                           | 説明   |
|---|------------------------------|--|
| 1 | public string cjid           | コントロールジョブ ID です。<br>エンジンはこの ID で管理します。       |
| 2 | public string name           | コントロールジョブ名です。cjid と同じ値です。                    |
| 3 | public string objspec        | オブジェクト仕様文字列です。                               |
| 4 | public string objtype        | オブジェクトのグループ またはクラスの ID です。                   |
| 5 | public int attr_count        | オブジェクト属性情報の数です。                              |
| 6 | public DshCjAttr[] attr_list | オブジェクト属性情報の配列リストです。<br>DshObjClass クラスの配列です。 |

### 14. 1. 3 メソッド

本クラスのメソッドは次の通りです。

|    | 名前                        | 説明   |
|----|---------------------------|--|
| 1  | public void set_id()      | cjidプロパティに ID を設定します。  |
| 2  | public void init_set()    | cjid, objspec, objtype を設定します。   |
| 3  | public void add_attr()    | DshCjAttr クラスに保存されている属性情報を attr_list に追加します。                                 |
| 4  | public int alloc_id()     | コントロールジョブ ID をエンジンに登録します。  |
| 5  | public int set()          | インスタンス内に設定されたコントロールジョブ情報をエンジンに設定します。   |
| 6  | public int get()          | エンジンからコントロールジョブ情報を取得します。   |
| 7  | public void delete()      | cjidに指定されたコントロールジョブ情報をエンジンの管理情報から削除します。                                      |
| 8  | public int decode()       | S14F9 メッセージを当該クラスにデコードします。   |
| 9  | public void copy()        | DshCj クラスの当該インスタンスの内容を別のインスタンスにコピーします。                                       |
| 10 | public int get_id_count() | エンジンに登録されているコントロールジョブ ID 情報の合計数を取得します。                                       |
| 11 | public int get_id_list()  | エンジンに登録されている全コントロールジョブ ID をリストに取得します。  |
| 12 | public void clear()       | attr_list 配列の内容を消去します。<br>attr_count=0 にします。                                 |
| 13 | public void Dispose()     | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。 |

### 14. 1. 3. 1 set\_id()

コントロールジョブ ID を設定します。

コントロールジョブ ID は、エンジンに登録し管理するための ID です。S14F9 などメッセージ処理に使用されます。

#### 【構文】

```
public void set_id( string cjid )
```

#### 【引数】

cjid

設定するコントロールジョブ ID です。

#### 【戻り値】

なし。

#### 【説明】

引数に与えられた cjid の値をインスタンス内の cjid に設定します。

### 14. 1. 3. 2 init\_set()

インスタンスにコントロールジョブの3つのプロパティ値を設定します。

#### 【構文】

```
public void init_set(string cjid, string objspec, string objtype)
```

#### 【引数】

cjid

コントロールジョブ ID です。

objspec

オブジェクト仕様文字列です。

objtype

オブジェクトのグループまたはクラスの ID です。

#### 【戻り値】

なし。

#### 【説明】

引数に与えられた値を、それぞれインスタンス内のプロパティの値として設定します。

### 14. 1. 3. 3 add\_attr()

インスタンスの attr\_list 配列に属性情報を 1 個追加します。

**【構文】**

```
public void add_attr(ref DshCjAttr attr)
```

**【引数】**

attr

属性情報が保存されている DshCjAttr クラスのインスタンスです。

**【戻り値】**

なし。

**【説明】**

attr\_list 配列に 1 個の属性情報を追加します( attr の内容を attr\_list[attr\_count] にコピーします。)その後、attr\_count + 1 します。

### 14. 1. 3. 4 alloc\_id()

コントロールジョブ ID をエンジンに新規登録します。

**【構文】**

```
public int alloc_id()
public int alloc_id(string cjid)
```

**【引数】**

cjid

登録するコントロールジョブ ID です。

**【戻り値】**

| 返却値    | 意味                               |
|--------|----------------------------------|
| 0 or 1 | 正常に登録できた。(1 は既に登録済みであったことを意味します) |
| (-1)   | 登録に失敗した。                         |

**【説明】**

引数として cjid が指定された場合は、そのコントロールジョブ ID をエンジンに登録します。

登録できた場合は、引数の cjid がインスタンス内の cjid に設定されます。

引数がない場合は、インスタンス内の cjid のコントロールジョブ ID をエンジンに登録します。

正常に登録できた場合は、0 または 1 を返却します。失敗した場合は、(-1) を返却します。

既に ID が登録されていた場合、エンジンは管理情報の中から ID 以外の情報をクリアします。



### 14. 1. 3. 5 set()

コントロールジョブ情報をエンジンに設定します。

**【構文】**

```
public int set()
```

**【引数】**

なし。

**【戻り値】**

| 返却値  | 意 味       |
|------|-----------|
| 0    | 正常に設定できた。 |
| (-1) | 設定に失敗した。  |

**【説明】**

インスタンス内に設定されたコントロールジョブ情報をエンジンの管理情報に設定します。

### 14. 1. 3. 6. get()

エンジンからコントロールジョブ情報を取得します。

**【構文】**

```
public int get()
public int get( string cjid)
```

**【引数】**

cjid  
情報を取得したいコントロールジョブ ID です。

**【戻り値】**

| 返却値  | 意 味       |
|------|-----------|
| 0    | 正常に取得できた。 |
| (-1) | 取得に失敗した。  |

**【説明】**

エンジンの管理情報からコントロールジョブ情報を取得します。

引数で cjid が指定された場合は、そのコントロールジョブの情報を取得します。

引数がない場合は、インスタンス内の cjid のコントロールジョブの情報を取得します。

正常に取得できた場合は、0 を、失敗した場合は(-1)を返却します。

### 14. 1. 3. 7 delete()

指定されたコントロールジョブの情報をエンジン管理情報の中から削除します。

**【構文】**

```
public int delete()
public int delete(string cjid)
```

**【引数】**

cjid  
削除したいコントロールジョブの ID です。

**【戻り値】**

| 返却値  | 意味                     |
|------|------------------------|
| 0    | 正常に削除できた。              |
| 1    | エンジンに登録されていなかった。       |
| (-1) | 削除できなかった。(cjidが無効であった) |

**【説明】**

1 個のコントロールジョブ情報をエンジンの管理情報から削除します。

削除対象は、eqid で指定された装置の情報であり、引数にコントロールジョブ ID の指定があるメソッドでは、引数で指定されたコントロールジョブを、そして、引数が無いメソッドの場合は、インスタンスの cjid プロパティのコントロールジョブを削除します。

一旦削除された情報は、他のメソッドでは復元できませんので注意してください。

### 14. 1. 3. 8 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshCj dst )
```

**【引数】**

dst  
コピー先の DshCj インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容 (プロパティ値) を dst に指定されるインスタンスにコピーします。

### 14. 1. 3. 9 decode()

S14F9に含まれる情報をDshCjクラス内のプロパティにデコードします。

#### 【構文】

```
public int decode(ref DSHMSG msg)
```

#### 【引数】

msg

S14F9のメッセージ情報(生情報)が格納されているDSHMSG構造体領域になります。DSHMSGは、1次メッセージをポーリングした際にエンジンから与えられる情報です。ユーザはDSHMSG構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

#### 【戻り値】

| 返却値  | 意味                                  |
|------|-------------------------------------|
| 0    | 正常にデコードできた。                         |
| (-1) | デコードできなかった (S14F9のメッセージの形式が正しくなかった) |

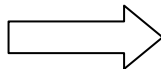
#### 【説明】

msgに含まれているコントロールジョブ情報をDshCjクラス内にデコードします。

msg S14F9

```
L, 3
  objspec
  objtype
L, a
  L, 2
    attridl
    attrdatal
  L, 2
  .
  .
```

decode



```
DshCj
Class
  ejid,
  objspec
  objtype
  attr_list[]
  .
  .
```

メッセージに含まれるコントロールジョブ情報(ejid, 属性情報等)は、クラスのプロパティに保存されます。

正常にデコードできた場合、0を返却します。

もし、S14F9のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1)を返却します。

#### 14. 1. 3. 10 get\_id\_count()

エンジンの当該装置に登録されているコントロールジョブ ID の合計数を取得します。

**【構文】**

```
public int get_id_count()
```

**【引数】**

なし。

**【戻り値】**

登録されているコントロールジョブ ID 数が返却されます。

**【説明】**

登録されているコントロールジョブ ID の合計数を取得します。

#### 14. 1. 3. 11 get\_id\_list()

エンジンの当該装置に登録されている全コントロールジョブの ID と名前のリストを取得します。

**【構文】**

```
public int get_id_list(string[] id_list, string[] n, name_list, int list_size)
```

**【引数】**

id\_list

コントロールジョブ ID を格納する配列です。

n, name\_list

コントロールジョブ名を格納する配列 (name は id と同じになります) です。

list\_size

準備されたリストの配列サイズです。

**【戻り値】**

取得できたコントロールジョブ ID の数が返却されます。

**【説明】**

エンジンに登録されている全コントロールジョブ ID とその名前をそれぞれ id\_list[], name\_list[] に取得します。

list\_size は配列のサイズを指定します。

配列は、登録されている全コントロールジョブ ID を保存できる充分のサイズの配列を準備してください。

返却値は取得できたコントロールジョブ ID の合計数です。

(注) ID と名前格納用リストのサイズは、先に説明した get\_id\_count() メソッドで合計 ID 数を取得し、そのサイズの配列を準備して本メソッドを使用すると便利です。

#### 14. 1. 3. 12 clear()

インスタンス内の attr\_list 配列に含まれる属性情報を全て消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし。

**【戻り値】**

なし。

**【説明】**

attr\_list 配列の中の attr\_count 分の情報を消去し、配列を空にします。そして attr\_count=0 にします。

具体的には、各配列の DshCjAttr クラスのインスタンスの消去です。それらの中に使用されている非管理メモリを開放します。

#### 14. 1. 3. 13 Dispose()

本クラスが使用済になった際、クラス内で使用していて、開放すべきメモリがあれば、それを開放し、また Dispose すべきオブジェクトがあれば、それらを Dispose します。

**【構文】**

```
public void Dispose()
```

**【引数】**

なし。

**【戻り値】**

なし。

**【説明】**

生成された後、当該クラスが使用済みになった時点で、ユーザプログラムが明示的に使用していた資源を解放するためのメソッドです。

本メソッドが実行されるとシステムから本クラスに対する Finalizer は呼び出されません。

## 14. 2 DshCjCmd クラス

コントロールジョブコマンド情報を保存するためのクラスです。本クラスはDshObjPara の派生クラスです。

本クラスは次のメッセージの送信または受信処理のために使用されます。

S16F27 メッセージ

### 14. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明  |
|---|--|---|
| 1 | public DshCjCmd()  | 空のインスタスを生成します。<br>プロパティ値は set_cmd_info() メソッドを使って設定します。                                     |
|   | public DshCjCmd(string cjid,<br>int cmd)   | プロセッサ ID とコマンドを指定してインスタスを生成します。<br>他のプロパティ値は set_cmd_info() メソッドを使って設定します。                  |
|   | public DshCjCmd(string cjid,<br>int cmd,<br>string cpname,<br>int fmt,<br>int size,<br>IntPtr cpval) | cjid, cmd, パラメータ名、フォーマット、値格納ポインタを指定して<br>インスタスを生成します。<br>全プロパティを指定します。                      |
|   | public DshCjCmd(string cjid,<br>int cmd,<br>string cpname,<br>string cpval)                          | cjid, cmd, パラメータ名、文字列タイプのパラメータ値を指定して<br>インスタスを生成します。<br>全プロパティを指定します。(HSMS. ICODE_A フォーマット) |

## 14. 2. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                   | 説明  |
|---|----------------------|---|
| 1 | public string cjid   | コントロールジョブ ID です。  |
| 2 | public int cmd       | コマンド文字列です。  |
| 3 | public string name   | DshObjPara の name メンバーです。<br>パラメータ名です。                            |
| 4 | public int format    | DshObjPara の format メンバーです。<br>パラメータ値のフォーマットです。(HSMS. ICODE_A など) |
| 5 | public int size      | DshObjPara の size メンバーです。<br>パラメータ値の配列サイズです。(HSMS. ICODE_A 以外は=1) |
| 6 | public IntPtr value  | DshObjPara の value メンバーです。<br>パラメータ値が保存されているポインタです。               |
| 7 | public int para_flag | パラメータの設定状態のフラグです。<br>ユーザの処理には関係ありません。                             |

## 14. 2. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                         | 説明   |
|---|----------------------------|--|
| 1 | public void set_cmd_info() | cjid、cmd とコマンドパラメータを設定します。   |
| 2 | public int decode()        | S16F27 メッセージを当該クラスにデコードします。  |
| 3 | public void free()         | DshObjPara クラスの value に使用した非管理メモリを解放します。<br>(DshObjPara のメソッドです。そちらを参照ください。)                                   |
| 4 | public void Dispose()      | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

### 14. 2. 3. 1 set\_cmd\_info()

コントロールジョブ ID とコマンドをインスタンス内に設定します。

#### 【構文】

```
public void set_cmd_info( string cjid, int cmd)
public void set_cmd_info(string cjid, int cmd,
                        string cpname, int format, int size, IntPtr cpval)
public void set_cmd_info(string cjid, int cmd, string cpname, string cpval)
```

#### 【引数】

cjid  
コントロールジョブ ID です。

cmd  
コントロールジョブに与えるコマンドです。

cpname  
パラメータ名です。

format  
パラメータ値のフォーマットです。(HSMS. ICODE\_U1 など)

size  
パラメータ値の配列サイズです。HSMS. ICODE\_A のとき、文字列長になります。  
その他のフォーマットでは =1 になります。

cpval  
パラメータ値が格納されているポインタです。  
string の場合は文字列です。

#### 【戻り値】

なし。

#### 【説明】

コントロールジョブ ID、cjid とコマンド、cmd をそれぞれのプロパティに設定します。



また、パラメータの引数が指定されているものについては、それぞれ、DshObjPara クラスのインスタンスに設定します。

## 14. 2. 3. 2 decode()

S16F27 メッセージに含まれる情報を DshCjCmd クラス内のプロパティにデコードします。  
本メソッドは S16F27 メッセージ受信処理時に使用することができます。

### 【構文】

```
public int decode(ref DSHMSG smsg)
```

### 【引数】

smsg

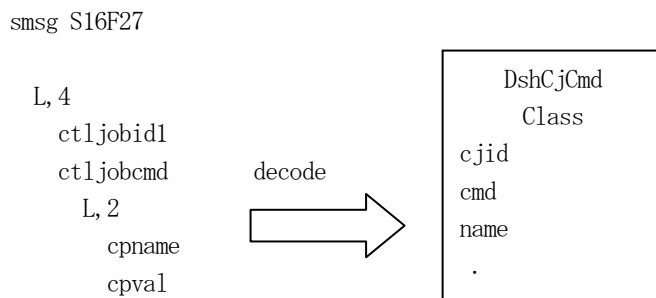
S16F27 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。  
DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。  
ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

### 【戻り値】

| 返却値  | 意味                                    |
|------|---------------------------------------|
| 0    | 正常にデコードできた。                           |
| (-1) | デコードできなかった (S16F27 のメッセージの形式が正しくなかった) |

### 【説明】

smsg に含まれている情報を DshCjCmd クラス内にデコードします。



### 14. 3 DshCjAttr クラス

S14F9 コントロールジョブ送信メッセージに含まれる 1 個の属性情報を保存するためのクラスです。

#### 14. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                               | 説明   |
|---|----------------------------------|--|
| 1 | public DshCjAttr()               | 空のインスタスを生成します。                                       |
| 2 | public DshCjAttr(string attrid)  | 属性 ID を指定してインスタスを生成します。<br>(属性インデックスの値を変換して設定します。)   |
| 3 | public DshCjAttr(int attr_index) | 属性 ID インデックスを指定してインスタスを生成します。<br>attrid の値も取得し設定します。 |

### 14. 3. 2 プロパティ

プロパティ一覧表に示します。

|    | 名前   | 説明  |
|----|--|---|
| 1  | public string attrid                             | 属性 ID を保存します。(下の表参照)  |
| 2  | public int attr_index                            | 属性 ID をインデックスに変換したものです。<br>dsh_const クラスに定義されています。              |
| 3  | public int num_data                              | 整数の属性値を保存します。   |
| 4  | public string str_data                           | 文字列の属性値を保存します。  |
| 5  | public int status_count                          | DshMaterialOutStatus が保存している属性値の数です。                            |
| 6  | public DshMaterialOutStatus[]<br>mtrl_out_status | DshMaterialOutStatus クラスの配列です。<br>MtrlOutByStatus 属性の情報を保存します。  |
| 7  | public int spec_count                            | DshMaterialOutSpec が保存している属性値の数です。                              |
| 8  | public DshMaterialOutSpec[]<br>mtrl_out_spec     | DshMaterialOutSpec クラスの配列です。<br>MtrlOutSpec 属性の情報を保存します。        |
| 9  | public int proc_count                            | DshProcControlSpec が保存している属性値の数です。                              |
| 10 | public DshProcControlSpec[]<br>proc_control_spec | DshProcControlSpec クラスの配列です。<br>ProcessingCtrlSpec 属性の情報を保存します。 |
| 11 | public DshPrjStateList<br>prj_state_list         | PRJobStatusList 属性の情報を保存します。                                    |
| 12 | public DshPauseEvent pause_event                 | PauseEvent 属性の情報を保存します。   |
| 13 | public DshTextList text_list                     | テキスト配列の属性の情報を保存します。   |

コントロールジョブ属性インデクス値 / 属性 ID

| index 値 | 定数名 - attr_index       | 属性 ID - attrid        |
|---------|------------------------|-----------------------|
| 0       | EN_ObjID               | “ObjID”               |
| 1       | EN_CarrierInputSpec    | “CarrierInputSpec”    |
| 2       | EN_CurrentPRJob        | “CurrentPRJob”        |
| 3       | EN_DataCollectionPlan  | “DataCollectionPlan”  |
| 4       | EN_MtrlOutByStatus     | “MtrlOutByStatus”     |
| 5       | EN_MtrlOutSpec         | “MtrlOutSpec”         |
| 6       | EN_PauseEvent          | “PauseEvent”          |
| 7       | EN_ProcessingCtrlSpec  | “ProcessingCtrlSpec”  |
| 8       | EN_ProcessingOrderMgmt | “ProcessingOrderMgmt” |
| 9       | EN_PRJobStatusList     | “PRJobStatusList”     |
| 10      | EN_StartMethod         | “StartMethod”         |
| 11      | EN_State               | “State”               |

### 14. 3. 3 メソッド

本クラスのメソッドは次の通りです。

|    | 名前  | 説明  |
|----|---|---|
| 1  | public void clear()                             | mtrl_out_status, mtrl_out_spec, proc_control_spec, prj_state_list, pause_event, text_list の内容を消去し、空にします。そしてそれぞれの配列の count を 0 にします。 |
| 2  | public void set_string()                        | 文字列情報を設定します。  |
| 3  | public void set_num()                           | 整数情報を設定します。   |
| 4  | public void add_MaterialOutStatus()             | MtrlOutByStatus 属性の情報を追加します。mtrl_out_status 配列に追加します。   |
| 5  | public void add_slot()                          | DshMaterialOutStatus の car_slot メンバーに slotid を 1 個追加します。  |
| 6  | public void add_MaterialOutSpec()               | MtrlOutSpec 属性の情報を追加します。mtrl_out_spec 配列に追加します。   |
| 7  | public void add_src_slot()                      | DshMaterialOutSpec の src_car_slot メンバーに slotid を 1 個追加します。  |
| 8  | public void add_dst_slot()                      | DshMaterialOutSpec の dst_car_slot メンバーに slotid を 1 個追加します。  |
| 9  | public void add_prj_proc_count()                | ProcessingCtrlSpec 属性の情報として proc_control_spec 配列に prjid を指定して DshProcControlSpec インスタンスを 1 個追加します。                                  |
| 10 | public int init_prj_proc_count()                | ProcessingCtrlSpec 属性の情報として proc_control_spec 配列の指定位置に prjid を指定して DshProcControlSpec インスタンスを 1 個生成します。                             |
| 11 | public void init_PrjStateList()                 | prj_state_list に DshPrjStateList クラスのインスタンスを生成します。  |
| 12 | public void init_PauseEvent()                   | pause_event に DshPauseEvent クラスのインスタンスを生成します。   |
| 13 | public void init_Text()                         | text_list に DshTextList クラスのインスタンスを生成します。   |
| 14 | public int set_string_attr()                    | attr_index から属性 ID を求め、attrid に設定した上で、文字列データを str_data に設定します。  |
| 15 | public int set_num_attr()                       | attr_index から属性 ID を求め、attrid に設定した上で、文字列データを str_data に設定します。  |
| 16 | public static string<br>get_attrid_from_index() | attr_index から属性 ID を求めます。   |
| 17 | public void copy()                              | 当該インスタンスの内容を別のインスタンスにコピーします。  |
| 18 | public void Dispose()                           | クラス内部で使用された資源(メモリ)をシステムに返却します。他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。                          |

### 14. 3. 3. 1 clear()

クラス内のプロパティの内容を消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

mtrl\_out\_status, mtrl\_out\_spec, proc\_control\_spec, prj\_state\_list, pause\_event, text\_list の内容を消去し、空にします。

そしてそれぞれの配列の count を 0 にします。

### 14. 3. 3. 2 set\_string()

文字列データを設定します。

**【構文】**

```
public void set_string(string data)
```

**【引数】**

data

文字列データです。

**【戻り値】**

なし。

**【説明】**

data に与えられた文字列を str\_data プロパティに設定します。

### 14. 3. 3. 3 set\_num()

整数データを設定します。

**【構文】**

```
public void set_num(int data)
```

**【引数】**

data  
整数データです。

**【戻り値】**

なし。

**【説明】**

data に与えられた整数を num\_data プロパティに設定します。

### 14. 3. 3. 4 add\_MaterialOutStatus()

MtrlOutByStatus 属性のクラス DshMaterialOutStatus の配列 mtrl\_out\_status に 1 個の属性情報を追加します。  
DshMaterialOutStatus クラスについては後述します。

**【構文】**

```
public void add_MaterialOutStatus(string carid, int status)
```

**【引数】**

carid  
キャリア ID です。  
status  
キャリアの状態です。

**【戻り値】**

なし。

**【説明】**

mtrl\_out\_status 配列の status\_count 位置に DshMaterialOutStatus クラスのインスタンスを生成し、そこに carid, status を設定します。その後、status\_count + 1 します。

### 14. 3. 3. 5 add\_slot()

mtrl\_out\_status の指定配列位置にスロット ID を追加します。

#### 【構文】

```
public int add_slot(int index, int slotid)
```

#### 【引数】

index

配列位置です。

slotid

設定するスロット ID です。

#### 【戻り値】

| 返却値  | 意味  |
|------|---|
| 0    | 正常に追加できた。                                       |
| (-1) | 追加できなかった。(index >=status_count であった。インスタンスが未生成) |

#### 【説明】

mtrl\_out\_status の指定配列位置のインスタンスの car\_slot メンバーにスロット ID を 1 個追加します。car\_slot については、後述する DshCjCarSlotList クラスの説明を参照してください。

### 14. 3. 3. 6 add\_MaterialOutSpec()

MtrlOutSpec 属性のクラス DshMaterialOutSpec の配列 mtrl\_out\_spec に 1 個の属性情報を追加します。DshMaterialOutSpec クラスについては後述します。

#### 【構文】

```
public void add_MaterialOutSpec(string src_carid, string dst_carid)
```

#### 【引数】

src\_carid

ソースのキャリア ID です。

dst\_carid

行き先のキャリア ID です。

#### 【戻り値】

なし。

#### 【説明】

mtrl\_out\_spec 配列の spec\_count 位置に DshMaterialOutSpec クラスのインスタンスを生成し、そこに src\_carid, dst\_carid を設定します。その後、spec\_count + 1 します。



### 14. 3. 3. 7 add\_src\_slot()

mtrl\_out\_spec の指定配列位置インスタンスの src\_car\_slot にスロット ID を追加します。

**【構文】**

```
public int add_src_slot(int index, int slotid)
```

**【引数】**

index

配列位置です。

slotid

設定するスロット ID です。

**【戻り値】**

| 返却値  | 意味  |
|------|---|
| 0    | 正常に追加できた。                                     |
| (-1) | 追加できなかった。(index >=spec_count であった。インスタンスが未生成) |

**【説明】**

mtrl\_out\_spec の指定配列位置のインスタンスの src\_car\_slot メンバーにスロット ID を1個追加します。

src\_car\_slot については、後述する DshMaterialOutSpec と DshCjCarSlotList クラスの説明を参照してください。

### 14. 3. 3. 8 add\_dst\_slot()

mtrl\_out\_spec の指定配列位置インスタンスの dst\_car\_slot にスロット ID を追加します。

**【構文】**

```
public int add_dst_slot(int index, int slotid)
```

**【引数】**

index

配列位置です。

slotid

設定するスロット ID です。

**【戻り値】**

| 返却値  | 意味  |
|------|---|
| 0    | 正常に追加できた。                                     |
| (-1) | 追加できなかった。(index >=spec_count であった。インスタンスが未生成) |

**【説明】**

mtrl\_out\_spec の指定配列位置のインスタンスの dst\_car\_slot メンバーにスロット ID を1個追加します。

dst\_car\_slot については、後述する DshMaterialOutSpec と DshCjCarSlotList クラスの説明を参照してください。

### 14. 3. 3. 9 add\_prj\_proc()

ProcessingCtrlSpec 属性のクラス DshProcControlSpec の配列 proc\_control\_spec に 1 個の属性情報を追加します。DshProcControlSpec クラスについては後述します。

**【構文】**

```
public void add_prj_proc( string prjid)
```

**【引数】**

prjid  
プロセスジョブ ID です。

**【戻り値】**

なし。

**【説明】**

proc\_control\_spec 配列の proc\_count 位置に DshProcControlSpec クラスのインスタンスを生成し prjobid プロパティに prjid を設定します。その後、proc\_count + 1 します。

### 14. 3. 3. 10 init\_prj\_proc()

proc\_control\_spec の指定配列位置インスタンスにプロセスジョブ ID を設定します。

**【構文】**

```
public int init_prj_proc( int index, string prjid)
```

**【引数】**

index  
配列位置です。  
prjid  
設定するプロセスジョブ ID です。

**【戻り値】**

| 返却値  | 意味  |
|------|---|
| 0    | 正常に追加できた。                                     |
| (-1) | 追加できなかった。(index >=proc_count であった。インスタンスが未生成) |

**【説明】**

proc\_control\_spec の指定配列位置のインスタンスの prjobid プロパティに 引数で与えられた prjid を設定します。

後述する DshProcControlSpec クラスの説明を参照してください。

### 14. 3. 3. 11 init\_PrjStateList()

PRJobStatusList 属性のクラス DshPrjStateList のインスタンスをに prj\_state\_list に生成します。  
DshPrjStateList クラスについては後述します。

**【構文】**

```
public void init_PrjStateList()
```

**【引数】**

なし。

**【戻り値】**

なし。

**【説明】**

prj\_state\_list に DshPrjStateList クラスのインスタンスを生成します。

### 14. 3. 3. 12 init\_PauseEvent()

PauseEvent 属性のクラス DshPauseEven のインスタンスをに pause\_event に生成します。  
DshPauseEven クラスについては後述します。

**【構文】**

```
public void init_PauseEvent()
```

**【引数】**

なし。

**【戻り値】**

なし。

**【説明】**

pause\_event に DshPauseEven クラスのインスタンスを生成します。

### 14. 3. 3. 13 init\_Text()

複数のテキスト（文字列）を保存する属性のクラス DshTextList のインスタンスをに text\_list に生成します。  
DshTextList クラスについては後述します。

**【構文】**

```
public void init_Text()
```

**【引数】**

なし。

**【戻り値】**

なし。

**【説明】**

text\_list に DshTextList クラスのインスタンスを生成します。

### 14. 3. 3. 14 set\_string\_attr()

文字列属性に対して文字列を設定します。

**【構文】**

```
public void set_string_attr(int attr_index, string value)  
public void set_string_attr(string value)
```

**【引数】**

attr\_index  
属性 ID のインデクスです。  
value  
文字列データです。

**【戻り値】**

なし。

**【説明】**

str\_data に value の文字列データを設定します。  
attr\_index の引数が指定された場合は、attr\_index と attrid も設定します。

### 14. 3. 3. 15 set\_num\_attr()

数値属性に対して整数値を設定します。

#### 【構文】

```
public void set_num_attr(int attr_index, int value)
public void set_num_attr( int value)
```

#### 【引数】

attr\_index  
属性 ID のインデクスです。

value  
整数データです。

#### 【戻り値】

なし。

#### 【説明】

num\_data に value の数値データを設定します。  
attr\_index の引数が指定された場合は、attr\_index と attrid も設定します。

### 14. 3. 3. 16 get\_attrid\_from\_index()

属性インデクスから属性 ID (文字列) を取得します。

#### 【構文】

```
public static string get_attrid_from_index(int index)
```

#### 【引数】

attr\_index  
属性 ID のインデクスです。

#### 【戻り値】

属性 ID が返却されます。  
もし、attr\_index が未定義の場合は空の文字列が返却されます。

#### 【説明】

属性インデクスから属性 ID (文字列) を取得します。

### 14. 3. 3. 17 copy()

当該インスタンスの内容を別の DshCjAttr クラスのインスタンスにコピーします。

**【構文】**

```
public void copy(ref DshCjAttr dst)
```

**【引数】**

dst

コピー先インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンス内に保存されているプロパティ情報を dst で指定されたインスタンスにコピーします。

## 14. 4 DshCjAttrList クラス

複数のコントロールジョブの属性を保存するための配列クラスです。

### 14. 4. 1 コンストラクタ

|   | 名前                     | 説明             |
|---|------------------------|----------------|
| 1 | public DshCjAttrList() | 空のインスタスを生成します。 |

### 14. 4. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                      | 説明                                 |
|---|-------------------------|------------------------------------|
| 1 | public int count        | list に保存されている属性数です。                |
| 2 | public DshCjAttr[] list | 属性情報保存配列です。<br>DshCjAttr クラスの配列です。 |

### 14. 4. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void clear()    | value に使用されているメモリを開放します。   |
| 2 | public void add_attr() | パラメータ情報を設定します。   |
| 3 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 4. 3. 1 clear()

list に保存されている属性情報を消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

list に保存されている DshCjAttr 属性情報を消去し、配列を空にします。  
そして、count=0 にします。

#### 14. 4. 3. 2 add\_attr()

list 配列に属性情報を 1 個追加します。

**【構文】**

```
public void add_attr(ref DshCjAttr attr)
```

**【引数】**

attr

追加したい DshCjAttr クラスのインスタンスです。

**【戻り値】**

なし。

**【説明】**

list 配列の count 位置に DshCjAttr のインスタンスを生成し、引数 attr のインスタンスの内容をコピーします。  
そして、count+1 します。



## 14. 5 DshStatusPara クラス

1個のステータスパラメータ値を保存するためのクラスです。

ステータスパラメータのデータタイプには、HSMS クラスで定義されている SECS-II メッセージに使用される様々なものがありますが、それらを非管理メモリ領域に保存します。

### 14. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前  | 説明   |
|---|---|--|
| 1 | <code>public DshStatusPara()</code>   | 空のインスタンスを生成します。<br>値はヌルで設定します。                           |
| 2 | <code>public DshStatusPara(int status,<br/>int format,<br/>int size,<br/>IntPtr value)</code> | 引数として status, フォーマット、size, 値が格納されているポインタ(IntPtr)が指定されます。 |

### 14. 5. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                               | 説明  |
|---|----------------------------------|---|
| 1 | <code>public int status</code>   | ステータスパラメータ ID です。                                   |
| 2 | <code>public int format</code>   | ステータスパラメータのフォーマットです。<br>(HSMS. ICODE_U1 など)         |
| 3 | <code>public int size</code>     | ステータスパラメータの配列サイズ (データ数、基本的に ICODE_A 以外は size=1 です)。 |
| 4 | <code>public IntPtr value</code> | ステータスパラメータ値を格納する非管理メモリのポインタです。                      |

### 14. 5. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void free()     | value に使用されている非管理メモリを開放します。  |
| 2 | public void set_para() | ステータスステータスパラメータ値を設定します。  |
| 3 | public void copy()     | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 4 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 5. 3. 1 free()

インスタンスの value に使用されている非管理メモリを開放します。

**【構文】**

```
public void free()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

ステータスパラメータ値 value に使用されている非管理メモリを開放します。

### 14. 5. 3. 2 set\_para()

ステータスパラメータを設定します。

#### 【構文】

```
public void set_para(int status, int format, int size, IntPtr value)
```

#### 【引数】

status

状態値です。

format

ステータスパラメータのデータフォーマットです。(HSMS. ICODE\_U1 など)

size

ステータスパラメータ値の配列サイズ

value

設定したいステータスパラメータ値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

各プロパティ値に引数の値を設定します。

value については別の IntPtr メモリを確保します。

### 14. 5. 3. 3 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

#### 【構文】

```
public void copy( ref DshStatusPara dst )
```

#### 【引数】

dst

コピー先の DshStatusPara インスタンスです。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンスの内容 (プロパティ値) を dst に指定されるインスタンスにコピーします。

## 14. 6 DshStatusParaList クラス

複数個のステータスパラメータを保存するためのDshStatusPara クラスの配列です。

### 14. 6. 1 コンストラクタ

|   | 名前                          | 説明                                 |
|---|-----------------------------|------------------------------------|
| 1 | public DshStatusParaList () | 空のインスタンスを生成します。<br>配列サイズは=0 になります。 |

### 14. 6. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                                  | 説明                                  |
|---|-------------------------------------|-------------------------------------|
| 1 | public int count                    | 設定されているステータスパラメータの数です。              |
| 2 | public DshStatusPara[]<br>para_list | ステータスパラメータ保存クラスDshStatusPara の配列です。 |

### 14. 6. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void clear()    | para_list 配列を空にします。  |
| 2 | public void add_para() | para_list 配列に 1 個のステータスパラメータを加えます。   |
| 3 | public void copy()     | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 4 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 6. 3. 1 clear()

インスタンスの para\_list 配列の中に含まれるステータスパラメータの value に使用されている全非管理メモリを開放し、para\_list リストを空にします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

para\_list 配列のなかのステータスパラメータ値 value に使用されている非管理メモリを開放し、リストを空にします。そして、count = 0 にします。

### 14. 6. 3. 2 add\_para()

インスタンスの para\_list 配列にステータスパラメータ情報を 1 個追加します。

#### 【構文】

```
public void add_para(DshStatusPara para)
public void add_para(int status, int format, int size, IntPtr value)
```

#### 【引数】

para  
DshStatusPara クラスのインスタンス

status  
追加するステータスパラメータ ID です。

format  
ステータスパラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size  
ステータスパラメータ値の配列サイズ

value  
設定したいステータスパラメータ値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

para\_list 配列に 1 個のステータスパラメータ情報を追加します。  
引数が、DshStatusPara クラスのインスタンスの場合は、それをコピーして設定します。  
2 番目のオーバーロードメソッドでは、DshStatusPara クラスの set\_para() メソッドを使ってインスタンスを生成し追加します。追加後、count + 1 します。

### 14. 6. 3. 3 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

#### 【構文】

```
public void copy( ref DshStatusParaList dst )
```

#### 【引数】

dst  
コピー先の DshStatusParaList インスタンスです。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンスの内容 (プロパティ値) を dst に指定されるインスタンスにコピーします。

## 14. 7 DshMaterialOutStatus クラス

MtrlOutByStatus 属性情報を保存するためのクラスです。

### 14. 7. 1 コンストラクタ

|   | 名前                            | 説明             |
|---|-------------------------------|----------------|
| 1 | public DshMaterialOutStatus() | 空のインスタスを生成します。 |

### 14. 7. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                                  | 説明  |
|---|-------------------------------------|---|
| 1 | public int status                   | ステータスです。(material out status)   |
| 2 | public int slot_count               | 設定されているスロット ID の数です。  |
| 3 | public DshCjCarSlotList<br>car_slot | キャリア ID とスロット ID を保存する DshCjCarSlotList クラスです。<br>(14.9 参照)<br>情報設定は DshCjCarSlotList クラスのメソッドを使用します。 |

### 14. 7. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                       | 説明  |
|---|--------------------------|---|
| 1 | public void set_status() | ステータスを設定します。  |
| 2 | public void copy()       | 当該インスタスの内容を別のインスタスにコピーします。  |
| 3 | public void Dispose()    | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14.1.3.13 と同様です。そちらを参照ください。 |

#### 14. 7. 3. 1 set\_status()

ステータス(material out status)を設定します。

**【構文】**

```
public void set_status(int mtrl_status)
```

**【引数】**

mtrl\_status  
ステータス値です。

**【戻り値】**

なし。

**【説明】**

ステータスを設定します。

#### 14. 7. 3. 2 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshMaterialOutStatus dst )
```

**【引数】**

dst  
コピー先のDshMaterialOutStatus インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。



## 14. 8 DshMaterialOutSpec クラス

MtrlOutSpec 属性情報を保存するためのクラスです。

### 14. 8. 1 コンストラクタ

|   | 名前                           | 説明              |
|---|------------------------------|-----------------|
| 1 | public DshMaterialOutSpec () | 空のインスタンスを生成します。 |

### 14. 8. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                                      | 説明  |
|---|---|---|
| 1 | public DshCjCarSlotList<br>src_car_slot | ソースのキャリア ID とスロット ID を保存する DshCjCarSlotList クラスです。(14.9 参照)<br>情報設定は DshCjCarSlotList クラスのメソッドを使用します。 |
| 2 | public DshCjCarSlotList<br>dst_car_slot | 行き先キャリア ID とスロット ID を保存する DshCjCarSlotList クラスです。(14.9 参照)<br>情報設定は DshCjCarSlotList クラスのメソッドを使用します。  |

### 14. 8. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                    | 説明   |
|---|-----------------------|--|
| 1 | public void copy()    | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 2 | public void Dispose() | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 8. 3. 1 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshMaterialOutSpec dst )
```

**【引数】**

dst

コピー先の DshMaterialOutSpec インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容 (プロパティ値) を dst に指定されるインスタンスにコピーします。

## 14. 9 DshCjCarSlotList クラス

キャリアのロット ID を配列で保存するためのクラスです。

### 14. 9. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明                                |
|---|--|-----------------------------------|
| 1 | <code>public DshCjCarSlotList ()</code>                  | 空のインスタスを生成します。<br>配列サイズは=0 になります。 |
| 2 | <code>public DshCjCarSlotList (<br/>string carid)</code> | キャリア ID を指定してインスタスを生成します。         |

### 14. 9. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                                  | 説明                  |
|---|-------------------------------------|---------------------|
| 1 | <code>public string carid</code>    | キャリア ID です。         |
| 2 | <code>public int slot_count</code>  | 設定されているロット ID の数です。 |
| 3 | <code>public int[] slot_list</code> | ロット ID を保存する配列です。   |

### 14. 9. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                        | 説明   |
|---|---------------------------|--|
| 1 | public void clear()       | slot_list 配列を空にします。  |
| 2 | public void set_carid()   | キャラ ID を設定します。<br>(slot_list を空にします。)  |
| 3 | public void add_slotid()  | slot_list 配列に 1 個のロット ID を加えます。  |
| 4 | public void copy_slotid() | 配列に保存されているロット ID を当該インスタンスにコピーします。   |
| 5 | public void copy()        | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 6 | public void Dispose()     | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 9. 3. 1 clear()

インスタンスの slot\_list 配列リストを空にします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

slot\_list 配列リストを空にします。そして、count = 0 にします。

### 14. 9. 3. 2 set\_carid()

キャリア ID を設定し、slot\_list を空にします。

**【構文】**

```
public void set_carid( string carid )
```

**【引数】**

carid

設定したいキャリア ID です。

**【戻り値】**

なし。

**【説明】**

キャリア ID を設定します。そして、スロット ID リストを空にします。

### 14. 9. 3. 3 add\_slotid()

インスタンスの slot\_list 配列にスロット ID を 1 個追加します。

**【構文】**

```
public void add_slotid(int id)
```

**【引数】**

id

追加するスロット ID です。

**【戻り値】**

なし。

**【説明】**

slot\_list 配列に 1 個のスロット ID を追加します。  
追加後、count + 1 します。

#### 14. 9. 3. 4 copy\_slotid()

複数個のスロット ID を当該インスタンスの slot\_list にコピーします。

**【構文】**

```
public void copy_slotid(IntPtr srcptr, int count)
```

**【引数】**

srcptr

複数のスロット ID が保存されているコピー元メモリのポインタです。

count

コピーしたいスロット ID 数です。(srcptr に保存されているスロット ID 数)

**【戻り値】**

なし。

**【説明】**

srcptr が指すメモリに保存されている count 分のスロット ID を当該インスタンスにコピーします。

#### 14. 9. 3. 5 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshCjCarSlotList dst )
```

**【引数】**

dst

コピー先の DshCjCarSlotList インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

## 14. 10 DshProcControlSpec クラス

ProcessingCtrlSpec 属性情報を保存するためのクラスです。

### 14. 10. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明                           |
|---|--|------------------------------|
| 1 | public DshProcControlSpec ()                 | 空のインスタンスを生成します。              |
|   | public DshProcControlSpec (<br>string prjid) | プロジェクト ID を指定してインスタンスを生成します。 |

### 14. 10. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前   | 説明  |
|---|--|---|
| 1 | public string prjobid                      | プロジェクト ID です。   |
| 2 | public DshControlRule<br>control_rule_list | control rule 情報リストです。<br>DshControlRule クラスは 14. 11 で説明します。 |
| 3 | public DshOutputRule<br>output_rule_list   | output rule 情報リストです。<br>DshOutputRule クラスは 14. 12 で説明します。   |

### 14. 10. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                             | 説明   |
|---|--------------------------------|--|
| 1 | public void clear()            | control_rule_list, output_rule_list を消去します。  |
| 2 | public void set_prjid()        | プロジェクト ID を設定します。  |
| 3 | public void add_control_rule() | 1 個の control_rule 情報を追加します。  |
| 4 | public void add_output_rule()  | 1 個の output rule 情報を追加します。   |
| 5 | public void copy()             | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 6 | public void Dispose()          | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 10. 3. 1 clear()

control\_rule\_list, output\_rule\_list の内容を消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし。

**【戻り値】**

なし。

**【説明】**

control\_rule\_list, output\_rule\_list の内容を消去します。

消去には、それぞれ、DshControlRule クラスの clear() メソッド、DshOutputRule クラスの clear() メソッドを使用します。



### 14. 10. 3. 2 set\_prjobid()

プロセスジョブ ID を設定します。

#### 【構文】

```
public void set_prjid(string prjid)
```

#### 【引数】

prjid  
プロセスジョブ ID です。

#### 【戻り値】

なし。

#### 【説明】

プロセスジョブ ID を設定します。  
同時に、control\_rule\_list, output\_rule\_list の内容を消去します。

### 14. 10. 3. 3 add\_control\_rule()

インスタンスの control\_rule\_list にコントロールルール情報を 1 個追加します。

#### 【構文】

```
public void add_control_rule(string name, int format, int size, IntPtr value)
```

#### 【引数】

name  
コントロールルール名です。  
format  
ルール値のデータフォーマットです。(HSMS. ICODE\_U1 など)  
size  
ルール値の配列サイズです。  
value  
設定したいルール値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

control\_rule\_list に 1 個のコントロールルール情報を追加します。  
DshControlRule クラスの add\_control\_rule メソッドを使って追加します。

#### 14. 10. 3. 4 add\_output\_rule()

インスタンスの output\_rule\_list にアウトプットルール情報を1個追加します。

##### 【構文】

```
public void add_output_rule(int status, int format, int size, IntPtr value)
```

##### 【引数】

status

材料(material)の状態です。

format

ルール値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

ルール値の配列サイズです。

value

設定したいルール値が格納されているポインタです。

##### 【戻り値】

なし。

##### 【説明】

control\_rule\_list に1個のアウトプットルール情報を追加します。  
DshOutPutRule クラスの add\_output\_rule メソッドを使って追加します

#### 14. 10. 3. 5 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

##### 【構文】

```
public void copy( ref DshProcControlSpec dst )
```

##### 【引数】

dst

コピー先の DshProcControlSpec インスタンスです。

##### 【戻り値】

なし。

##### 【説明】

当該インスタンスの内容 (プロパティ値) を dst に指定されるインスタンスにコピーします。

## 14. 11 DshControlRule クラス

ProcessingCtrlSpec 属性のコントロールルール情報を保存するためのクラスです。

### 14. 11. 1 コンストラクタ

|   | 名前                      | 説明             |
|---|-------------------------|----------------|
| 1 | public DshControlRule() | 空のインスタスを生成します。 |

### 14. 11. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                                  | 説明  |
|---|-------------------------------------|---|
| 1 | public int count                    | 2 の control_list に保存されている control rule 情報の数です。                                  |
| 2 | public DshObjPara[]<br>control_list | control rule 情報リストです。<br>DshObjPara クラスの配列です。<br>(18.1 DshObjPara クラスを参照してください) |

### 14. 11. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                             | 説明   |
|---|--------------------------------|--|
| 1 | public void clear()            | control_list 配列の内容を空にします。  |
| 2 | public void add_control_rule() | 1 個の control_rule 情報を追加します。  |
| 3 | public void copy()             | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 4 | public void Dispose()          | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 11. 3. 1 clear()

control\_list 配列の内容を空にします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

control\_list 配列の内容を空にします。  
DshObjPara クラス内に使用されている非管理メモリを開放します。  
count = 0 にします。

### 14. 11. 3. 2 add\_control\_rule()

control\_list にコントロールルール情報を1個追加します。

#### 【構文】

```
public void add_control_rule(string name, int format, int size, IntPtr value)
```

#### 【引数】

name

コントロールルール名です。

format

ルール値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

ルール値の配列サイズです。

value

設定したいルール値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

control\_list 配列に1個のコントロールルール情報を追加します。  
追加した後、count + 1 します。

### 14. 11. 3. 3 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshControlRule dst )
```

**【引数】**

dst

コピー先のDshControlRuleクラスのインスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

## 14. 12 DshOutputRule クラス

ProcessingCtrlSpec 属性のアウトプットルール情報を保存するためのクラスです。

### 14. 12. 1 コンストラクタ

|   | 名前                     | 説明             |
|---|------------------------|----------------|
| 1 | public DshOutputRule() | 空のインスタスを生成します。 |

### 14. 12. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                                 | 説明   |
|---|------------------------------------|--|
| 1 | public int count                   | 2 の output_list に保存されている output rule 情報の数です。                                   |
| 2 | public DshObjPara[]<br>output_list | output rule 情報リストです。<br>DshObjPara クラスの配列です。<br>(18.1 DshObjPara クラスを参照してください) |

### 14. 12. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                            | 説明   |
|---|-------------------------------|--|
| 1 | public void clear()           | output_list 配列の内容を空にします。   |
| 2 | public void add_output_rule() | 1 個の output_rule 情報を追加します。   |
| 3 | public void copy()            | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 4 | public void Dispose()         | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 12. 3. 1 clear()

output\_list 配列の内容を空にします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

output\_list 配列の内容を空にします。

DshObjPara クラス内に使用されている非管理メモリを開放します。

count = 0 にします。



### 14. 12. 3. 2 add\_output\_rule()

インスタンスの output\_list にアウトプットルール情報を 1 個追加します。

#### 【構文】

```
public void add_output_rule(int status, int format, int size, IntPtr value)
```

#### 【引数】

status

材料 (material) の状態です。

format

ルール値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

ルール値の配列サイズです。

value

設定したいルール値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

output\_list 配列に 1 個のアウトプットルール情報を追加します。  
追加した後、count + 1 します。

### 14. 12. 3. 3 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

#### 【構文】

```
public void copy( ref DshOutputRule dst )
```

#### 【引数】

dst

コピー先の DshOutputRule インスタンスです。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンスの内容 (プロパティ値) を dst に指定されるインスタンスにコピーします。

## 14. 13 DshTextList クラス

コントロールジョブの属性で、テキスト配列情報を保存するためのクラスです。

### 14. 13. 1 コンストラクタ

|   | 名前                   | 説明             |
|---|----------------------|----------------|
| 1 | public DshTextList() | 空のインスタスを生成します。 |

### 14. 13. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                        | 説明                                |
|---|---------------------------|-----------------------------------|
| 1 | public int count          | 2 の text_list に保存されているテキスト情報の数です。 |
| 2 | public string[] text_list | テキスト情報を保存するための配列です。               |

### 14. 13. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void clear()    | output_list 配列の内容を空にします。   |
| 2 | public void add_text() | 1 個の text 情報を追加します。  |
| 3 | public void copy()     | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 4 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 13. 3. 1 clear()

text\_list 配列の内容を空にします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

text\_list 配列の内容を空にします。  
count = 0 にします。

### 14. 13. 3. 2 add\_text()

インスタンスの text\_list にテキスト情報を1個追加します。

**【構文】**

```
public void add_text(string text)
```

**【引数】**

text  
追加するテキストです。

**【戻り値】**

なし。

**【説明】**

text\_list 配列に1個のテキスト情報を追加します。  
追加した後、count + 1 します。

### 14. 13. 3. 3 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshTextList dst )
```

**【引数】**

dst  
コピー先の DshTextList インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

## 14. 14 DshS14Rsp クラス

プロセスジョブ関連 SECS-II メッセージの応答情報を保存するためのクラスです。  
S14F10, S14F12 メッセージが対象メッセージです。

### 14. 14. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明   |
|---|--|--|
| 1 | public DshS14Rsp()                               | 空のインスタンスを生成します。  |
| 2 | public DshS14Rsp( DshCj cj_class,<br>int objack) | Cj 情報と objack を指定してインスタンスを生成します。<br>DshCj クラスについては 14. 1 参照  |
| 3 | public DshS14Rsp(string objspec,<br>int objack)  | objspec と objack を指定してインスタンスを生成します。<br>S14F10 のためのコンストラクタです。 |

### 14. 14. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                        | 説明   |
|---|---------------------------|--|
| 1 | public objspec            | S14F10 に設定する objspec です。                             |
| 2 | public int objack         | S14F10, F12 に設定する objack です。                         |
| 3 | public DshCj cj_class     | 付属するコントロール情報 DshCj クラスのインスタンスです。                     |
| 4 | public int count          | 付属するエラー情報の数です。<br>使用 S14 メッセージ 共通です。                 |
| 5 | public DshObjError[] list | エラー情報を保存する配列です。<br>使用 S14 メッセージ 共通です。 Vol-1 19. 1 参照 |

### 14. 14. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                      | 説明   |
|---|-------------------------|--|
| 1 | public void clear()     | cj_class と list の配列に保存されている情報を消去します。   |
| 2 | public void init_set()  | cj_calss, objack, objspec を設定します。  |
|   | public void set_count() | エラーリスト list 配列のサイズを設定します。  |
| 3 | public void add_error() | エラー情報(DshObjError)を1個 list に追加します。   |
| 4 | public void Dispose()   | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 14. 3. 1 clear()

cj\_class に含む属性情報と list に設定されているエラー情報 (DshObjError の内容) を全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

cj\_class に含む属性情報と list に設定されているエラー情報 (DshObjError の内容) を全て消去します。  
cj\_class については、使用されている非管理メモリの開放も行います。  
そして、count = 0 にします。

### 14. 14. 3. 2 void init\_set()

cj\_class に含む属性情報と list に設定されているエラー情報 (DshObjError の内容) を全て消去します。

#### 【構文】

```
public void init_set(DshCj cj_class )  
public void init_set(DshCj cj_class, int objack)  
public void init_set(DshCj cj_class, string objspec, int objack)  
public void init_set(string objspec, int objack)
```

#### 【引数】

cj\_class  
DshCj クラスのインスタンスです。CJ 情報が保存されます。

objack  
S14F10, S14F12 に含まれる objack です。

objspec  
S14F10 に設定される objspec です。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンスのプロパティに情報を設定します。  
必要に応じて、オーバーロードされたメソッドを使用できます。

### 14. 14. 3. 3 set\_count()

list 配列に、指定されたエラー情報分のサイズの配列を作成します。

#### 【構文】

```
public void set_count(int count)
```

#### 【引数】

count

配列サイズです。

#### 【戻り値】

なし。

#### 【説明】

list の配列を引数に与えられたサイズにします。またプロパティの count に値を設定します。

本メソッドは、IntPtr で指された配列リストに格納されている複数のエラー情報をまとめて list 配列に設定する際に使用します。エラー情報設定に使用する関数は、DshObjError.copy\_err\_list() です。

list にエラー情報を設定したい場合は、通常、次に説明する add\_err() メソッドを使ってください。

### 14. 14. 3. 4 add\_err()

インスタンスの list 配列にエラー情報を 1 個追加します。

#### 【構文】

```
public void add_error(int code, string text)
```

#### 【引数】

code

エラーコードです。

text

エラーテキストです。

#### 【戻り値】

なし。

#### 【説明】

list 配列に、1 個のエラー情報を DshObjError のクラスに設定し、追加します。

追加した後、count を +1 します。



## 14. 15 DshS16F27Rsp クラス

プロセスジョブ関連 SECS-II メッセージの応答情報を保存するためのクラスです。  
S16F27 コントロールジョブコマンドに対する S16F28 応答メッセージが対象メッセージです。

### 14. 15. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前  | 説明  |
|---|---|---|
| 1 | public DshS16F27Rsp()   | 空のインスタンスを生成します。   |
| 2 | public DshS16F27Rsp(int ack)                                    | ack (acka)を指定してインスタンスを生成します。<br>err_flag=false にします。              |
| 3 | public DshS16F27Rsp(int ack,<br>int errcode,<br>string errtext) | ack, errcode, errtext 情報を指定してインスタンスを生成します。<br>err_flag=true にします。 |

### 14. 15. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                    | 説明   |
|---|-----------------------|--|
| 1 | public int acka       | 応答メッセージの設定する acka です。<br>使用 S16 メッセージ 共通です。                          |
| 2 | public bool err_flag  | errcode, errtext が有効かどうかを示します。<br>false=無効(エラー情報なし)、true=有効(エラー情報あり) |
| 3 | public int errcode    | エラーコードです。  |
| 4 | public string errtext | エラーテキストです。   |

### 14. 15. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                        | 説明   |
|---|---------------------------|--|
| 1 | public void set_obj_err() | value に使用されているメモリを開放します。   |
| 2 | public void Dispose()     | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 14. 15. 3. 1 set\_obj\_err ()

エラーコードとエラーテキストを設定します。

##### 【構文】

```
public void set_obj_err(int code, string text)
public void set_obj_err(int code, IntPtr text)
```

##### 【引数】

code  
エラーコードです。

text  
エラーテキストです。

##### 【戻り値】

なし。

##### 【説明】

エラーコードとエラーテキストを設定します。  
IntPtr で与えられた text は string に変換して設定します。

## 15. スプール関連クラス

スプール(SPOOL)関連情報を保存するクラスです。

関連クラスの一覧を次表に示します。

|   | クラス名            | 用途  |
|---|-----------------|---|
| 1 | DshSpool        | スプール情報保存クラスです。                                |
| 2 | DshSpoolList    | スプール情報保存用クラスです。<br>ただし、S16F27 が含む情報を保存します。    |
| 3 | DshSpoolUnit    | 1 個の stream のスプール情報を保存します。                    |
| 4 | DshSpoolRsp     | S2F43 に対する S2F44 応答メッセージ情報を保存するクラスです。         |
| 5 | DshSpoolRspUnit | S2F44 DshSpoolRsp に含まれる stream1 個分に対する応答情報です。 |

スプールの通信関連クラスとして、以下のものがあります。 これらについては、Vol-2 で説明します。

|                  |
|------------------|
| DshS2F44Response |
|------------------|

## 15. 1 DshSpool クラス

スプール情報を保存するために使用されます。

本クラスはエンジンに登録するスプール情報保存用に使用されます。

スプール対象になる可能性がある全てのストリームに対するスプール情報を保存することができます。

### 15. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                 | 説明            |
|---|--------------------|---------------|
| 1 | public DshSpool () | インスタンスを生成します。 |

### 15. 1. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                         | 説明   |
|---|----------------------------|--|
| 1 | public DshSpoolUnit[] list | stream=1 から 127 までの情報を保存するための配列です。<br>DshSpoolUnit クラスは 1 個のストリームのスプール情報を保存するためのクラスです。 |

### 15. 1. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                         | 説明   |
|---|----------------------------|--|
| 1 | public void clear_all()    | クラス内の全スプール情報を消去し、空にします。<br>エンジンの管理情報には影響しません。  |
| 2 | public void clear_stream() | クラス内の指定ストリームのスプール情報を消去します。エンジンの管理情報には影響しません。   |
| 3 | public void reset()        | エンジンに管理されているスプール情報を消去します。<br>クラス内の情報は消去しません。<br>全ストリームの消去またはストリームを指定して、その情報だけを消去します。                     |
| 4 | public void add_func()     | クラス内のストリームにファンクションを1個追加します。  |
| 5 | public int set()           | クラス内に設定された指定ストリームのスプール情報をエンジンに設定します。   |
| 6 | public int get()           | 指定されたストリームにスプール対象として設定されているファンクションを配列に取得します。   |
| 7 | public void Dispose()      | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスのDispose()も行います。<br>14.1.3.13と同様です。そちらを参照ください。 |

### 15. 1. 3. 1 clear\_all()

クラス内に設定されているスプール情報を全て消去します。

**【構文】**

```
public void clear_all()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

クラス内の全てのストリームのスプール情報を消去します。  
各ストリームに与えられた DshSpoolUnit のファンクションリストを空にし、count = 0 にします。  
エンジン管理情報のスプール情報は、消去しません。エンジン管理情報上のスプール情報を消去するためには、reset() メソッドを使用してください。

### 15. 1. 3. 2 clear ()

指定されたストリームに対して設定されているスプール情報を消去し、空にします。

**【構文】**

```
public void clear(int s)
```

**【引数】**

s  
スプール情報を消去したいストリームコードです。

**【戻り値】**

なし。

**【説明】**

クラス内の s で指定されたストリームのスプール情報を全て消去します。  
s ストリームに与えられた DshSpoolUnit のファンクションリストを空にし、count = 0 にします。

### 15. 1. 3. 3 reset()

エンジンの管理情報の中に設定されているスプール情報を全て消去します。

#### 【構文】

```
public void reset()  
public void reset( int s )
```

#### 【引数】

s

スプール情報を消去したいストリームコードです。

#### 【戻り値】

なし。

#### 【説明】

引数がない（ストリームの指定なし）の場合、エンジン管理情報の全ストリームのスプール情報を全て消去します。

引数 s が指定されていた場合は、エンジン管理情報の中から s で指定されたスプール情報だけを消去します。クラス内のスプール情報は、消去しません。クラス内の消去は、clear\_all() または clear() メソッドを使用してください。

### 15. 1. 3. 4 add\_func ()

ストリームに対してファンクションを1個追加します。  
クラス内のストリームが対象です。

#### 【構文】

```
public void add_func(int s, int f)
```

#### 【引数】

s

ストリームコードです。

f

追加するファンクションコードです。

#### 【戻り値】

なし。

#### 【説明】

クラス内の s で指定されたストリームのスプール情報に f で指定されたファンクションを1個追加します。  
list[s]のDshSpoolUnitのfunc\_listに1個追加します。

### 15. 1. 3. 5 set ()

クラス内に設定されている s に指定されたストリームのスプール情報をエンジンの管理情報に設定します。

**【構文】**

```
public int set(int s)
```

**【引数】**

s  
ストリームコードです。

**戻り値】**

| 返却値  | 意味        |
|------|-----------|
| 0    | 正常に設定できた。 |
| (-1) | 設定に失敗した。  |

**【説明】**

クラス内の s で指定されたストリームのスプール情報に f で指定されたファンクションを 1 個追加します。  
list[s]のDshSpoolUnitのfunc\_listに1個追加します。



### 15. 1. 3. 6 get()

エンジン管理情報から、sに指定されたストリームに設定されているスプール情報を取得します。

#### 【構文】

```
public int get(int s, int[] flist)
```

#### 【引数】

s

ストリームコードです。

flist

スプール情報に設定されているファンクションを格納するための配列です。

#### 戻り値】

| 返却値  | 意 味  |
|------|--|
| >= 0 | 取得できたファンクションの数です。<br>0はファンクションがまったく設置されていなかったことを意味します。 |
| (-1) | 取得に失敗した。   |

#### 【説明】

エンジン管理情報から、sで指定されたストリームに設定されているファンクションをflistの配列領域に取得します。

戻り値は、設定されていたファンクションの数です。(-1)の値の場合は、sの値が正しくなかったことを意味します。

## 15. 2 DshSpoolList クラス

S2F43 メッセージから得られたスプール情報を保存するために使用されます。

15. 1 の DshSpool クラスは、エンジンに設定される全てのストリームスプール情報を保存できますが、本クラスは、S2F43 に含まれているストリーム分だけのスプール情報を保存します。

### 15. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                     | 説明              |
|---|------------------------|-----------------|
| 1 | public DshSpoolList () | 空のインスタンスを生成します。 |

### 15. 2. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                         | 説明   |
|---|----------------------------|--|
| 1 | public int count           | スプール情報を保存しているストリームの数です。  |
| 2 | public DshSpoolUnit[] list | S2F43 に含まれているスプール情報を順次保存するための DshSpoolUnit クラスの配列です。<br>DshSpoolUnit クラスは1個のストリームのスプール情報を保存するためのクラスです。 |

## 15. 2. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                       | 説明  |
|---|--------------------------|---|
| 1 | public void clear ()     | クラス内の全スプール情報を消去し、空にします。   |
| 2 | public void add_stream() | 1個のストリームを list 配列に追加します。  |
| 3 | public int get_func()    | func_list からファンクションを1個取得します。<br>配列位置を指定します。   |
| 4 | public void Dispose()    | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14.1.3.13 と同様です。そちらを参照ください。 |

### 15. 2. 3. 1 clear()

クラス内に設定されているスプール情報を全て消去します。

#### 【構文】

```
public void clear ()
```

#### 【引数】

なし

#### 【戻り値】

なし。

#### 【説明】

クラス内に含む全てのファンクションを消去します。

各ストリームに与えられた DshSpoolListUnit のファンクションリストを空にし、count = 0 にします。

## 14. 2. 3. 2 decode()

S2F43 メッセージに含まれるスプール情報を DshSpoolList クラス内のプロパティにデコードします。  
本メソッドは S2F43 メッセージ受信処理時に使用します。

### 【構文】

```
public int decode(ref DSHMSG msg)
```

### 【引数】

msg

S2F43 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。  
DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。  
ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

### 【戻り値】

| 返却値  | 意味                                   |
|------|--------------------------------------|
| 0    | 正常にデコードできた。                          |
| (-1) | デコードできなかった (S2F43 のメッセージの形式が正しくなかった) |

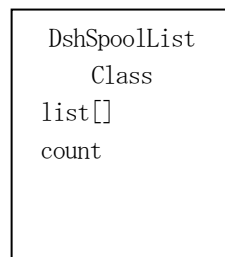
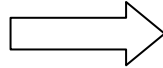
### 【説明】

msg に含まれている情報を DshSpoolList クラス内にデコードします。

msg S2F43

```
L, m
L, 2
  strid1
L, n
  fcid1
  fcid2
.
L, 2
  strid2
.
```

decode



### 15. 3 DshSpoolUnit クラス

1個のストリームのスプール情報を保存するために使用されます。

#### 15. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前  | 説明                     |
|---|---|------------------------|
| 1 | <code>public DshSpoolUnit()</code>        | 空のインスタスを生成します。         |
| 2 | <code>public DshSpoolUnit( int s )</code> | ストリームを指定してインスタスを生成します。 |

#### 15. 3. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                                  | 説明               |
|---|-------------------------------------|------------------|
| 1 | <code>public int stream</code>      | ストリームです。         |
| 2 | <code>public int count</code>       | 保存している関数の数です。    |
| 3 | <code>public int[] func_list</code> | ストリームに含む関数の配列です。 |

### 15. 3. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明  |
|---|------------------------|---|
| 1 | public void clear()    | クラス内の全スプール情報を消去し、空にします。   |
| 2 | public void add_func() | 1個のファンクションを func_list 配列に追加します。   |
| 3 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14.1.3.13 と同様です。そちらを参照ください。 |

#### 15. 3. 3. 1 clear()

クラス内に設定されているファンクション情報を全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

クラス内の func\_list に含むファンクション情報を消去します。count = 0 にします。

### 15. 3. 3. 2 add\_func ()

ファンクションを1個追加します。

**【構文】**

```
public void add_func(int f)
```

**【引数】**

f  
追加するファンクションコードです。

**【戻り値】**

なし。

**【説明】**

クラスの func\_list に f で指定されたファンクションを1個追加します。count+1 します。

## 15. 4 DshSpoolRsp クラス

S2F43 の応答メッセージ S2F44 に対する応答情報を保存するために使用されます。

### 15. 4. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                             | 説明                       |
|---|--------------------------------|--------------------------|
| 1 | public DshSpoolRsp()           | 空のインスタスを生成します。           |
| 2 | public DshSpoolRsp( int rsack) | 応答 ack を指定してインスタスを生成します。 |

### 15. 4. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                            | 説明  |
|---|-------------------------------|---|
| 1 | public int rsack              | S2F44 の ack である rsack の値です。   |
| 2 | public int count              | クラスに含まれる DshSpoolRspUnit クラスの数です。   |
| 3 | public DshSpoolRspUnit[] list | S2F43 メッセージでうけいれることができなかったスプール応答情報が保存される配列です。<br>DshSpoolRspUnit については 15.6 で説明します。 |



### 15. 4. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                    | 説明   |
|---|-----------------------|--|
| 1 | public void clear()   | クラス内の全スプール応答情報を消去し、空にします。  |
| 2 | public void add ()    | ストリーム1個分のスプール応答情報を list に追加します。  |
| 3 | public void Dispose() | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 15. 4. 3. 1 clear()

クラス内に設定されているスプール応答情報を全て消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

クラス内の list に含まれる全てのストリームのスプール応答情報を消去します。  
count=0 にします。

### 15. 4. 3. 2 add()

1個のストリームのスプール応答情報を list に追加します。  
ストリームに対する ack である strack、ストリーム、ファンクション情報が含まれています。

**【構文】**

```
public void add(DshSpoolRspUnit uclass)
```

**【引数】**

uclass  
1個のストリームに対する応答情報です。

**【戻り値】**

なし。

**【説明】**

uclass の含まれているスプール応答情報を list に追加します。  
追加後、count + 1 します。

## 15. 5 DshSpoolRspUnit クラス

S2F43 の応答メッセージ S2F44 に対する 1 個のストリームに対する応答情報を保存するために使用されます。  
本クラスは、DshSpoolUnit クラスから派生しています。

### 15. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                                  | 説明                                 |
|---|-------------------------------------|------------------------------------|
| 1 | public DshSpoolRspUnit()            | 空のインスタンスを生成します。                    |
| 2 | public DshSpoolRspUnit( int strack) | ストリームに対する応答 ack を指定してインスタンスを生成します。 |

### 15. 5. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                         | 説明  |
|---|----------------------------|---|
| 1 | public int strack          | S2F44 の 1 個のストリームに対する ack である strack の値です。                    |
| 2 | public int count           | 派生元 DshSpoolUnit 内のプロパティです。<br>クラスに含まれる DshSpoolUnit クラスの数です。 |
| 3 | public DshSpoolUnit[] list | 派生元 DshSpoolUnit 内のプロパティです。<br>クラスに含まれる DshSpoolUnit の配列です。   |

### 15. 5. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                    | 説明   |
|---|-----------------------|--|
| 1 | public void Dispose() | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

## 16. ホストコマンド、キャリアアクションメッセージ関連クラス

S2F41, S2F49, S3F17, S3F23, S3F25とS3F27メッセージで使用するクラスについて説明します。

関連クラスの一覧を次表に示します。

|   | クラス名               | 用途   |
|---|--------------------|--|
| 1 | DshRCmd            | S2F41 ホストコマンドメッセージ情報を保存するクラスです。                      |
| 2 | DshERCmd           | S2F49 拡張リモートコマンドメッセージ情報を保存するクラスです。                   |
| 3 | DshCarAction       | S3F17 キャリアアクション要求メッセージ情報を保存するクラスです。                  |
| 4 | DshCarActionPara   | S3F17 キャリアアクション要求メッセージに含まれるパラメータ情報を保存するクラスです。        |
| 5 | DshContentMap      | S3F17 キャリアアクション要求メッセージに含まれる contentmap 情報を保存するクラスです。 |
| 6 | DshSlotMap         | S3F17 キャリアアクション要求メッセージに含まれるスロットマップ情報を保存するクラスです。      |
| 7 | DshPortGroupAction | S3F23 ポートグループアクション要求メッセージ情報を保存するクラスです。               |
| 8 | DshPortAction      | S3F25 ポートアクション要求メッセージ情報を保存するクラスです。                   |
| 9 | DshPortAccess      | S3F27 ポートアクセス変更メッセージ情報を保存するクラスです。                    |

通信関連クラスとして、以下のものがあります。これらについては、Vol-2で説明します。

|                  |
|------------------|
| DshS2F42Response |
| DshS2F50Response |
| DshS3F18Response |
| DshS3F24Response |
| DshS3F24Response |
| DshS3F28Response |

## 16. 1 DshRCmd クラス

S2F41 ホストリモートコマンド送信メッセージに含まれるコマンド情報を保存するためのクラスです。

decode メソッドによって S2F41 メッセージに含まれる情報を本クラス内にデコードすることができます。

### 16. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                         | 説明                      |
|---|----------------------------|-------------------------|
| 1 | public DshRCmd()           | 空のインスタンスを生成します。         |
| 2 | public DshRCmd(string cmd) | インスタンスをコマンド名を指定して生成します。 |

### 16. 1. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                              | 説明  |
|---|---------------------------------|---|
| 1 | public string cmd               | ホスト(リモート)コマンドです。  |
| 2 | public int count                | 含まれるコマンドパラメタの数です。   |
| 3 | public DshObjPara[] list        | コマンドパラメタ情報を保存するための DshObjPara クラスの配列です。<br>パラメタ値が LIST 構造(ICODE_L)の場合は、その LIST に含まれるサブパラメタは次の sub_list に保存されます。 |
| 4 | public DshObjSubPara[] sub_list | 上の list のパラメタ値が LIST 構造の場合、その LIST に含まれるパラメタ情報を保存します。<br>LIST 構造ではない位置の配列は null になります。                          |

### 16. 1. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                        | 説明   |
|---|---------------------------|--|
| 1 | public void clear()       | list と sub_list 配列内の情報を消去します。  |
| 2 | public void set_cmd()     | ホストコマンドを設定します。   |
| 3 | public void add_para()    | list 配列に1個のコマンドパラメータを追加します。  |
| 4 | public int add_sub_para() | sub_list 配列に1個のサブパラメータを追加します。  |
| 5 | public int decode()       | S2F41メッセージを当該クラスにデコードします。  |
| 6 | public void Dispose()     | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 1. 3. 1 clear()

クラス内に設定されているコマンドパラメータを全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

クラスの list, sub\_list 配列内に含む全てのコマンドパラメータを消去します。  
非管理メモリを使用しているパラメータ値についてはメモリを開放します。  
そして、count = 0にします。

### 16. 1. 3. 2 set\_cmd()

ホストコマンドを設定します。

**【構文】**

```
public void set_cmd(string cmd)
```

**【引数】**

cmd

ホストコマンド名です。

**【戻り値】**

なし。

**【説明】**

引数に与えられたコマンドを cmd プロパティに設定します。

### 16. 1. 3. 3 add\_para()

インスタンスの list 配列にコマンドパラメータ情報を 1 個追加します。

**【構文】**

```
public void add_para(string pname, int format, int size, IntPtr value)
public void add_para(string pname, string value)
```

**【引数】**

pname

パラメータ名です。

format

パラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

パラメータ値の配列サイズです。

value

設定したいパラメータ値が格納されているポインタです。

**【戻り値】**

なし。

**【説明】**

list 配列に 1 個のパラメータ情報を DshObjPara のクラスに設定し、追加します。  
string で与えられた value は HSMS. ICODE\_A のフォーマットとして保存されます。

format が HSMS. ICODE\_L、リストの場合は、sub\_list に DshObjSubPara クラスのインスタンスを生成します。  
( sublist[count] の配列に)  
追加した後、count を +1 します。

### 16. 1. 3. 4 add\_sub\_para()

インスタンスの sub\_list 配列にコマンドパラメータ情報を1個追加します。

#### 【構文】

```
public int add_sub_para(int pos, int format, int size, IntPtr value)
public int add_sub_para(int pos, string value)
```

#### 【引数】

pos

sub\_list の配列位置を指定します。

format

パラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

パラメータ値の配列サイズ

value

設定したいパラメータ値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

sub\_list の pos で指定された配列位置に1個のパラメータ情報を追加します。  
string で与えられた value は HSMS. ICODE\_A のフォーマットとして保存されます。

### 16. 1. 3. 5 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

#### 【構文】

```
public void copy( ref DshRCmd dst )
```

#### 【引数】

dst

コピー先の DshRCmd インスタンスです。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。



### 16. 1. 3. 6 decode()

S2F41 に含まれるホストコマンド情報を DshRCmd クラス内のプロパティにデコードします。

#### 【構文】

```
public int decode(ref DSHMSG msg)
```

#### 【引数】

msg

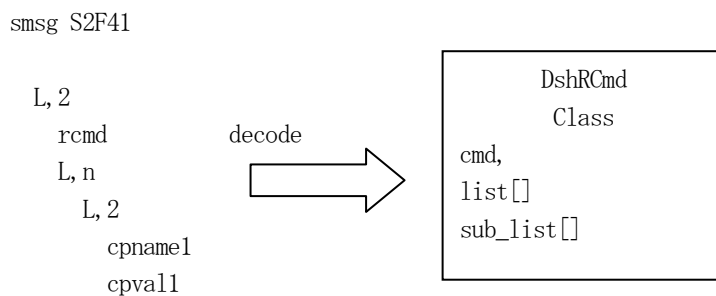
S2F41 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

#### 【戻り値】

| 返却値  | 意味                                    |
|------|---------------------------------------|
| 0    | 正常にデコードできた。                           |
| (-1) | デコードできなかった (S2F41 のメッセージの形式が正しくなかった。) |

#### 【説明】

msg に含まれている情報を DshRCmd クラス内にデコードします。



メッセージに含まれるコマンドとパラメータがクラスのプロパティに保存されます。

正常にデコードできた場合、0 を返却します。

もし、S2F41 のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1) を返却します。

## 16. 2 DshERCmd クラス

S2F49 拡張リモートコマンド送信メッセージに含まれるコマンド情報を保存するためのクラスです。

decode メソッドによって S2F49 メッセージに含まれる情報を本クラス内にデコードすることができます。

### 16. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明                               |
|---|--|----------------------------------|
| 1 | public DshERCmd()                              | 空のインスタスを生成します。                   |
| 2 | public DshERCmd(string objspec,<br>string cmd) | インスタスを objspec とコマンド名を指定して生成します。 |

## 16. 2. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                                | 説明  |
|---|-----------------------------------|---|
| 1 | public int eqid                   | 装置 ID です。<br>デフォルト値は 0 です。  |
| 2 | public string cmd                 | ホスト(リモート)コマンドです。<br>(下の表参照)   |
| 3 | public int count                  | 含まれるコマンドパラメータの数です。  |
| 4 | public DshObjParaList[] list      | コマンドパラメータ情報を保存するための DshObjParaList クラスの配列です。<br>パラメータのネスティング情報も保存されます。                    |
| 5 | public DshDataItemList[] sub_list | 上の list のパラメータ値が LIST 構造の場合で、複数のパラメータを有するものについてデータ情報を保存します。<br>使用されていない場合は、null がセットされます。 |

### 拡張リモートコマンド名

| コマンド名         |
|---------------|
| "abort"       |
| "cancel"      |
| "infoupdate"  |
| "pause"       |
| "resume"      |
| "stage"       |
| "stagedelete" |
| "transfer"    |
| "pp_select"   |
| "start"       |
| "stop"        |

## 16. 2. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                           | 説明   |
|---|------------------------------|--|
| 1 | public void clear()          | list と sub_list 配列内の情報を消去します。  |
| 2 | public void set_cmd()        | objspec と拡張リモートコマンドを設定します。   |
| 3 | public void add_para()       | list 配列に 1 個のコマンドパラメータを追加します。  |
| 4 | public int add_sub_para()    | sub_list 配列に 1 個のサブパラメータを追加します。  |
| 5 | public int set_multi()       | パラメータが、cpname, cpval のリスト構造で多重にネ스팅した場合に、指定された list 位置に対して multi フラグをセットします。                                    |
| 6 | public void add_multi_para() | cpname, cpval のリスト構造が多重にネ스팅したパラメータに対して 1 個のサブパラメータを追加します。  |
| 7 | public int decode()          | S2F49 メッセージを当該クラスにデコードします。   |
| 8 | public void Dispose()        | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

### 16. 2. 3. 1 clear()

クラス内に設定されているコマンドパラメータを全て消去します。

#### 【構文】

```
public void clear ()
```

#### 【引数】

なし

#### 【戻り値】

なし。

#### 【説明】

クラスの list, sub\_list 配列内に含む全てのコマンドパラメータを消去します。  
非管理メモリを使用しているパラメータ値についてはメモリを開放します。  
そして、count = 0 にします。

## 16. 2. 3. 2 set\_cmd()

objspec と拡張リモートコマンドを設定します。

### 【構文】

```
public void set_cmd(string objspec, string cmd)
```

### 【引数】

objspec

objspec (オブジェクト仕様) です。

cmd

拡張リモートコマンド名です。

### 【戻り値】

なし。

### 【説明】

引数に与えられた objspec とコマンドを objspec, cmd プロパティに設定します。

### 16. 2. 3. 3 add\_para()

インスタンスの list 配列にコマンドパラメータ情報を 1 個追加します。

#### 【構文】

```
public void add_para(string pname, int format, int size, IntPtr value)
public void add_para(string pname, string value)
```

#### 【引数】

pname

パラメータ名です。

format

パラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

パラメータ値の配列サイズです。

value

設定したいパラメータ値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

list 配列に、1 個のパラメータ情報を DshObjParaList のクラスに設定し、追加します。

string で与えられた value は HSMS. ICODE\_A のフォーマットとして保存されます。

format が HSMS. ICODE\_L、リストの場合は、sub\_list に DshDataItemList クラスのインスタンスを生成します。  
( sublist[count] の配列に)

追加した後、count を +1 します。

## 16. 2. 3. 4 add\_sub\_para()

インスタンスの sub\_list 配列にコマンドパラメータ情報を1個追加します。

### 【構文】

```
public int add_sub_para(int pos, int format, int size, IntPtr value)
public int add_sub_para(int pos, string value)
```

### 【引数】

pos

sub\_list の配列位置を指定します。

format

パラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

パラメータ値の配列サイズです。

value

設定したいパラメータ値が格納されているポインタです。

### 【戻り値】

なし。

### 【説明】

sub\_list の pos で指定された配列位置に1個のパラメータ情報を追加します。  
string で与えられた value は HSMS. ICODE\_A のフォーマットとして保存されます。

### 16. 2. 3. 5 set\_multi()

list 配列の指定された配列位置のパラメータが cpname, cpval の構造でネスティングする旨のフラグを設定します。

**【構文】**

```
public int set_multi(int index)
```

**【引数】**

index

list の配列位置を指定します。

**戻り値】**

| 返却値  | 意味                                |
|------|-----------------------------------|
| 0    | 正常に設定できた。                         |
| (-1) | 設定に失敗した。index で指定された配列位置は未設定であった。 |

**【説明】**

index で指定された list の配列位置のパラメータ情報に対し、同じ構造の情報がネスティングすることを示すフラグ multi\_flag に 1 を設定します。(multi\_flag については DshERCPaList クラスの説明を参照下さい。

もし、list[index] のパラメータが未設定の場合は (-1) を返却します。



### 16. 2. 3. 6 add\_multi\_para()

インスタンスの list 配列にコマンドパラメータ情報を 1 個追加します。

#### 【構文】

```
public int add_multi_para(int index, string pname, int format, int size, IntPtr value)
public int add_multi_para(int index, string pname, string value)
```

#### 【引数】

index  
ネスティングしたパラメータを加えたい list 配列の位置を指定します。

pname  
パラメータ名です。

format  
パラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size  
パラメータ値の配列サイズです。

value  
設定したいパラメータ値が格納されているポインタです。

#### 【戻り値】

| 返却値  | 意味                                |
|------|-----------------------------------|
| 0    | 正常に追加できた。                         |
| (-1) | 追加に失敗した。index で指定された配列位置は未設定であった。 |

#### 【説明】

list の index で指定された位置の配列に多重パラメータとして 1 個のサブパラメータ情報を追加します。  
string で与えられた value は HSMS. ICODE\_A のフォーマットとして保存されます。

追加した後、count を +1 します。

もし、list の index で指定された配列位置が未設定であったり、multi\_flag が =1 でなかった場合には、(-1) を返却します。

### 16. 2. 3. 7 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshERCmd dst )
```

**【引数】**

dst  
コピー先のDshERCmd インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

### 16. 2. 3. 8 decode()

S2F49に含まれる拡張リモートコマンド情報を DshERCmd クラス内のプロパティにデコードします。

#### 【構文】

```
public int decode(ref DSHMSG msg)
```

#### 【引数】

msg

S2F49 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

#### 【戻り値】

| 返却値  | 意味                                    |
|------|---------------------------------------|
| 0    | 正常にデコードできた。                           |
| (-1) | デコードできなかった (S2F49 のメッセージの形式が正しくなかった。) |

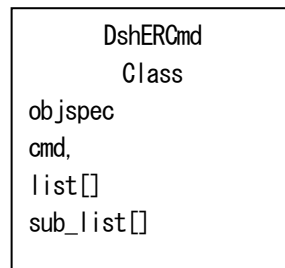
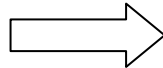
#### 【説明】

msg に含まれている情報を DshERCmd クラス内にデコードします。

msg S2F49

```
L, 4
  dataid
  objspec
  rcmd
  L, m
    L, 2
      cpname
      cpval
  .
```

decode



メッセージに含まれるコマンドとパラメータがクラスのプロパティに保存されます。

正常にデコードできた場合、0 を返却します。

もし、S2F49 のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1) を返却します。

### 16. 3 DshCarAction クラス

S3F17 キャリアアクション送信メッセージに含まれるキャリアアクション情報を保存するためのクラスです。

decode メソッドによって S3F17 メッセージに含まれる情報を本クラス内にデコードすることができます。

#### 16. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明   |
|---|--|--|
| 1 | public DshCarAction()  | 空のインスタスを生成します。                               |
| 2 | public DshCarAction(<br>string caction,<br>string carid,<br>int port)    | インスタスを caction, キャリア名, ポートを指定して生成します。        |
| 3 | public DshCarAction(<br>int caction_index,<br>string carid,<br>int port) | インスタスを caction のインデクス, キャリア名, ポートを指定して生成します。 |

### 16. 3. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                             | 説明  |
|---|--------------------------------|---|
| 1 | public string carid            | キャリア ID です。                                     |
| 2 | public int default_capacity    | 容量のデフォルト値です。<br>装置起動ファイルの capacity で設定される値です。   |
| 3 | public string action           | アクション名です。(下の表参照)                                |
| 4 | public caction_index           | アクション名に対するインデクス値です。<br>(action から変換します)         |
| 5 | public int port                | ポートポートです。                                       |
| 6 | public int count               | アクション属性情報の数です。                                  |
| 7 | public DshCarActionPara[] list | アクション属性(パラメタ)情報を保存する<br>DshCarActionPara の配列です。 |

caction\_index - キャリアアクションインデクス値 / アクション名

| index 値 | 定数名 - index                  | アクション名 - action             |
|---------|------------------------------|-----------------------------|
| 0       | CA_Bind                      | "Bind"                      |
| 1       | CA_CancelBind                | "CancelBind"                |
| 2       | CA_CancelCarrier             | "CancelCarrier"             |
| 3       | CA_CancelCarrierAtPort       | "CancelCarrierAtPort"       |
| 4       | CA_CancelCarrierNotification | "CancelCarrierNotification" |
| 5       | CA_CancelCarrierOut          | "CancelCarrierOut"          |
| 6       | CA_CarrierIn                 | "CarrierIn"                 |
| 7       | CA_CarrierNotification       | "CarrierNotification"       |
| 8       | CA_CarrierOut                | "CarrierOut"                |
| 9       | CA_CarrierReCreate           | "CarrierReCreate"           |
| 10      | CA_CarrierRelease            | "CarrierRelease"            |
| 11      | CA_ProceedWithCarrier        | "ProceedWithCarrier"        |

### 16. 3. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                        | 説明   |
|---|---------------------------|--|
| 1 | public void clear()       | list 配列内の属性情報を消去します。   |
| 2 | public void init_set()    | アクション、キャリア ID, ポートを設定します。  |
| 3 | public void add_para()    | list 配列に 1 個の属性を追加します。   |
| 4 | public int get_attr_pos() | 指定属性(パラメタ)が保存されている配列位置を取得する。   |
| 5 | public int decode()       | S3F17 メッセージを当該クラスにデコードします。   |
| 6 | public void Dispose()     | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 3. 3. 1 clear()

クラス内に設定されている属性情報を全て消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

クラスの list, sub\_list 配列内に含む全てのキャリア属性を消去します。  
非管理メモリを使用している属性値についてはメモリを開放します。  
そして、count = 0 にします。

### 16. 3. 3. 2 init\_set()

アクション名、キャリア ID とポートを設定します。

#### 【構文】

```
public void init_set(string caction, string carid, int port)
public void init_set(int caction_index, string carid, int port)
```

#### 【引数】

caction

キャリアアクション名です。(文字列)

caction\_index

キャリアアクションのインデクス値です。

(アクションインデクスは dsh\_const クラス内に定義されています。)

carid

キャリア ID です。

port

ポート番号です。

#### 【戻り値】

なし。

#### 【説明】

引数に与えられた caction 名、carid, port をそれぞれプロパティに設定します。  
設定時に 属性リスト list 配列を空にします。

### 16. 3. 3. 3 add\_para()

インスタンスの list 配列にキャリア属性情報を 1 個追加します。

#### 【構文】

```
public void add_para(string cattrid, int attrdata)
public void add_para(string cattrid, string attrdata)
public void add_para(int cattr_index, string attrdata)
public void add_para(int cattr_index, int attrdata)
```

#### 【引数】

cattrid

属性 ID です。

cattrid\_index

属性 ID インデクスです。 (dsh\_const クラスに定義されているインデクス値です。)

attrdata

属性値です。文字列型(string)と整数型(int)のものがあります。

#### 【戻り値】

なし。

#### 【説明】

list 配列に、1 個の属性情報を DshObjParaList のクラスに設定し、追加します。  
cattrid attr\_index によって属性値の型が決まっていますので、正しく設定してください。  
追加した後、count を +1 します。



### 16. 3. 3. 4 get\_attr\_pos ()

特定の属性が保存されている配列位置を検索します。

**【構文】**

```
public int get_attr_pos(int attr_index)
```

**【引数】**

index

属性のインデクス値です。(dsh\_const クラスで定義されています。)

**戻り値】**

| 返却値  | 意 味                   |
|------|-----------------------|
| >= 0 | 保存されている list の配列位置です。 |
| (-1) | 見つけることができなかった。        |

**【説明】**

list の属性情報配列リスト上に attr\_index で指定された属性が存在するかどうかを検索します。  
 見つかったら、list の配列位置を返却します。  
 見つからなかったら、(-1)を返却します。

### 16. 3. 3. 5 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshCarAction dst )
```

**【引数】**

dst

コピー先の DshCarAction インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容 (プロパティ値) を dst に指定されるインスタンスにコピーします。

### 16. 3. 3. 6 decode()

S3F17に含まれるキャリアアクション情報をDshCarActionクラス内のプロパティにデコードします。

#### 【構文】

```
public int decode(ref DSHMSG msg)
```

#### 【引数】

msg

S3F17のメッセージ情報（生情報）が格納されているDSHMSG構造体領域になります。DSHMSGは、1次メッセージをポーリングした際にエンジンから与えられる情報です。ユーザはDSHMSG構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

#### 【戻り値】

| 返却値  | 意味                                  |
|------|-------------------------------------|
| 0    | 正常にデコードできた。                         |
| (-1) | デコードできなかった（S3F17のメッセージの形式が正しくなかった。） |

#### 【説明】

msgに含まれている情報をDshCarActionクラス内にデコードします。

msg S3F17

L, 5

dataid

carrieraction

carrierspec

ptn

L, n

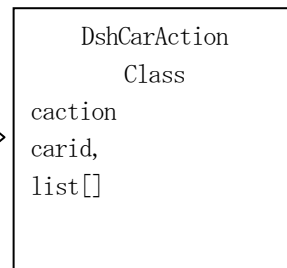
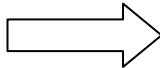
L, 2

catridl

catrdata1

.

decode



メッセージに含まれるアクション名、キャリアIDと属性がクラスのプロパティに保存されます。

正常にデコードできた場合、0を返却します。

もし、S3F17のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1)を返却します。

## 16. 4 DshCarActionPara クラス

S3F17 キャリアアクション送信メッセージに含まれる1個のアクション属性情報を保存するためのクラスです。

### 16. 4. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                         | 説明              |
|---|----------------------------|-----------------|
| 1 | public DshCarActionPara () | 空のインスタンスを生成します。 |

## 16. 4. 2 プロパティ

プロパティ一覧表に示します。

|    | 名前                               | 説明   |
|----|----------------------------------|--|
| 1  | public string attrid             | アクション属性 ID です。(下の表参照)                                    |
| 2  | public int attr_index            | 属性 ID をインデクスに変換したものです。<br>dsh_const クラスに定義されています。        |
| 3  | public DshContentMap content_map | ContentMap 情報を保存します。<br>DshContentMap クラスに保存します。 16.5 参照 |
| 4  | public DshSlotMap slot_map       | SlotMap 情報を保存します。<br>DshSlotMap クラスに保存します。 16.6 参照       |
| 5  | public string objtype            | オブジェクトタイプです。   |
| 6  | public string objid              | オブジェクト ID をです。   |
| 7  | public int capacity              | キャリア容量(slot 数)です。  |
| 8  | public int access_status         | アクセス状態です。  |
| 9  | public int id_status             | キャリア ID 読取り状態です。   |
| 10 | public string location           | キャリアの所在位置です。   |
| 11 | public int map_status            | スロットマップ の読取り状態です。  |
| 12 | public int subst_count           | 基板数です。   |
| 13 | public string usage              | usage(用途)です。   |

キャリアアクション属性インデクス値 / 属性 ID

| index 値 | 定数名 - attr_index          | 属性 ID - attrid           |
|---------|---------------------------|--------------------------|
| 0       | CA_ObjType                | "ObjType"                |
| 1       | CA_ObjId                  | "ObjId"                  |
| 2       | CA_Capacity               | "Capacity"               |
| 3       | CA_CarrierAccessingStatus | "CarrierAccessingStatus" |
| 4       | CA_CarrierIDStatus        | "CarrierIDStatus"        |
| 5       | CA_ContentMap             | "ContentMap"             |
| 6       | CA_LocationID             | "LocationID"             |
| 7       | CA_SlotMap                | "SlotMap"                |
| 8       | CA_SlotMapStatus          | "SlotMapStatus"          |
| 9       | CA_SubStrateCount         | "SlotMapStatus"          |
| 10      | CA_Usage                  | "Usage"                  |

### 16. 4. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                            | 説明  |
|---|-------------------------------|---|
| 1 | public void clear()           | content_map, slot_map の内容を消去します。  |
| 2 | public void set_info()        | 1 個の属性情報を設定します。   |
| 3 | public void add_content_map() | content_map の要素情報を追加します。<br>lotid, substid  |
| 4 | public void add_slot_map()    | slot_map に slotid を追加します。   |
| 5 | public void Dispose()         | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14.1.3.13 と同様です。そちらを参照ください。 |

#### 16. 4. 3. 1 clear()

クラス内の content\_map, slot\_map プロパティの内容を消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

content\_map, slot\_map プロパティの内容を消去し、空にします。

### 16. 4. 3. 2 set\_info()

属性 ID、属性インデクスと属性値を設定します。

#### 【構文】

```
public void set_info(string attrid, int attr_index, string attrdata)
public void set_info(string attrid, int attr_index, int attrdata)
public void set_info(int attr_index, int attr_data)
public void set_info(int attr_index, string attr_data)
```

#### 【引数】

attrid  
属性 ID です。

attr\_index  
属性 ID のインデクス値です。  
(属性インデクスは dsh\_const クラス内に定義されています。)

attr\_data  
属性値です。文字列型と整数型の 1 つの種類があります。

#### 【戻り値】

なし。

#### 【説明】

引数に与えられた属性 ID または属性インデクスに従って、それに対応したプロパティに値を設定します。  
attr\_index が指定された場合、インデクスから属性 ID を求めてそれを attrid に設定します。  
また、attrid が指定された場合、属性 ID からインデクスを求めて attr\_index に設定します。

属性(ID or Index)が決まると値のデータ型がきまりますので、メソッドを使い分けして下さい。

属性が CONTENTMAP, SLOTMAP につきましては、本メソッドで属性を設定した後、それぞれ、  
add\_content\_map()、add\_slot\_map() メソッドを使ってそれぞれの詳細情報を設定してください。

### 16. 4. 3. 3 add\_content\_map()

インスタンスの content\_map 属性情報にロット ID と基板 ID を追加します。

**【構文】**

```
public void set_content_map(string lotid, string substid)
```

**【引数】**

lotid  
ロット ID です。  
substid  
基板 ID です。

**【戻り値】**

なし。

**【説明】**

content\_map プロパティに 1 個の情報 lotid, substid を追加します。  
DshContentMap クラスの lotid\_list、subst\_list の配列に追加されます。

### 16. 4. 3. 4 add\_slot\_map()

インスタンスの slot\_map 属性情報をスロット ID を 1 個追加します。

**【構文】**

```
public void add_slot_map(int slotid)
```

**【引数】**

slotid  
スロット ID です。

**【戻り値】**

なし。

**【説明】**

slot\_map プロパティに slotid を 1 個だけ追加します。  
DshSlotMap クラスの slotid\_list 配列に追加されます。

#### 16. 4. 3. 5 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshCarActionPara dst )
```

**【引数】**

dst

コピー先のDshCarActionPara インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。



## 16. 5 DshContentMap クラス

S3F17 キャリアアクション送信メッセージに含まれる CONTENTMAP 属性情報を保存するためのクラスです。

### 16. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                     | 説明             |
|---|------------------------|----------------|
| 1 | public DshContentMap() | 空のインスタスを生成します。 |

### 16. 5. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                         | 説明  |
|---|----------------------------|---|
| 1 | public int count           | lotid_list, subst_list の配列に保存されている情報の数です。 |
| 2 | public string[] lotid_list | ロット ID を保存する配列です。                         |
| 3 | public string[] subst_list | 基板 ID を保存する配列です。                          |

### 16. 5. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void clear()    | lotid_list, subst_list の内容を消去します。  |
| 2 | public void add_info() | 1 個のコンテンツマップ 情報を配列に追加します。  |
| 3 | public void copy()     | 当該インスタンスの内容を他のDshContentMapクラスにコピーします。   |
| 4 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 5. 3. 1 clear()

クラス内の lotid\_list, subst\_list 配列の内容を消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

lotid\_list, subst\_list 配列の内容を消去し、空にします。  
count は =0 にします。

### 16. 5. 3. 2 add\_info()

ロット ID、基板 ID を lotid\_list, subst\_list 配列に追加します。

**【構文】**

```
public void add_info(string lotid, string substid)
```

**【引数】**

lotid  
追加するロット ID です。

substid  
追加する基板 ID です。

**【戻り値】**

なし。

**【説明】**

lotid を lotid\_list 配列に、substid を subst\_list の配列にそれぞれ追加します。  
追加した後、count + 1 します。

### 16. 5. 3. 3 copy()

当該インスタンスの内容を別の DshContentMap クラスのインスタンスにコピーします。

**【構文】**

```
public void copy(ref DshContentMap dst)
```

**【引数】**

dst  
コピー先インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンス内に保存されている count 分の lotid\_list, subst\_list 配列の情報を dst で指定されたインスタンスにコピーします。

## 16. 6 DshSlotMap クラス

S3F17 キャリアアクション送信メッセージに含まれる SLOTMAP 属性情報を保存するためのクラスです。

### 16. 6. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                  | 説明             |
|---|---------------------|----------------|
| 1 | public DshSlotMap() | 空のインスタスを生成します。 |

### 16. 6. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                       | 説明                             |
|---|--------------------------|--------------------------------|
| 1 | public int count         | slotid_list の配列に保存されている情報の数です。 |
| 2 | public int[] slotid_list | スロット ID を保存する配列です。             |

### 16. 6. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void clear()    | slotid_list の内容を消去します。   |
| 2 | public void add_info() | 1 個のスロット ID を配列に追加します。   |
| 3 | public void copy()     | 当該インスタンスの内容を他の DshSlotMap クラスにコピーします。  |
| 4 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 6. 3. 1 clear()

クラス内の slotid\_list 配列の内容を消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

slotid\_list 配列の内容を消去し、空にします。  
count は =0 にします。

### 16. 6. 3. 2 add\_info()

スロット ID を slotid\_list 配列に追加します。

**【構文】**

```
public void add_info(int slotid)
```

**【引数】**

slotid

追加するスロット ID です。

**【戻り値】**

なし。

**【説明】**

slotid を slotid\_list 配列に追加します。  
追加した後、count + 1 します。

### 16. 6. 3. 3 copy()

当該インスタンスの内容を別の DshSlotMap クラスのインスタンスにコピーします。

**【構文】**

```
public void copy(ref DshSlotMap dst)
```

**【引数】**

dst

コピー先インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンス内に保存されている count 分の slotid\_list 配列の情報を dst で指定されたインスタンスにコピーします。

## 16. 7 DshPortGroupAction クラス

S3F23 ポートグループアクション要求メッセージに含まれるアクション情報を保存するためのクラスです。

decode メソッドによって S3F23 メッセージに含まれる情報を本クラス内にデコードすることができます。

### 16. 7. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明                             |
|---|--|--------------------------------|
| 1 | <code>public DshPortGroupAction()</code>   | 空のインスタンスを生成します。                |
| 2 | <code>public DshPortGroupAction(<br/>    string action,<br/>    string groupname)</code> | インスタンスをアクション名、グループ名を指定して生成します。 |

### 16. 7. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                                    | 説明  |
|---|---------------------------------------|---|
| 1 | <code>public string action</code>     | アクション名です。                                 |
| 2 | <code>public int action_index</code>  | アクション名に対するインデックスです。                       |
| 3 | <code>public string group_name</code> | ポートグループ名です。                               |
| 4 | <code>public int count</code>         | 含まれるアクションパラメータの数です。                       |
| 5 | <code>public DshObjPara[] list</code> | アクションパラメータ情報を保存するための DshObjPara クラスの配列です。 |

### 16. 7. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void clear()    | list と sub_list 配列内の情報を消去します。  |
| 2 | public void set_info() | アクション名とグループ名を設定します。  |
| 3 | public void add_para() | list 配列に1個のアクションパラメータを追加します。   |
| 4 | public int decode()    | S3F23メッセージを当該クラスにデコードします。  |
| 5 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 7. 3. 1 clear()

クラス内に設定されているアクションパラメータを全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

クラスの list 配列内に含む全てのアクションパラメータを消去します。  
非管理メモリを使用しているパラメータ値についてはメモリを開放します。  
そして、count = 0 にします。



### 16. 7. 3. 2 set\_info()

グループ名とアクション名を設定します。

#### 【構文】

```
public void set_info(string action , string group_name)
```

#### 【引数】

action

アクション名です。

group\_name

ポートグループ名です。

#### 【戻り値】

なし。

#### 【説明】

引数に与えられたアクション名とグループ名をそれぞれ action と group\_name プロパティに設定します。

### 16. 7. 3. 3 add\_para()

インスタンスの list 配列にアクションパラメータ情報を 1 個追加します。

#### 【構文】

```
public void add_para(string pname, int format, int size, IntPtr value)
public void add_para(string pname, string value)
```

#### 【引数】

pname

パラメータ名です。

format

パラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

パラメータ値の配列サイズです。

value

設定したいパラメータ値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

list 配列に 1 個のパラメータ情報を DshObjPara のクラスに設定し、追加します。  
string で与えられた value は HSMS. ICODE\_A のフォーマットとして保存されます。  
追加した後、count を +1 します。

### 16. 7. 3. 4 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

#### 【構文】

```
public void copy( ref DshCarPortGroupAction dst )
```

#### 【引数】

dst

コピー先の DshCarPortGroupAction インスタンスです。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンスの内容 (プロパティ値) を dst に指定されるインスタンスにコピーします。

### 16. 7. 3. 5 decode()

S3F23 に含まれるポートグループアクション情報を DshPortGroupAction クラス内のプロパティにデコードします。

#### 【構文】

```
public int decode(ref DSHMSG msg)
```

#### 【引数】

msg

S3F23 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

#### 【戻り値】

| 返却値  | 意味                                    |
|------|---------------------------------------|
| 0    | 正常にデコードできた。                           |
| (-1) | デコードできなかった (S3F23 のメッセージの形式が正しくなかった。) |

#### 【説明】

msg に含まれている情報を DshPortGroupAction クラス内にデコードします。

msg S3F23

L, 3

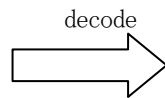
grpaction

portgrpname

L, m

param1

param2



decode

DshPortGroupAction

Class

action,

group\_name

list[]

メッセージに含まれるアクション名、グループ名とパラメータ情報がクラスのプロパティに保存されます。

正常にデコードできた場合、0 を返却します。

もし、S3F23 のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1) を返却します。

## 16. 8 DshPortAction クラス

S3F25 ポートアクション要求メッセージに含まれるアクション情報を保存するためのクラスです。

decode メソッドによって S3F25 メッセージに含まれる情報を本クラス内にデコードすることができます。

### 16. 8. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明                           |
|---|--|------------------------------|
| 1 | public DshPortAction()                           | 空のインスタンスを生成します。              |
| 2 | public DshPortAction(string action,<br>int port) | インスタンスをアクション名、ポートを指定して生成します。 |

### 16. 8. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                       | 説明  |
|---|--------------------------|---|
| 1 | public string action     | アクション名です。                                 |
| 2 | public int action_index  | アクション名に対するインデックスです。                       |
| 3 | public int port          | ポートです。                                    |
| 4 | public int count         | 含まれるアクションパラメータの数です。                       |
| 5 | public DshObjPara[] list | アクションパラメータ情報を保存するための DshObjPara クラスの配列です。 |

### 16. 8. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void clear()    | list 配列内の情報を消去します。   |
| 2 | public void set_info() | アクション名とポートを設定します。  |
| 3 | public void add_para() | list 配列に 1 個のアクションパラメータを追加します。   |
| 4 | public int decode()    | S3F25 メッセージを当該クラスにデコードします。   |
| 5 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 8. 3. 1 clear()

クラス内に設定されているアクションパラメータを全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

クラスの list 配列内に含む全てのアクションパラメータを消去します。  
非管理メモリを使用しているパラメータ値についてはメモリを開放します。  
そして、count = 0 にします。

### 16. 8. 3. 2 set\_info()

アクション名とポートを設定します。

#### 【構文】

```
public void set_info(string action, int port)
```

#### 【引数】

action

アクション名です。

port

アクション対象ポート番号です。

#### 【戻り値】

なし。

#### 【説明】

引数に与えられたアクション名、port をそれぞれ action と port プロパティに設定します。

### 16. 8. 3. 3 add\_para()

インスタンスの list 配列にアクションパラメータ情報を 1 個追加します。

#### 【構文】

```
public void add_para(string pname, int format, int size, IntPtr value)
public void add_para(string pname, string value)
```

#### 【引数】

pname

パラメータ名です。

format

パラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

パラメータ値の配列サイズ

value

設定したいパラメータ値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

list 配列に 1 個のパラメータ情報を DshObjPara のクラスに設定し、追加します。string で与えられた value は HSMS. ICODE\_A のフォーマットとして保存されます。追加した後、count を +1 します。

### 16. 8. 3. 4 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

#### 【構文】

```
public void copy( ref DshCarPortAction dst )
```

#### 【引数】

dst

コピー先の DshCarPortAction インスタンスです。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

### 16. 8. 3. 5 decode()

S3F25 に含まれるポートアクション情報を DshPortAction クラス内のプロパティにデコードします。

#### 【構文】

```
public int decode(ref DSHMSG msg)
```

#### 【引数】

msg

S3F25 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

#### 【戻り値】

| 返却値  | 意味                                    |
|------|---------------------------------------|
| 0    | 正常にデコードできた。                           |
| (-1) | デコードできなかった (S3F25 のメッセージの形式が正しくなかった。) |

#### 【説明】

msg に含まれている情報を DshPortAction クラス内にデコードします。

msg S3F25

L, 3

portaction

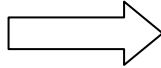
port

L, m

paraname1

paraval

decode



DshPortAction  
Class

action,  
port  
list[]

メッセージに含まれるアクション名、ポート番号とパラメータ情報がクラスのプロパティに保存されます。

正常にデコードできた場合、0 を返却します。

もし、S3F25 のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1) を返却します。



## 16. 9 DshPortAccess クラス

S3F27 ポートアクセス変更要求メッセージに含まれるアクセスモード変更情報を保存するためのクラスです。

decode メソッドによって S3F27 メッセージに含まれる情報を本クラス内にデコードすることができます。

### 16. 9. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                             | 説明                  |
|---|--------------------------------|---------------------|
| 1 | public DshPortAccess()         | 空のインスタスを生成します。      |
| 2 | public DshPortAccess(int mode) | インスタスをモード指定して生成します。 |

### 16. 9. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                | 説明                 |
|---|-------------------|--------------------|
| 1 | public int mode   | アクセスモードです。         |
| 2 | public int count  | 対象となるポートの数です。      |
| 3 | public int[] list | ポート番号を保存するための配列です。 |

### 16. 9. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void clear()    | port_list 配列内の情報を消去します。  |
| 2 | public void set_info() | モードを設定します。   |
| 3 | public void add_port() | list 配列に1個のポート番号を追加します。  |
| 4 | public int decode()    | S3F27メッセージを当該クラスにデコードします。  |
| 5 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 9. 3. 1 clear()

port\_list に設定されているポートを全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

クラスの list 配列内に含む全てのポートを消去します。  
そして、count = 0 にします。

### 16. 9. 3. 2 set\_info()

アクセスモードを設定します。

**【構文】**

```
public void set_info(int mode)
```

**【引数】**

mode

アクセスモードです。

**【戻り値】**

なし。

**【説明】**

引数に与えられたアクセスモードをプロパティに設定します。

### 16. 9. 3. 3 add\_port()

インスタンスの port\_list 配列にポート番号を 1 個追加します。

**【構文】**

```
public void add_port(int port)
```

**【引数】**

port

port\_list 配列に追加するポート番号です。

**【戻り値】**

なし。

**【説明】**

list 配列に 1 個のポートを追加します。

追加した後、count を +1 します。

#### 16. 9. 3. 4 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshCarPortAccess dst )
```

**【引数】**

dst

コピー先のDshCarPortAccess インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

### 16. 9. 3. 5 decode()

S3F27 に含まれるポートアクセス変更情報を DshPortAccess クラス内のプロパティにデコードします。

#### 【構文】

```
public int decode(ref DSHMSG msg)
```

#### 【引数】

msg

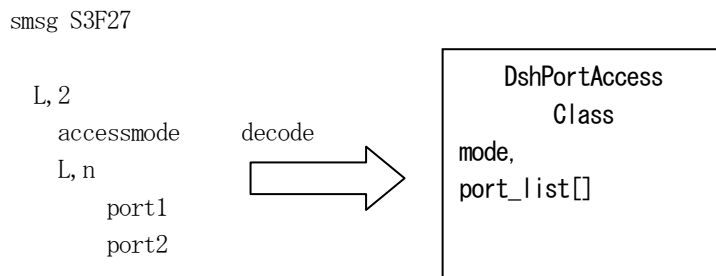
S3F27 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

#### 【戻り値】

| 返却値  | 意味                                    |
|------|---------------------------------------|
| 0    | 正常にデコードできた。                           |
| (-1) | デコードできなかった (S3F27 のメッセージの形式が正しくなかった。) |

#### 【説明】

msg に含まれている情報を DshPortAccess クラス内にデコードします。



メッセージに含まれるアクセスモード、ポートがクラスのプロパティに保存されます。

正常にデコードできた場合、0 を返却します。

もし、S3F27 のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1) を返却します。

## 16. 10 DshRCmdRsp クラス

S2F41, S2F49 の応答メッセージ S2F42, S2F50 の情報を保存するためのクラスです。

### 16. 10. 1 コンストラクタ

|   | 名前                           | 説明                      |
|---|------------------------------|-------------------------|
| 1 | public DshRCmdRsp ()         | 空のインスタスを生成します。          |
| 2 | public DshRCmdRsp(int hcack) | hcack を指定してインスタスを生成します。 |

### 16. 10. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                     | 説明  |
|---|------------------------|---|
| 1 | public int hcack       | S2F42, S2F50 の hcack です。  |
| 2 | public int count       | 3 の list 配列に保存されている DshCpAck クラスのインスタスの数です。                                     |
| 3 | public DshCpAck[] list | コマンドパラメータ単位の応答情報を保存する DshCpAck クラスの配列リストです。<br>DshCpAck クラスについては 16. 11 で説明します。 |

### 16. 10. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                    | 説明   |
|---|-----------------------|--|
| 1 | public void clear()   | list 配列の内容を空にします。  |
| 2 | public void add_ack() | 1 個のコメントパラメータ応答情報を追加します。   |
| 3 | public void copy()    | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 4 | public void Dispose() | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 10. 3. 1 clear()

control\_list 配列の内容を空にします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

list 配列の内容を空にして、count = 0 にします。

### 16. 10. 3. 2 add\_ack()

コマンドパラメータ応答情報を 1 個追加します。

**【構文】**

```
public void add_ack(string cp_name, int cp_ack)
```

**【引数】**

cp\_name

コマンドパラメータ名です。

cp\_ack

コマンドパラメータに対する ack です。

**【戻り値】**

なし。

**【説明】**

list 配列に 1 個のコマンドパラメータ応答情報を追加します。  
追加した後、count + 1 します。

### 16. 10. 3. 3 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy(ref DshRCmdRsp dst)
```

**【引数】**

dst

コピー先の DshRCmdRsp クラスのインスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。



## 16. 11 DshCpAck クラス

S2F41, S2F49 の応答メッセージ S2F42, S2F50 の情報を保存するためのクラスです。

ホストコマンドのコマンドパラメータ単位に対する応答情報を保存します。

### 16. 11. 1 コンストラクタ

|   | 名前  | 説明                                |
|---|---|-----------------------------------|
| 1 | public DshCpAck(<br>string name,<br>int cp_ack) | コマンドパラメータ名と ACK を指定してインスタスを生成します。 |

### 16. 11. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                    | 説明                         |
|---|-----------------------|----------------------------|
| 1 | public string cp_name | パラメータ名です。(cpname)          |
| 2 | public int cp_ack     | cpname に対する ack です。(cpack) |

### 16. 11. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                    | 説明   |
|---|-----------------------|--|
| 1 | public void Dispose() | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

## 16. 12 DshCarActionRsp クラス

キャリアアクション関連 SECS-II メッセージの応答情報を保存するためのクラスです。  
S3F18 メッセージが対象メッセージです。

### 16. 12. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                              | 説明                            |
|---|---------------------------------|-------------------------------|
| 1 | public DshCarActionRsp()        | 空のインスタスを生成します。                |
| 2 | public DshCarActionRsp(int ack) | ack (caack) を指定してインスタスを生成します。 |

### 16. 12. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                        | 説明                               |
|---|---------------------------|----------------------------------|
| 1 | public int caack          | 応答メッセージの設定する caack です。           |
| 2 | public int count          | 付属するエラー情報の数です。                   |
| 3 | public DshObjError[] list | エラー情報を保存する配列です。<br>Vol-1 19.1 参照 |

### 16. 12. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                      | 説明   |
|---|-------------------------|--|
| 1 | public void clear()     | エラーリスト list の配列を空にします。   |
| 2 | public void set_count() | エラーリスト list 配列のサイズを設定します。  |
| 3 | public void add_error() | エラー情報(DshObjError)を1個 list に追加します。   |
| 4 | public void Dispose()   | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 12. 3. 1 clear()

list に設定されているエラー情報 (DshObjError の内容) を全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

list に設定されているエラー情報 (DshObjError の内容) を全て消去します。  
そして、count = 0 にします。

### 16. 12. 3. 2 set\_count()

list 配列に、指定されたエラー情報分のサイズの配列を作成します。

#### 【構文】

```
public void set_count(int count)
```

#### 【引数】

count

配列サイズです。

#### 【戻り値】

なし。

#### 【説明】

list の配列を引数に与えられたサイズにします。またプロパティの count に値を設定します。

本メソッドは、IntPtr で指された配列リストに格納されている複数のエラー情報をまとめて list 配列に設定する際に使用します。エラー情報設定に使用する関数は、DshObjError.copy\_err\_list() です。

list にエラー情報を設定したい場合は、通常、次に説明する add\_error() メソッドを使ってください。

### 16. 12. 3. 3 add\_err()

インスタンスの list 配列にエラー情報を 1 個追加します。

#### 【構文】

```
public void add_error(int code, string text)
```

#### 【引数】

code

エラーコードです。

text

エラーテキストです。

#### 【戻り値】

なし。

#### 【説明】

list 配列に、1 個のエラー情報を DshObjError のクラスに設定し、追加します。

追加した後、count を +1 します。

## 16. 13 DshPortActionRsp クラス

ポートアクション関連 SECS-II メッセージの応答情報を保存するためのクラスです。  
S3F18 メッセージが対象メッセージです。

### 16. 13. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                               | 説明                             |
|---|----------------------------------|--------------------------------|
| 1 | public DshPortActionRsp()        | 空のインスタンスを生成します。                |
| 2 | public DshPortActionRsp(int ack) | ack (caack) を指定してインスタンスを生成します。 |

### 16. 13. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                        | 説明                               |
|---|---------------------------|----------------------------------|
| 1 | public int caack          | 応答メッセージの設定する caack です。           |
| 2 | public int count          | 付属するエラー情報の数です。                   |
| 3 | public DshObjError[] list | エラー情報を保存する配列です。<br>Vol-1 19.1 参照 |

### 16. 13. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                      | 説明  |
|---|-------------------------|---|
| 1 | public void clear()     | エラーリスト list の配列を空にします。  |
| 2 | public void set_count() | エラーリスト list 配列のサイズを設定します。   |
| 3 | public void add_error() | エラー情報(DshObjError)を1個 list に追加します。  |
| 4 | public void Dispose()   | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをパーティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 13. 3. 1 clear()

list に設定されているエラー情報 (DshObjError の内容) を全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

list に設定されているエラー情報 (DshObjError の内容) を全て消去します。  
そして、count = 0 にします。

### 16. 13. 3. 2 set\_count()

list 配列に、指定されたエラー情報分のサイズの配列を作成します。

#### 【構文】

```
public void set_count(int count)
```

#### 【引数】

count

配列サイズです。

#### 【戻り値】

なし。

#### 【説明】

list の配列を引数に与えられたサイズにします。またプロパティの count に値を設定します。本メソッドは、IntPtr で指された配列リストに格納されている複数のエラー情報をまとめて list 配列に設定する際に使用します。エラー情報設定に使用する関数は、DshObjError.copy\_err\_list() です。

list にエラー情報を設定したい場合は、通常、次に説明する add\_err() メソッドを使ってください。

### 16. 13. 3. 3 add\_err()

インスタンスの list 配列にエラー情報を 1 個追加します。

#### 【構文】

```
public void add_error(int code, string text)
```

#### 【引数】

code

エラーコードです。

text

エラーテキストです。

#### 【戻り値】

なし。

#### 【説明】

list 配列に、1 個のエラー情報を DshObjError のクラスに設定し、追加します。追加した後、count を +1 します。

## 16. 14 DshAccessError クラス

エラー情報を保存するためのクラスです。

### 16. 14. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明                              |
|---|--|---------------------------------|
| 1 | public DshAccessError()                                | 空のインスタスを生成します。                  |
| 2 | public DshAccessError(int port, int code, string text) | ポート番号、エラーコードとエラーテキストを指定して生成します。 |

### 16. 14. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                    | 説明         |
|---|-----------------------|------------|
| 1 | public int port       | ポート番号です。   |
| 2 | public int errcode    | エラーコードです。  |
| 3 | public string errtext | エラーテキストです。 |

### 16. 14. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                    | 説明   |
|---|-----------------------|--|
| 1 | public void set_err() | value に使用されているメモリを開放します。   |
| 2 | public void Dispose() | クラス内部で使用された資源(メモリ)をシステムに返却します。他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。14. 1. 3. 13 と同様です。そちらを参照ください。 |



### 16. 14. 3. 1 set\_err ()

ポート番号、エラーコードとエラーテキストを設定します。

#### 【構文】

```
public void set_obj_err(int port, int code, string text)
```

#### 【引数】

code

エラーコードです。

text

エラーテキストです。

#### 【戻り値】

なし。

#### 【説明】

ポート番号、エラーコードとエラーテキストを設定します。

## 16. 15 DshPortAccessRsp クラス

ポートアクセス変更関連 SECS-II メッセージの応答情報を保存するためのクラスです。  
S3F18 メッセージが対象メッセージです。

### 16. 15. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                               | 説明                             |
|---|----------------------------------|--------------------------------|
| 1 | public DshPortAccessRsp()        | 空のインスタンスを生成します。                |
| 2 | public DshPortAccessRsp(int ack) | ack (caack) を指定してインスタンスを生成します。 |

### 16. 15. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                           | 説明                                |
|---|------------------------------|-----------------------------------|
| 1 | public int caack             | 応答メッセージの設定する caack です。            |
| 2 | public int count             | 付属するエラー情報の数です。                    |
| 3 | public DshAccessError[] list | エラー情報を保存する配列です。<br>Vol-1 16.13 参照 |

### 16. 15. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                      | 説明   |
|---|-------------------------|--|
| 1 | public void clear()     | エラーリスト list の配列を空にします。   |
| 2 | public void add_error() | エラー情報(DshAccessError)を1個 list に追加します。  |
| 3 | public void add_error() | エラー情報(DshAccessError)を1個 list に追加します。  |
| 4 | public void Dispose()   | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 16. 15. 3. 1 clear()

list に設定されているエラー情報 (DshAccessError の内容) を全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

list に設定されているエラー情報 (DshAccessError の内容) を全て消去します。  
そして、count = 0 にします。

### 16. 15. 3. 2 add\_err()

インスタンスの list 配列にエラー情報を 1 個追加します。

#### 【構文】

```
public void add_error(int port, int code, string text)
```

#### 【引数】

port

ポート番号です。

code

エラーコードです。

text

エラーテキストです。

#### 【戻り値】

なし。

#### 【説明】

list 配列に、1 個のエラー情報を DshAccessError のクラスに設定し、追加します。  
追加した後、count を +1 します。

## 17. 端末表示メッセージ関連クラス

端末表示メッセージ関連情報を保存するクラスです。

関連クラスの一覧を次表に示します。

|   | クラス名            | 用途  |
|---|-----------------|---|
| 1 | DshTermMsg      | 端末メッセージ保存用クラスです。<br>S10F1, S10F3 に使用されます。 |
| 2 | DshTermMultiMsg | 端末複数メッセージ保存用クラスです。<br>S10F5 に使用されます       |

通信関連クラスとして、以下のものがあります。 これらについては、Vol-2 で説明します。

|                  |
|------------------|
| DshS10F1Send     |
| DshS10F4Response |
| DshS10F6Response |

## 17. 1 DshTermMsg クラス

端末メッセージ情報のためのクラスであり、S10F1, S10F3 メッセージの送受信時に使用されます。

### 17. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明                             |
|---|--|--------------------------------|
| 1 | public DshTermMsg()                        | 空のインスタスを生成します。                 |
| 2 | public DshTermMsg(int tid,<br>string text) | 端末 ID と表示テキストを指定してインスタスを生成します。 |

### 17. 1. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                 | 説明            |
|---|--------------------|---------------|
| 1 | public int tid     | 端末 ID です。     |
| 2 | public string_text | 端末に表示する文字列です。 |

### 17. 1. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void set_info() | 端末 ID と表示文字列を設定します。  |
| 2 | public int decode()    | S10F1 または S10F3 メッセージ の情報を当該クラスに保存します。   |
| 3 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 17. 1. 3. 1 set\_info()

端末 ID と表示文字列 (1 行) をプロパティ tid, text に設定します。

**【構文】**

```
public void set_info(int tid, string text)
```

**【引数】**

tid

端末 ID です。

text

表示文字列です。

**【戻り値】**

なし。

**【説明】**

端末 ID と表示文字列 (1 行) をプロパティ tid, text に設定します。

### 17. 1. 3. 2 decode()

S10F1 または S10F3 に含まれる情報を DshTermMsg クラス内のプロパティにデコードします。

#### 【構文】

```
public int decode(ref DSHMSG msg)
```

#### 【引数】

msg

S10F1 または S10F3 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。

DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。

ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

#### 【戻り値】

| 返却値  | 意味   |
|------|--|
| 0    | 正常にデコードできた。  |
| (-1) | デコードできなかった<br>(S10F1 または S10F3 のメッセージの形式が正しくなかった。) |

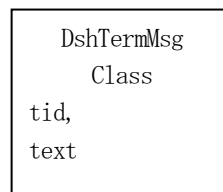
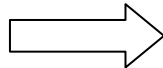
#### 【説明】

msg に含まれているレポート情報を DshTermMsg クラス内にデコードします。

msg S10F1 または S10F3

L, 2  
tid  
text

decode



メッセージに含まれる端末表示情報は、クラスのプロパティに保存されます。

正常にデコードできた場合、0 を返却します。

もし、S10F1 または S10F3 のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1) を返却します。



## 17. 2 DshTermMultiMsg クラス

複数行の表示文字列を含む端末メッセージ情報のためのクラスであり、S10F5 メッセージの送受信時に使用されます。

### 17. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明                      |
|---|--|-------------------------|
| 1 | <code>public DshTermMultiMsg()</code>        | 空のインスタスを生成します。          |
| 2 | <code>public DshTermMultiMsg(int tid)</code> | 端末 ID を指定してインスタスを生成します。 |

### 17. 2. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                                | 説明                               |
|---|-----------------------------------|----------------------------------|
| 1 | <code>public int tid</code>       | 端末 ID です。                        |
| 1 | <code>public int count</code>     | 設定されている文字列の数です。                  |
| 2 | <code>public string[] list</code> | string クラスの配列であり、表示文字列を保存する配列です。 |

## 17. 2. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                    | 説明  |
|---|-----------------------|---|
| 1 | public void set_tid() | 端末 ID を設定します。   |
| 2 | public void clear()   | list 配列を空にします。  |
| 3 | public void add()     | list 配列に 1 個の文字列を加えます。  |
| 4 | public void copy()    | 当該インスタンスの内容を別のインスタンスにコピーします。  |
| 5 | public int decode()   | S10F5 メッセージの情報を当該クラスに保存します。   |
| 6 | public void Dispose() | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14.1.3.13 と同様です。そちらを参照ください。 |

### 17. 2. 3. 1 set\_tid()

端末 ID をプロパティ tid に設定します。

**【構文】**

```
public void set_tid(int tid)
```

**【引数】**

tid

端末 ID です。

**【戻り値】**

なし。

**【説明】**

端末 ID をプロパティ tid に設定します。

### 17. 2. 3. 2 clear()

list 配列を空にします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

list 配列リストを空にします。そして count = 0 にします。

### 17. 2. 3. 3 add()

インスタンスの list 配列に文字列を 1 個追加します。

**【構文】**

```
public void add(string data)
```

**【引数】**

data

追加したい文字列です。

**【戻り値】**

なし。

**【説明】**

list 配列に 1 個のデータアイテムを追加します。  
そして、count + 1 します。

### 17. 2. 3. 4 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshStrList dst )
```

**【引数】**

dst  
コピー先のDshStrList インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

### 17. 2. 3. 5 decode()

S10F5 に含まれる情報を DshTermMultiMsg クラス内のプロパティにデコードします。

#### 【構文】

```
public int decode(ref DSHMSG msg)
```

#### 【引数】

msg

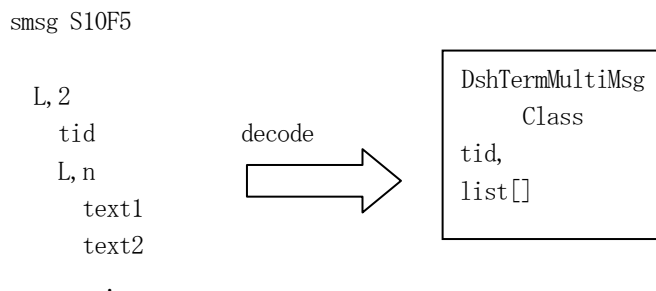
S10F5 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

#### 【戻り値】

| 返却値  | 意味                                       |
|------|--|
| 0    | 正常にデコードできた。                              |
| (-1) | デコードできなかった<br>(S10F5 のメッセージの形式が正しくなかった。) |

#### 【説明】

msg に含まれているレポート情報を DshTermMultiMsg クラス内にデコードします。



メッセージに含まれる端末表示情報は、クラスのプロパティに保存されます。

正常にデコードできた場合、0 を返却します。

もし、S10F5 のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1) を返却します。

## 18. オブジェクト属性、パラメータ情報関連クラス

本章では、メッセージに付属するオブジェクトの属性(attribute)、パラメータ情報を保存するためのクラスについて説明します。

関連クラスの一覧を次表に示します。

|   | クラス名            | 用途  |
|---|-----------------|---|
| 1 | DshObjPara      | オブジェクト関連クラスのパラメータ(属性)情報を保存するためのクラスです。<br>S2F41, S2F49, S3F17, S3F23, S3F25, S14F9, 11, S15F13 などのメッセージが含んでいる情報を保存します。 |
| 2 | DshObjSubPara   | DshObjPara の情報が中で多重化していた際に、そのサブパラメータを保存するためのクラスです。<br>S2F41 のメッセージ情報保存に使用されます。  |
| 3 | DshObjParaList  | DshObjPara の配列リストのクラスです。  |
| 4 | DshDataItem     | SECS-II のデータアイテムの情報を 1 個保存するためのクラスです。   |
| 5 | DshDataItemList | DshDataItem の配列クラスです。   |
| 6 | DshStrList      | 文字列(string)の配列クラスです。  |

## 18. 1 DshObjPara クラス

オブジェクトのパラメータ（属性）情報を保存するためのクラスです。

（注） コントロールジョブ属性情報の保存には、DshCjAttr クラスを使います。

### 18. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前  | 説明  |
|---|---|---|
| 1 | public DshObjPara()   | 空のインスタンスを生成します。   |
| 2 | public DshObjPara(string name,<br>int format,<br>int size,<br>IntPtr value) | 引数としてパラメータ ID, フォーマット、size, 値が格納されているポインタ(IntPtr)が指定されます。 |
| 3 | public DshObjPara(string name,<br>string value)                             | 引数としてパラメータ ID, と文字列値が指定されます。                              |

### 18. 1. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                  | 説明   |
|---|---------------------|--|
| 1 | public string name  | パラメータ名です。                                      |
| 2 | public int format   | パラメータ値のフォーマットです。<br>(HSMS. ICODE_U1 など)        |
| 3 | public int size     | パラメータ値の配列サイズ（データ数、基本的に ICODE_A 以外は size=1 です）。 |
| 4 | public IntPtr value | パラメータ値を格納する非管理型のポインタです。                        |

### 18. 1. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void free()     | value に使用されているメモリを開放します。   |
| 2 | public void set_para() | パラメータ情報を設定します。   |
| 3 | public void copy()     | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 4 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 18. 1. 3. 1 free()

インスタンスの value に使用されている非管理メモリを開放します。

**【構文】**

```
public void free()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

パラメータ値 value に使用されている非管理メモリを開放します。



### 18. 1. 3. 2 set\_para ()

パラメータ名とパラメータ値を設定します。

#### 【構文】

```
public void set_para(string name, int format, int size, IntPtr value)
public void set_para(string name, string value)
```

#### 【引数】

name

パラメータ名です。

format

パラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

パラメータ値の配列サイズです。

value

設定したいパラメータ値が格納されているポインタです。

#### 【戻り値】

なし。

#### 【説明】

パラメータ情報を DshObjPara クラスに設定します。

value が string で与えられるメソッドの場合、HSMS. ICODE\_A のフォーマットで設定します。

### 18. 1. 3. 3 copy ()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

#### 【構文】

```
public void copy( ref DshObjPara dst )
```

#### 【引数】

dst

コピー先の DshObjPara インスタンスです。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

## 18. 2 DshObjSubPara クラス

オブジェクトのサブパラメータ（属性）情報を保存するためのクラスです。

DshObjPara クラスのパラメータ値が ICODE\_L(リスト構造)のとき、ネスティングした複数のパラメータ値を保存するためのクラスです。

### 18. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                      | 説明              |
|---|-------------------------|-----------------|
| 1 | public DshObjSubPara () | 空のインスタンスを生成します。 |

### 18. 2. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                        | 説明                                |
|---|---------------------------|-----------------------------------|
| 1 | public int count          | パラメータ値の数です。                       |
| 2 | public DshDataItem[] list | パラメータ値を保存する DshDataItem クラスの配列です。 |

### 18. 2. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明  |
|---|------------------------|---|
| 1 | public void free ()    | value に使用されているメモリを開放します。  |
| 2 | public void add ()     | list 配列にデータを1個追加します。  |
| 3 | public void copy ()    | 当該インスタンスの内容を別のインスタンスにコピーします。  |
| 4 | public void Dispose () | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose () も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

### 18. 2. 3. 1 clear()

クラス内の list 配列に設定されているデータを全て消去します。

**【構文】**

```
public void clear ()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

クラスの list 配列内に含む全てのデータを消去します。  
非管理メモリを使用しているパラメータ値についてはメモリを開放します。  
そして、count = 0にします。

### 18. 2. 3. 2 add()

パラメータ値を配列に 1 個追加します。

**【構文】**

```
public void add(int format, int size, IntPtr value)  
public void add(string value)  
public void add(DshDataItem data)
```

**【引数】**

format

パラメータ値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size

パラメータ値の配列サイズです。

value

設定したいパラメータ値が格納されているポインタです。

data

パラメータ値が保存されている DshDataItem クラスのインスタンスです。

**【戻り値】**

なし。

**【説明】**

パラメータ値を DshObjSubPara クラスの list 配列に追加します。  
value が string で与えられるメソッドの場合、HSMS. ICODE\_A のフォーマットで設定します。

### 18. 2. 3. 3 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshObjSubPara dst )
```

**【引数】**

dst  
コピー先の DshObjSubPara インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

### 18. 3 DshDataItem クラス

1 個の SECS データアイテムのデータを保存するためのクラスです。

#### 18. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前  | 説明                                |
|---|---|-----------------------------------|
| 1 | public DshDataItem()                                      | 空のインスタスを生成します。                    |
| 2 | public DshDataItem(int format,<br>int size, IntPtr value) | データのフォーマット、サイズ、値を指定してインスタスを生成します。 |

#### 18. 3. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                  | 説明  |
|---|---------------------|---|
| 1 | public int format   | データアイテムのフォーマットです。<br>HSMS. ICODE_U1 など  |
| 2 | public int size     | データアイテムの配列サイズです。<br>HSMS. ICODE_A, J では、文字列長になります。<br>HSMS. ICODE_L では、リストサイズになります。<br>その他のフォーマットでは、1 です。 |
| 3 | public IntPtr value | データ値が保存されているメモリのポインタです。<br>非管理メモリです。  |

### 18. 3. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void free()     | value に使用されているメモリを開放します。   |
| 2 | public void set_data() | データを設定します。   |
| 3 | public void copy()     | 当該インスタンスの内容を別のインスタンスにコピーします。   |
| 4 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 18. 3. 3. 1 free()

クラス内の value に使用されている非管理メモリを開放します。

**【構文】**

```
public void free()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

value に使用されている非管理メモリを開放します。

value を IntPtr.Zero にします。

### 18. 3. 3. 2 set\_data()

パラメータ値を配列に1個追加します。

#### 【構文】

```
public void set_data(int format, int size, IntPtr value)
public void set_data( string value)
```

#### 【引数】

format  
データのフォーマットです。(HSMS. ICODE\_U1 など)

size  
データの配列サイズ

value  
設定したいメータが格納されているポインタです。  
string の場合は文字列です。

#### 【戻り値】

なし。

#### 【説明】

データを設定します。  
value が string で与えられるメソッドの場合、HSMS. ICODE\_A のフォーマットで設定します。

### 18. 3. 3. 3 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshDataItem dst )
```

**【引数】**

dst  
コピー先のDshDataItem インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。



## 18. 4 DshDataItemList クラス

複数個のデータアイテムをリストで保存するための DshDataItem クラスの配列です。

### 18. 4. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                       | 説明                                |
|---|--------------------------|-----------------------------------|
| 1 | public DshDataItemList() | 空のインスタスを生成します。<br>配列サイズは=0 になります。 |

### 18. 4. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                             | 説明                    |
|---|--------------------------------|-----------------------|
| 1 | public int count               | 設定されているデータアイテム値の数です。  |
| 2 | public DshDataItem[] data_list | DshDataItem クラスの配列です。 |

### 18. 4. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                     | 説明   |
|---|------------------------|--|
| 1 | public void clear()    | data_list 配列を空にします。  |
| 2 | public void add_data() | data_list 配列に 1 個のデータアイテム値を加えます。   |
| 3 | public void Dispose()  | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

#### 18. 4. 3. 1 clear()

インスタンスの data\_list 配列の中に含まれるデータアイテムの value に使用されている非管理メモリを開放し、data\_list リストを空にします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

data\_list 配列のなかのデータアイテム値 value に使用されている非管理メモリを開放し、リストを空にします。そして count = 0 にします。

### 18. 4. 3. 2 add\_data()

インスタンスの data\_list 配列にデータアイテム値情報を 1 個追加します。

#### 【構文】

```
public void add_data(int format, int size, IntPtr value)
public void add_data(ref DshDataItem data)
```

#### 【引数】

format  
データアイテム値のデータフォーマットです。(HSMS. ICODE\_U1 など)

size  
データアイテム値の配列サイズです。

value  
設定したいデータアイテム値が格納されているポインタです。

data  
データアイテムが保存されている DshDataItem クラスのインスタンスです。

#### 【戻り値】

なし。

#### 【説明】

data\_list 配列に 1 個のデータアイテムを追加します。  
value, data に使用されている IntPtr のメモリは新規に非管理メモリを取得して保存します。  
そして、count + 1 します。

## 18. 5 DshStrList クラス

複数個の文字列を保存するための string クラスの配列です。

### 18. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前                  | 説明                              |
|---|---------------------|---------------------------------|
| 1 | public DshStrList() | 空のインスタスを生成します。<br>配列サイズは0になります。 |

### 18. 5. 2 プロパティ

プロパティ一覧表を示します。

|   | 名前                   | 説明               |
|---|----------------------|------------------|
| 1 | public int count     | 設定されている文字列の数です。  |
| 2 | public string[] list | string クラスの配列です。 |

### 18. 5. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                    | 説明   |
|---|-----------------------|--|
| 1 | public void clear()   | list 配列を空にします。   |
| 2 | public void add()     | list 配列に1個の文字列を加えます。   |
| 3 | public void copy()    | 当該インスタスの内容を別のインスタスにコピーします。   |
| 4 | public void Dispose() | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>14. 1. 3. 13 と同様です。そちらを参照ください。 |

### 18. 5. 3. 1 clear()

list 配列を空にします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

list 配列リストを空にします。そして count = 0 にします。

### 18. 5. 3. 2 add()

インスタンスの list 配列に文字列を 1 個追加します。

**【構文】**

```
public void add(string data)
```

**【引数】**

data

追加したい文字列です。

**【戻り値】**

なし。

**【説明】**

list 配列に 1 個のデータアイテムを追加します。  
そして、count + 1 します。

### 18. 5. 3. 3 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

**【構文】**

```
public void copy( ref DshStrList dst )
```

**【引数】**

dst  
コピー先のDshStrList インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

## 19. エラー情報関連クラス

本章では、応答メッセージに使用されるエラー情報を保存するクラスについて説明します。

関連クラスの一覧を次表に示します。

|   | クラス名        | 用途  |
|---|-------------|---|
| 1 | DshObjError | オブジェクト関連メッセージに対する応答メッセージに含まれるエラー情報を保存するためのクラスです。<br>S3F18, S14F10, S15F14, S16F12 などの応答メッセージが含んでいる情報を保存します。 |

## 19. 1 DshObjError クラス

エラー情報を保存するためのクラスです。

### 19. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前  | 説明                                   |
|---|---|--------------------------------------|
| 1 | public DshObjError()                      | 空のインスタスを生成します。                       |
| 2 | public DshObjError(int code, string text) | エラーコードとエラーテキストを指定して生成します。            |
| 3 | public DshObjError(int code, IntPtr text) | エラーコードと IntPtr で与えられるテキストを指定して生成します。 |

### 19. 1. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                    | 説明         |
|---|-----------------------|------------|
| 1 | public int errcode    | エラーコードです。  |
| 2 | public string errtext | エラーテキストです。 |

### 19. 1. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前                                 | 説明  |
|---|------------------------------------|---|
| 1 | public void set_obj_err()          | エラー情報を設定します。  |
| 2 | public void copy()                 | 当該インスタスの内容を別のインスタスにコピーします。  |
| 3 | public static void copy_err_list() | IntPtr で示されるオブジェクトエラー情報をブロックコピーします。   |
| 4 | public void Dispose()              | クラス内部で使用された資源(メモリ)をシステムに返却します。他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。14.1.3.13 と同様です。そちらを参照ください。 |



### 19. 1. 3. 1 set\_obj\_err ()

エラーコードとエラーテキストを設定します。

#### 【構文】

```
public void set_obj_err(int code, string text)
public void set_obj_err(int code, IntPtr text)
```

#### 【引数】

code  
エラーコードです。

text  
エラーテキストです。

#### 【戻り値】

なし。

#### 【説明】

エラーコードとエラーテキストを設定します。  
IntPtr で与えられた text は string に変換して設定します。

### 19. 1. 3. 2 copy ()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

#### 【構文】

```
public void copy( ref DshObjError dst )
```

#### 【引数】

dst  
コピー先の DshObjError インスタンスです。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

### 19. 1. 3. 3 copy\_err\_list()

IntPtr のポインタで渡される 1 個以上のオブジェクトエラー情報を DshObjError クラスの配列にブロックでコピーします。

#### 【構文】

```
public static void copy_err_list(IntPtr err_ptr_list, ref DshObjError[] list, int count)
```

#### 【引数】

err\_ptr\_list

1 個以上のオブジェクトエラー情報が格納されている構造体のポインタ配列です。

TERR\_INFO 構造体のポインタが格納されている配列のポインタです。

#### 【戻り値】

なし。

#### 【説明】

IntPtr のポインタで渡される 1 個以上のオブジェクトエラー情報を DshObjError クラスの配列にブロックでコピーします。

本クラス・ライブラリのユーザは、ほとんど使用することがないメソッドです。

本メソッドは、static 宣言されていますので、DshObjError クラスのインスタンスを生成しないで使用することができます。( DshObjError.copy\_err\_list(...) のように)

## 20. データアイテム関連、エラー情報コピークラス

SECS-IIメッセージに使用するデータアイテムの形式変換などを行う以下のクラスについて説明します。

|   | クラス名   | 用途  |
|---|--------|---|
| 1 | DshLib | データアイテムの形式変換などを行うためのクラスです。<br>データアイテム値 から文字列への変換<br>文字列からデータアイテム値への変換 |
|   |        |   |
|   |        |   |
|   |        |   |
|   |        |   |

### 20. 1 DshLib クラス

ここで述べる本クラスに属しているメソッドは全て static 関数になっています。

文字列からデータアイテムへの変換、データアイテム値から文字列への変換するためのメソッド（関数）が準備されています。本クラスに属する関数は、全て static で宣言されていますので、特にDshLib のインスタンスを生成しないで使用することができます。

#### 20. 1. 1 コンストラクタ

ありません。

#### 20. 1. 2 プロパティ

ありません。

## 20. 1. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前  | 説明  |
|---|---|---|
| 1 | public static string<br>DshPtrToFmtDataString() | データアイテムのデータを文字列に変換します。                                    |
| 2 | public static IntPtr<br>StringToFormatData()    | 文字列データを指定フォーマットのデータに変換し、非管理メモリに格納してその IntPtr のポインタを返却します。 |
| 3 | public static int DshPtrFmtDataInt()            | データフォーマットが 32 ビット整数で表現できるフォーマットの場合、int 型の値に変換します。         |
| 4 | public static IntPtr PtrToFormatData()          | IntPtr で与えられるデータアイテムの値を、別の非管理メモリに格納し、そのポインタを返します。         |
| 5 | public static IntPtr DataToPtr()                | 数値データをデータフォーマットにあわせて IntPtr メモリに格納し、それを返却します。             |
| 6 | public static void<br>StartPerformanceTimer()   | パフォーマンスタイマーを開始します。<br>処理時間計測の開始です。                        |
| 7 | public static float<br>StopPerformanceTimer()   | パフォーマンスタイマーを停止します。<br>処理時間を取得できます。                        |
| 8 | public static void Beep()                       | ビーブ音を鳴らします。   |
| 9 | public static string DshPtrToString()           | HSMS.D_GetItem() で得られた ICODE_A の情報を string 型に変換するための関数です。 |

### 20. 1. 3. 1 DshPtrToFmtDataString()

IntPtr で与えられるデータアイテムの値を文字列に変換します。

#### 【構文】

```
public static string DshPtrToFmtDataString(IntPtr ptr, int fmt, int size)
```

#### 【引数】

ptr

fmt で指定されたデータが格納されているポインタです。

fmt

ptr に格納されている値のフォーマットです。(HSMS.ITEM\_U1 など)

size

ptr に格納されているデータの数です。(HSMS.ICODE\_L のときだけ使用します。)

#### 【戻り値】

変換された文字列を返却します。

#### 【説明】

全データアイテムコードに対する変換を行います。

以下のように変換処理を行います。

- (1) fmt が ICODE\_L の場合は、size の値を文字列に変換します。
- (2) その他のフォーマットの場合、もし、ptr が IntPtr.Zero の場合、文字列 ” (null)” を返却します。
- (3) ICODE\_A の場合は、そのまま、ansi コードの文字列に変換します。
- (4) ICODE\_A 以外のものについて size=0 の場合、0 長文字列 ” ” を返却します。  
size > 0 であれば、先頭のデータ 1 個だけをフォーマットにあわせて文字列に変換します。

## 20. 1. 3. 2 StringToFormatData()

文字列で表現されているデータを指定フォーマットの値に変換し、それを確保した非管理メモリに格納し、その IntPtr のポインタを返却します。

### 【構文】

```
public static IntPtr StringToFormatData(string str, int fmt)
```

### 【引数】

str

データが表現されている文字列です。

fmt

変換したいデータのフォーマットです。(HSMS, ITEM\_U1 など)

### 【戻り値】

文字列から変換された非管理メモリのポインタを返却します。

### 【説明】

全データアイテムコードに対する変換を行います。

以下のように変換処理を行います。

- (1) str が="" (null) で、ICODE\_A, ICODE\_J 以外のフォーマットであれば、str="0" にして変換します。
- (2) 以下、指定フォーマットに必要なサイズのメモリを非管理メモリから確保し、変換した値をそのメモリに格納します。
- (3) (2) で確保したメモリの IntPtr によるポインタを返却します。

### 20. 1. 3. 3 DshPtrFmtDataInt ()

IntPtr で与えられるデータアイテムの値を 32 ビット整数に変換します。

#### 【構文】

```
public static int DshPtrFmtDataInt(int fmt, IntPtr ptr)
```

#### 【引数】

fmt

ptr に格納されている値のフォーマットです。(HSMS. ICODE\_U1 など)

適用フォーマットは、ICODE\_L, ICODE\_B, ICODE\_BOOL, ICODE\_U1, ICODE\_U2, ICODE\_U4, ICODE\_I1, ICODE\_I2, ICODE\_I4 です。(それ以外のフォーマットでは、返却値が=0 になります。)

ptr

fmt で指定されたデータが格納されているポインタです。

#### 【戻り値】

変換結果を整数(int)で返却します。

ICODE\_A, J, F4, F8 の場合は、=0 を返却します。

#### 【説明】

IntPtr が指すメモリに格納されている数値データを int 型に変換して返却します。

適用フォーマットは、ICODE\_L, ICODE\_B, ICODE\_BOOL, ICODE\_U1, ICODE\_U2, ICODE\_U4, ICODE\_I1, ICODE\_I2, ICODE\_I4 です。

それ以外のフォーマットでは、0 を返却します。

使用する際は、予め適用するフォーマットであることを前提に使用してください。

## 20. 1. 3. 4 PtrToFormatData()

IntPtr で与えられるデータアイテムの値を、別の非管理メモリに格納し、そのポインタを返します。

### 【構文】

```
public static IntPtr PtrToFormatData(IntPtr ptr, int fmt, int size)
```

### 【引数】

ptr

fmt で指定されたデータが格納されているポインタです。

fmt

ptr に格納されている値のフォーマットです。(HSMS. ICODE\_U1 など)

size

データの配列サイズです。

### 【戻り値】

新しく確保した非管理メモリのポインタを返却します。

引数の ptr が = IntPtr.Zero の場合は、IntPtr.Zero を返却します。

### 【説明】

IntPtr が指すメモリに格納されているデータと同じサイズの非管理メモリを確保し、引数で ptr に与えられたデータをコピーし、新たに確保した非管理メモリのポインタを返却します。

もし、ptr が IntPtr.Zero であった場合は、IntPtr.Zero を返却します。



### 20. 1. 3. 5 DataToPtr()

数値データを非管理メモリに格納し、そのポインタを返します。

#### 【構文】

```
public static IntPtr DataToPtr(byte val)
public static IntPtr DataToPtr(sbyte val)
public static IntPtr DataToPtr(short val)
public static IntPtr DataToPtr(int val)
public static IntPtr DataToPtr(long val)
public static IntPtr DataToPtr(float val)
public static IntPtr DataToPtr(double val)
```

#### 【引数】

val  
データ数値です。

#### 【戻り値】

val が格納された非管理メモリのポインタが返却されます。

#### 【説明】

引数の val にあわせて、値を格納するための非管理メモリを確保し、そこに値を格納し、非管理メモリのポインタを返却します。

### 20. 1. 3. 6 StartPerformanceTimer()

プログラムの処理時間を計測するタイマーを開始します。  
計測する時間の単位は、秒で、float 型の値を得ることができます。

**【構文】**

```
public static void StartPerformanceTimer(int index)
```

**【引数】**

index

計測タイマーのインデクスです。0～7 の値の範囲です。

**【戻り値】**

なし。

**【説明】**

index で指定されたタイマーを開始します。

停止は、次に説明するメソッド StopPerformanceTimer () を使用します。計測時間が取得できます。

### 20. 1. 3. 7 StopPerformanceTimer()

StartPerformanceTimer () で開始したタイマーを停止し、開始してから停止するまでの経過時間を秒単位の、float 型の値で取得します。

**【構文】**

```
public static float StopPerformanceTimer(int index)
```

**【引数】**

index

計測タイマーのインデクスです。0～7 の値の範囲です。

**【戻り値】**

開始から停止までの経過時間を float 型の秒単位の値をが返却します。

時間の分解能は、float の小数点部になります。

**【説明】**

StartPerformanceTimer () で開始したタイマーを停止し、開始してから停止するまでの経過時間を秒単位の、float 型の値で取得します。

## 20. 1. 3. 8 beep()

ビーブ音を鳴らします。

### 【構文】

```
public static void Beep(int a, int b)
```

### 【引数】

a

鳴らす回数です。

b

鳴らすインターバルです。(ms 単位)

### 【戻り値】

なし。

### 【説明】

周波数 2000HZ で、継続時間 600ms の音を b で指定された間隔で、a 回鳴らします。  
その後、戻ります。

### 20. 1. 3. 9 DshPtrToString ()

IntPtr メモリ内の文字列データを string 型の文字列に変換します。  
指定の文字列位置に NULL 文字を設定し、string 型の文字列を返します。

#### 【構文】

```
public static string DshPtrToString(IntPtr ptr, int max_size, int fixed_len)
```

#### 【引数】

ptr  
文字列情報が格納されているメモリのポインタです。

max\_size  
ptr のメモリサイズです。

fixed\_len  
NULL 文字をセットする配列位置です。

#### 【戻り値】

NULL 文字がセットされた後の文字列が返却されます。

#### 【説明】

ptr メモリの fixed\_len 位置に NULL 文字をセットします。  
もし、fixed\_len >= max\_size であれば、NULL 文字をセットしません。

NULL 文字セットを行った後、string 型の文字列データに変換して、それを返却します。

本関数は、SECS-II メッセージから 文字列 (HSMS. ICODE\_A) のデータアイテムを IntPtr メモリに HSMS.D\_GetItem() 関数によって取得した後、得られたデータアイテム長にあわせて string 型文字列に変換するときを使用することができます。

### 20. 1. 3. 10 ptr\_to\_string ()

IntPtr メモリ内に指定フォーマットで保存されているデータを string 型の文字列に変換します。

#### 【構文】

```
public static string ptr_to_string(IntPtr ptr, int format)
```

#### 【引数】

ptr

2 番目の引数 format のフォーマットのデータが格納されているメモリのポインタです。

format

データフォーマット(SECS のフォーマット)を指定します。

HSMS クラスの ICODE\_XX 定数を指定します。

ICODE\_XX の XX 部分は、L, A, B, I1, U1 など。

#### 【戻り値】

データが変換された string 型の文字列が返却されます。

#### 【説明】

format で指定されたフォーマットで ptr に指定された領域に保存されているデータを文字列に変換します。

ptr が null (IntPtr.Zero) の場合は、文字列長=0 の文字列を戻します。

本メソッドは、2 1. 2 で説明する DshPtrToData クラスを使用します。

## 20. 2 DshPtrToData クラス

IntPtr 型に保存されている SECS-II が定めるフォーマットのデータを、クラス内の対応するプロパティにコピーします。

### 20. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明   |
|---|--|--|
| 1 | public DshPtrToData ()                       | インスタスを生成します。   |
| 2 | public DshPtrToData (IntPtr ptr, int format) | データが保存されているポイントとデータのフォーマットを指定します。ただしに、ptr_to_value() メソッドを実行し、フォーマットに対応したプロパティにコピーします。 |

### 20. 2. 2 プロパティ

変換結果データの保存用のプロパティがフォーマットに対応して準備されています。

|    | 名前                    | 説明   |
|----|-----------------------|--|
| 1  | public sbyte int_8    | HSMS. ICODE_B, HSMS. ICODE_I1 フォーマットのデータがコピーされます。(ICODE_B は、int_8, uint_8 の両方にコピーされます) |
| 2  | public byte uint_8    | HSMS. ICODE_B, HSMS. ICODE_U1 フォーマットのデータがコピーされます。                                      |
| 3  | public bool b_bool    | HSMS. ICOE_BOOLEAN のデータがコピーされます。   |
| 4  | public Int16 int_16   | HSMS. ICOE_I2 のデータがコピーされます。  |
| 5  | public UInt16 uint_16 | HSMS. ICOE_U2 のデータがコピーされます。  |
| 6  | public Int32 int_32   | HSMS. ICOE_I4 のデータがコピーされます。  |
| 7  | public UInt32 uint_32 | HSMS. ICOE_U4 のデータがコピーされます。  |
| 8  | public Int64 int_64   | HSMS. ICOE_I8 のデータがコピーされます。  |
| 9  | public UInt64 uint_64 | HSMS. ICOE_U8 のデータがコピーされます。  |
| 10 | public float f_32     | HSMS. ICOE_F4 のデータがコピーされます。  |
| 11 | public double f_64    | HSMS. ICOE_F8 のデータがコピーされます。  |
| 12 | public string s_str   | HSMS. ICOE_A, HSMS. ICODE_J のデータがコピーされます。  |

## 20. 2. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前  | 説明   |
|---|---|--|
| 1 | public void ptr_to_value(IntPtr ptr, int fmt) | ptr に与えられたデータを fmt に対応するプロパティにコピーします。                                    |
| 2 | public void Dispose()                         | クラス内部で使用された資源(メモリ)をシステムに返却します。他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。 |

### 20. 2. 3. 1 ptr\_to\_value()

IntPtr メモリに格納されているデータを指定されたフォーマットに対応するプロパティにコピーします。

#### 【構文】

```
public void ptr_to_value(IntPtr ptr, int fmt)
```

#### 【引数】

ptr

fmt で指定されたデータが格納されているポインタです。

fmt

ptr に格納されている値のフォーマットです。(HSMS.ITEM\_U1 など)

#### 【戻り値】

ありません。

#### 【説明】

ptr で指定された IntPtr メモリに保存されている fmt のデータを、fmt のフォーマットに対応するプロパティにコピーします。

ptr の値が null (=IntPtr.Zero) の場合には、該当プロパティに、数値データの場合は、=0 を、文字列の場合には、= "" を設定します。

また、fmt に対応しない他のプロパティには、数値データの場合は、=0、文字列データの場合は、"" (長さ 0 の文字列) が設定されます。

クラスのユーザは、本メソッド実行のあと、fmt に対応するプロパティの値を参照あるいは取得し、処理に使用することができます。

## 21. DshDebug - クラス・トレースのためのクラス

クラスライブラリに含まれるクラスの生成、資源開放、消滅3つの事象の発生回数をカウントするとともに、事象発生時にそれらの情報をリアルタイムで、ユーザが提供する Windows Form に表示させる処理を行うためのクラスです。

説明の中で、クラスの生成、資源開放、消滅の3つの事象のことを**トレース事象**と呼ぶことにします。

### 21. 1 DshDebug クラス

本クラスは、他のクラスのトレース事象(Construct, Dispose, Finalize)の発生回数を管理し、トレースが有効になっているクラスについてはそのトレース事象発生を、ユーザの Windows フォームに通知する役割を果たします。

本クラスが有するメソッドは全て static 関数になっています。したがって、ユーザが本クラスのメソッドを呼び出す場合、**DshDebug.<method>**のように、DshDebug クラスを直接呼び出すことになります。

なお、クラスのトレース表示機能については、次のドキュメントを参照ください。

「**文書番号 DSHEng4-11-30360-00 クラス生成・消滅トレースと表示機能について**」

本トレース機能を使用するためには、以下の処理を行う必要があります。

- (1) `init_DshDebug()` で初期設定する。  
(DshEngine が生成されるときに自動的に呼び出されます。)
- (2) ユーザが用意するクラス・トレース表示用の Windows Form の情報を設定する。  
Form ハンドルと、その Form にトレース情報を送信するための Windows Message ID です。
- (3) DshDebug クラスのトレース表示有効設定 (クラス全体)
- (4) トレースしたいクラスの有効設定 (クラス個別)

なお、トレース表示とは別に、トレース事象のカウンター値は (2) ~ (3) の設定に関係なく、いつでも取得することができます。



## 21. 1. 1 コンストラクタ

使用しません。

## 21. 1. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                           | 説明   |
|---|------------------------------|--|
| 1 | public constant int DBG_xxxx | ユーザが参照できるクラスのインデクス名です。<br>xxxx の部分がクラス名です。<br>インデクスを必要とするメソッドに引数として説明します。<br>例 DshAlarm クラスのインデクス名は DBG_DshAlarm になります。  |
| 2 | private な情報                  | これらは、メソッドを通して設定または取得できます。<br>formHandle, form への Windows Message の設定<br>トレース有効/無効フラグ<br>トレースカウンタ, クラス名<br>トレースカウンタには、Construct, Dispose, Finalize の<br>3つのカウンタがあります。 |

## 21. 1. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前  | 説明   |
|---|---|--|
| 1 | public static void init_DshDebug()          | DshDebug クラス内部の情報をセットアップします。<br>本メソッドは、DshEngine クラスが生成されたときに自動的に実行されます。 |
| 2 | public static void set_trace_form_handle () | ユーザがトレース表示出力したい Windows Form のハンドルを指定します。                                |
| 3 | public static void set_trace_WM()           | 上の2. で指定した Form に通知して貰う Windows Message ID を指定します。                       |
| 4 | public static void enable_trace_log()       | DshDebug のトレース表示通知の有効/無効設定をします。(全体)                                      |
| 5 | public static string get_class_name()       | インデクスで指定したクラスの名前を取得します。  |
| 6 | public static bool set_trace_flag()         | クラスを指定してトレース表示通知の有効/無効を設定します。  |
| 7 | public static bool get_trace_flag()         | クラスを指定してトレース表示通知の有効/無効設定状態を取得します。  |
| 8 | public static void reset_trace_count()      | クラスを指定して、そのクラスのトレース事象カウンタを 0 にリセットします。                                   |
| 9 | public static int get_trace_count()         | クラスを指定して、トレース事象カウンタ値を取得します。<br>(トレース事象の3つのカウンタ値を取得します。)                  |

### 21. 1. 3. 1 init\_DshDebug()

クラスのトレース機能を開始するための準備をします。

DshEngine クラスが生成された際、本メソッドが未実行状態である場合、自動的に実行してくれます。

クラス名の設定、カウンターのリセットなどを行います。

#### 【構文】

```
public static void init_DshDebug()
```

#### 【引数】

なし。

#### 【戻り値】

なし。

#### 【説明】

本メソッドはクラスのトレース処理のために必ず実行する必要があります。

DshEngine クラスが生成されたときに、本メソッドが未実行状態であれば自動的に本メソッドを実行します。

ユーザプログラムからは、DshEngine クラスの生成（new キーワードによる）の前に実行することもできます。

本メソッドが実行する処理は、トレース対象となる全クラスの名前情報の設定、トレース事象カウンター（Construct, Dispose, Finalize カウンタ）の初期化などです。

### 21. 1. 3. 2 set\_trace\_form\_handle ()

トレースしたクラスの内容を表示通知するための宛先 Windows フォームのハンドルを設定します。

ユーザに対するトレース表示情報は、ここで与えられたハンドルのフォームへ通知されます。

**【構文】**

```
public static void set_trace_form_handle(IntPtr handle)
```

**【引数】**

handle

Windows フォームのハンドルを指定します。

**【戻り値】**

なし。

**【説明】**

クラスのトレース事象の発生通知する相手の Windows Form のハンドルを設定します。

### 21. 1. 3. 3 set\_trace\_WM()

先に説明した set\_trace\_form\_handle() で指定されたフォームにトレース情報を通知するときに使用する Windows Message の ID を設定します。

**【構文】**

```
public static void set_WM(int wm)
```

**【引数】**

wm

Windows Message の ID です。

**【戻り値】**

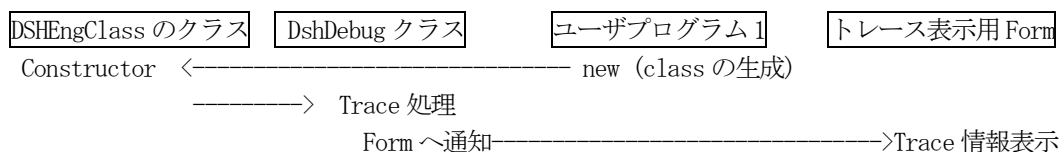
なし。

**【説明】**

先の 21. 1. 3. 2 で設定したハンドルの Form に通知する Windows Message ID を設定します。

**【補足】**

たとえば、Construct 時のトレースの流れは次のようになります。



### 21. 1. 3. 4 enable\_trace\_log()

DshDebug クラスに対して、トレース表示通知の有効/無効（通知を行うかどうか）を指定します。

**【構文】**

```
public static void enable_trace_log (bool flag)
```

**【引数】**

flag

有効にする場合は true、無効にする場合は false を指定します。

**【戻り値】**

なし。

**【説明】**

トレース事象が発生したときに、トレース情報をユーザが指定したフォームに通知するかどうかを指定します。

true（有効）を指定すると通知することになり、false（無効）を指定すると通知しないことになります。

### 21. 1. 3. 5 get\_class\_name()

クラス・インデクスで指定したクラスの名前を取得します。

**【構文】**

```
public static string get_class_name(int x)
```

**【引数】**

x

クラスを特定するクラス・インデクスです。（0～）

（DshDebug クラスの定数としてクラス名に対応するクラスインデクスの名前が定義されています。）

**【戻り値】**

クラス名が返却されます。存在しないクラスの場合は “????” が返却されます。

**【説明】**

クラス・インデクスを指定してクラス名を取得します。

### 21. 1. 3. 6 set\_trace\_flag()

クラス単位で、トレース表示の有効/無効を設定します。

#### 【構文】

```
public static bool set_trace_flag(int x, bool flag)
```

#### 【引数】

x

クラスを特定するクラス・インデクスです。(0～)  
(DshDebug クラスの定数としてクラス名に対応するクラスインデクスの名前が定義されています。)

flag

有効/無効を true/false で設定します。

#### 【戻り値】

正常に設定できた場合は true, インデクスのクラスが存在しなかった場合は false を返却します。

#### 【説明】

クラス単位でトレース表示の有効/無効 (通知するかどうか) を設定します。

### 21. 1. 3. 7 get\_trace\_flag()

クラス単位で、トレース表示の有効/無効情報を取得します。

#### 【構文】

```
public static bool get_trace_flag(int x,)
```

#### 【引数】

x

クラスを特定するクラス・インデクスです。(0～)

#### 【戻り値】

有効であった場合は true, 無効であった場合は false を返却します。

#### 【説明】

クラス単位でトレース表示の有効/無効 (通知するかどうか) 情報を取得します。

### 21. 1. 3. 8 reset\_trace\_count()

DshDebug クラスとして、ユーザに対し、トレース表示通知の有効/無効（通知を行うかどうか）を指定します。

**【構文】**

```
public static void reset_trace_count(int x)
```

**【引数】**

x  
クラスを特定するクラス・インデクスです。(0～)

**【戻り値】**

なし。

**【説明】**

指定されたクラスのトレースカウンタを =0 にします。

### 21. 1. 3. 9 get\_trace\_count()

クラス・インデクスで指定したクラスのトレース事象カウンター値を取得します。

**【構文】**

```
public static int get_trace_count(int x, ref int nc, ref int dc, ref int fc)
```

**【引数】**

x  
クラスを特定するクラス・インデクスです。(0～)

nc  
new で construct されたカウント格納用変数です。

d  
Disposer が呼ばれたカウント格納用変数です。

fc  
Finalizer が呼ばれたカウント格納用変数です。

**【戻り値】**

nc に設定された値を返却します。

**【説明】**

現時点までのトレース事象が発生した回数をそれぞれ指定された変数に格納返却します。

## 22. フォーマット付きプロセス・プログラム関連クラス

フォーマット付きプロセス・プログラム(FPP)情報を保存するクラスです。

関連クラスの一覧を次表に示します。

|   | クラス名     | 用途                         |
|---|----------|----------------------------|
| 1 | DshFPP   | FPP 定義情報保存クラスです。<br>S7F23  |
| 2 | DshCCode | DshFPP のコメントコード情報保存に使用します。 |
| 3 | DshPPara | DshCCode の<br>S7F25        |

FPP の通信関連クラスとして、以下のものがあります。 これらについては、Vo1-2 で説明します。

|                  |
|------------------|
| Dsh4S7F3Send     |
| DshS7F25Send     |
| DshS7F24Response |

## 22. 1 DshFPP クラス

FPP フォーマット付きプロセスプログラム情報を保存するために使用されます。  
 通常、S7F23 メッセージを通して装置、ホスト間でやり取りされる情報です。  
 本クラスは、22.2 で説明する DshCCode クラスを使用します。

### 22. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前  | 説明  |
|---|---|---|
| 1 | public DshFPP()   | インスタスを生成します。<br>(FPPID の指定を、後で set_id メソッドを使って行います) |
| 2 | public DshFPP(string fppid)                                 | インスタスを FPPID を指定して生成します。                            |
| 3 | public DshFPP(string fppid,<br>string mdlm, string softrev) | 同様に mdlm, softrev を指定して生成します。                       |

### 22. 2. プロパティ

プロパティ一覧表に示します。

|   | 名前                           | 説明   |
|---|------------------------------|--|
| 1 | public string fppid          | FPP の ID です。(文字列)<br>エンジンはこの ID で管理します。  |
| 2 | public string name           | FPP 名です。fppid と同じ値です。  |
| 3 | public string mdlm           | FPP のモデル名  |
| 4 | public string softrev        |  |
| 4 | public int ccode_count       | FPP に付属するプロセッサ情報の数です。  |
| 5 | public DshCCode[] ccode_list | FPP に付属するプロセッサ情報を保存するクラスの配列です。<br>ccode_count が保存されているプロセッサ数です。<br>DshCCode クラスは、内部パラメータ保存用に DshPPPara クラスを使用する。 |
| 6 | public int state             | FPP の状態です。<br>(状態値はユーザで定義してください)   |



## 22. 1. 3 メソッド

本クラスのメソッドは次の通りです。

|     | 名前                        | 説明  |
|-----|---------------------------|---|
| 1   | public void set_id()      | fppidプロパティに ID を設定します。  |
| 2   | public void set_mdln()    | mdln を設定します。  |
|     | public void set_softrev() | softrev を設定します。   |
| 3   | public void add_attr()    | プロパティを 1 個 attr_list に追加します。  |
| 4   | public int get_attr()     | attr_list から指定された配列位置のプロパティ情報を取得します。  |
| 5   | public int alloc_id()     | FPPID をエンジンに登録します。  |
| 6   | public int set()          | インスタンス内に設定された FPP 情報をエンジンに設定します。  |
| 7   | public int get()          | エンジンから FPP 情報を取得します。  |
| 8   | public void delete()      | fppid に指定された FPP 情報をエンジンの管理情報から削除します。   |
| 9   | public int decode()       | S7F23 メッセージを当該クラスにデコードします。  |
| 10  | public void copy()        | DshFPP クラスの当該インスタンスの内容を別のインスタンスにコピーします。   |
| 11  | public int get_id_count() | エンジンに登録されている FPPID 情報の合計数を取得します。  |
| 22. | public int get_id_list()  | エンジンに登録されている全 FPPID をリストに取得します。   |
| 13  | public void clear()       | attr_list 配列に設定されているプロパティ情報を全てクリアします。そして、ccode_count =0 にします。   |
| 14  | public void Dispose()     | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。<br>3. 1. 3. 24 と同様です。そちらを参照ください。 |

### 22. 1. 3. 1 set\_id()

FPPID を設定します。

FPPID は、エンジンに登録し管理するための ID です。S7F23 などメッセージ送信などに使用されます。

**【構文】**

```
public void set_id( string fppid )
```

**【引数】**

fppid

設定する FPPID です。

**【戻り値】**

なし。

**【説明】**

引数に与えられた fppid の値をインスタンス内の fppid に設定します。

### 22. 1. 3. 2 set\_mdln()

プロパティ mdln に値を設定します。

**【構文】**

```
public void set_mdln(string mdln)
```

**【引数】**

mdln

設定したい値です。

**【戻り値】**

なし。

**【説明】**

引数に与えられた mdln の値を設定します。

### 22. 1. 3. 3 set\_softrev()

プロパティ softrev に値を設定します。

**【構文】**

```
public void set_softrev(string softrev)
```

**【引数】**

softrev

設定したい値です。

**【戻り値】**

なし。

**【説明】**

引数に与えられた softrev の値を設定します。

### 22. 1. 3. 4 add\_ccode()

プロパティ ccode\_list、コマンドコードリストに DshCCode クラスを 1 個追加設定します。  
DshCCode クラスのインスタンスの情報を予め設定した上で追加します。

**【構文】**

```
public int add_ (DshCCode ccode)
```

**【引数】**

ccode

DshCCode クラスのインスタンスを指定します。

**【戻り値】**

なし。

**【説明】**

引数に与えられた ccode のインスタンスを、ccode\_list 配列に追加します。  
ccode\_list の配列が 1 個拡張され、そこに設定されます。  
追加したあとは、ccode\_count の値が +1 されます。

### 22. 1. 3. 5 set()

クラス内の FPP 情報をエンジンに設定します。

**【構文】**

```
public int set()
```

**【引数】**

なし。

**【戻り値】**

| 返却値  | 意 味       |
|------|-----------|
| 0    | 正常に設定できた。 |
| (-1) | 設定に失敗した。  |

**【説明】**

インスタンス内に設定された FPP 情報をエンジンの管理情報に設定します。

### 22. 1. 3. 6. get()

エンジンから指定された FPPID の情報を取得します。

**【構文】**

```
public int get()
public int get( string fppid)
```

**【引数】**

fppid  
情報を取得したい FPPID です。

**【戻り値】**

| 返却値  | 意 味                       |
|------|---------------------------|
| 0    | 正常に取得できた。                 |
| (-1) | 取得に失敗した。(エンジン内に見つからなかった。) |

**【説明】**

エンジンの管理情報から FPP 情報を取得します。  
引数で fppid が指定された場合は、その FPP の fppid を含めて、情報を取得します。

引数がない場合は、インスタンス内の fppid の FPP の情報を取得します。  
情報は、プロパティ内に読み込まれます。

正常に取得できた場合は、0 を、失敗した場合は(-1)を返却します。

## 22. 1. 3. 7 alloc\_id()

FPPID をエンジンに新規登録します。

### 【構文】

```
public int alloc_id()
public int alloc_id(string fppid)
```

### 【引数】

fppid  
登録する FPPID です。

### 【戻り値】

| 返却値    | 意味                              |
|--------|---------------------------------|
| 0 or 1 | 正常に登録できた。(1は既に登録済みであったことを意味します) |
| (-1)   | 登録に失敗した。                        |

### 【説明】

引数として fppid が指定された場合は、その FPPID をエンジンに登録します。  
登録できた場合は、引数の fppid がインスタンス内の fppid に設定されます。

引数がない場合は、インスタンス内の fppid の FPPID をエンジンに登録します。

正常に登録できた場合は、0 または 1 を返却します。失敗した場合は、(-1) を返します。

既に ID が登録されていた場合、エンジンは管理情報の中から ID 以外の情報をクリアします。



## 22. 1. 3. 8 delete()

指定された FPP の情報をエンジン管理情報の中から削除します。

### 【構文】

```
public int delete()
public int delete(string fppid)
```

### 【引数】

fppid  
削除したい FPP の ID です。

### 【戻り値】

| 返却値  | 意味                      |
|------|-------------------------|
| 0    | 正常に削除できた。               |
| 1    | エンジンに登録されていなかった。        |
| (-1) | 削除できなかった。(fppidが無効であった) |

### 【説明】

1 個の FPP 情報をエンジンの管理情報から削除します。

削除対象は、eqid で指定された装置の情報であり、引数に FPPID の指定があるメソッドでは、引数で指定された FPP を、そして、引数が無いメソッドの場合は、インスタンスの fppid プロパティの FPP を削除します。

一旦削除された情報は、他のメソッドでは復元できませんので注意してください。

## 22. 1. 3. 9 copy()

当該クラスのインスタンスの内容を他のインスタンスにコピーします。

### 【構文】

```
public void copy( ref DshFPP dst )
```

### 【引数】

dst  
コピー先の DshFPP インスタンスです。

### 【戻り値】

なし。

### 【説明】

当該インスタンスの内容（プロパティ値）を dst に指定されるインスタンスにコピーします。

## 22. 1. 3. 10 decode()

S7F23 メッセージに含まれる情報を DshFPP クラス内のプロパティにデコードします。

### 【構文】

```
public int decode(ref DSHMSG msg)
```

### 【引数】

msg

S7F23 のメッセージ情報（生情報）が格納されている DSHMSG 構造体領域になります。DSHMSG は、1 次メッセージをポーリングした際にエンジンから与えられる情報です。ユーザは DSHMSG 構造体については、特に意識しないで、ポーリングした後、本メソッドに渡すだけです。

### 【戻り値】

| 返却値  | 意味                                   |
|------|--------------------------------------|
| 0    | 正常にデコードできた。                          |
| (-1) | デコードできなかった (S7F23 のメッセージの形式が正しくなかった) |

### 【説明】

msg に含まれている FPP 情報を DshFPP クラス内にデコードします。

msg S7F23

```
L, 4
  ppid
  mdln
  softrev
L, c
  L2
    ccode
  L, p
    pparam1
    .
    pparamp
L, 2
  .
```

decode

```
DshFPP
Class
  fppid
  mdln
  softrev
  ccode_count
  ccode_list[]
```

メッセージに含まれる FPP 情報(fppid, mdln, softrev, プロパティ名)は、クラスのプロパティに保存されます。

正常にデコードできた場合、0 を返却します。

もし、S7F23 のメッセージフォーマットが正しくないなどの理由でデコードできなかった場合、(-1) を返却します。



### 22. 1. 3. 11 get\_id\_count()

エンジンの当該装置に登録されている FPPID の合計数を取得します。

**【構文】**

```
public int get_id_count()
```

**【引数】**

なし。

**【戻り値】**

登録されている FPPID 数が返却されます。

**【説明】**

登録されている FPPID の合計数を取得します。

### 22. 1. 3. 12 get\_id\_list()

エンジンの当該装置に登録されている全 FPP の ID と名前のリストを取得します。

**【構文】**

```
public int get_id_list(string[] id_list, string[] n, name_list, int list_size)
```

**【引数】**

id\_list

FPPID を格納する配列です。

name\_list

FPP 名を格納する配列 (name は id と同じになります) です。

list\_size

準備されたリストの配列サイズです。

**【戻り値】**

取得できた FPPID の数が返却されます。

**【説明】**

エンジンに登録されている全 FPPID とその名前をそれぞれ id\_list[], name\_list[] に取得します。

list\_size は配列のサイズを指定します。

配列は、登録されている全 FPPID を保存できる充分のサイズの配列を準備してください。

返却値は取得できた FPPID の合計数です。

(注) ID と名前格納用リストのサイズは、先に説明した get\_id\_count() メソッドで合計 ID 数を取得し、そのサイズの配列を準備して本メソッドを使用すると便利です。

### 22. 1. 3. 13 clear()

インスタンス内のプロセスコマンド情報の配列内の情報を消去します。

**【構文】**

```
public void clear()
```

**【引数】**

なし。

**【戻り値】**

なし。

**【説明】**

attr\_list 配列の中にプロセスコマンド値に使用されていた非管理メモリの開放を行い、そして空にします。  
ccode\_count を =0 にします。  
また、mdl, softrev も空にします。

## 22. 2 DshCCode プロセスコマンド・クラス

DshPPP クラスを構成するプロセスコマンド情報を保存するためのクラスです。

また、本クラスは、22.3 で説明する DshPPara クラスを使用します。

### 22. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前  | 説明  |
|---|---|---|
| 1 | public DshCCode()                                   | 空のインスタスを生成します。<br>コマンドコード ccode のフォーマットは A(20) とします。              |
| 2 | public DshCCode(string ccode)                       | コマンドコードを指定してインスタスを生成します。<br>コマンドコードのフォーマットが A(20) であることを前提としています。 |
| 3 | public DshCCode(int format, int size, IntPtr ccode) | コマンドコードのフォーマット、データサイズとコードが格納されているメモリのポインタを指定し、生成します。              |

### 22. 2. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                          | 説明   |
|---|-----------------------------|--|
| 1 | public IntPtr ccode         | プロセスコマンドのコマンドコードです。                                  |
| 2 | public int code_format      | ccode のフォーマットです<br>デフォルトは A(20) です。(DshCCode 生成時)    |
| 3 | public int code_size        | ccode のデータ長です。                                       |
| 4 | public string a_code        | ccode です。但し、ccode のフォーマットが A(20) の場合だけ有効です。          |
| 5 | public int para_count       | ccode に付属するパラメータの数です。<br>下の para_list[] の配列サイズになります。 |
| 6 | public DshPPara[] para_list | パラメータ情報のリストです。<br>DshPPara クラスの配列です。                 |

## 22. 2. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前  | 説明   |
|---|---|--|
| 1 | public void clear()                                       | para_list 配列を空にします。  |
| 2 | public void add_para(<br>DshPPara para)                   | DshPPara クラスのインスタンス情報を 1 個 para_list に追加します。                                 |
| 3 | public void add_para(string para)                         | 文字列パラメータを 1 個 para_list に追加します。  |
| 4 | public void add_para(int format,<br>int size, IntPtr ptr) | 指定フォーマットのパラメータを 1 個 para_list に追加します。  |
| 5 | public void copy()  | DshCCode クラスの当該インスタンスの内容を別のインスタンスにコピーします。                                    |
| 6 | public void Dispose()                                     | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。 |

### 22. 2. 3. 1 clear()

インスタンスの para\_list 配列リストを空にします。

#### 【構文】

```
public void clear()
```

#### 【引数】

なし

#### 【戻り値】

なし。

#### 【説明】

para\_list 配列に含まれるパラメータ情報に使用されている非管理メモリを開放します。  
そして、para\_list 配列を空にし、 para\_count=0 にします。

## 22. 2. 3. 2 add\_para()

パラメータを para\_list リストに追加します。

### 【構文】

```
public void add_para(string str_para)
public void add_para(int format, int size, IntPtr ptr )
public void add_para( DshPPara para )
```

### 【引数】

str\_para  
追加する文字列型(フォーマット-A)のパラメータデータ

format  
パラメータのフォーマット

size  
パラメータのデータ数(文字列であればバイト数)

ptr  
パラメータデータが格納されているメモリのポインタ

para  
1個のパラメータ情報が格納されている DshPPara のインスタンス

### 【戻り値】

なし。

### 【説明】

引数に指定されている1個のパラメータ情報を ccode\_list に追加する。追加した後、para\_count を +1 する。

なお、DshPPara は、format, size, 値のポインタのプロパティを有するクラスです。

str\_para を引数にした場合、DshPara クラスのインスタンスを生成し、format = A(20), size=文字列のバイト数、そして、value には、str\_para の値を確保した非管理メモリに移し変え、そのポインタを設定します。

(注) format A は、本クラスライブラリでは、HSMS. ICODE\_A で参照します。

### 22. 2. 3. 3 copy()

当該インスタンスの内容を別の DshCCode クラスのインスタンスにコピーします。

**【構文】**

```
public void copy(ref DshCCode dst)
```

**【引数】**

dst

コピー先インスタンスです。

**【戻り値】**

なし。

**【説明】**

当該インスタンス内に保存されているプロパティ情報を dst で指定されたインスタンスにコピーします。

## 22. 3 DshPPara プロセスコマンド・パラメータ・クラス

DshPPP クラスを構成する DshCCode プロセスコマンド・クラス情報のパラメータ情報を保存するためのクラスです。

### 22. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

|   | 名前   | 説明  |
|---|--|---|
| 1 | <code>public DshPPara()</code>                                       | 空のインスタンスを生成します。   |
| 2 | <code>public DshPPara(string val)</code>                             | パラメータを指定してインスタンスを生成します。<br>コマンドコードのフォーマットが A(20)であることを前提としています。 |
| 3 | <code>public DshCCode(int format,<br/>int size, IntPtr ccode)</code> | コマンドコードのフォーマット、データサイズとコードが格納されているメモリのポインタを指定し、生成します。            |

### 22. 3. 2 プロパティ

プロパティ一覧表に示します。

|   | 名前                               | 説明                                    |
|---|----------------------------------|---------------------------------------|
| 1 | <code>public int format</code>   | パラメータのフォーマットです                        |
| 2 | <code>public int size</code>     | パラメータのデータ長です。                         |
| 3 | <code>public IntPtr value</code> | パラメータ値が格納されているメモリのポインタです。<br>(非管理メモリ) |

## 22. 3. 3 メソッド

本クラスのメソッドは次の通りです。

|   | 名前  | 説明   |
|---|---|--|
| 1 | public void clear()                                       | プロパティをクリアします。  |
| 3 | public void set_data(string val)                          | 文字列データを設定します。  |
| 4 | public void set_para(int format,<br>int size, IntPtr val) | 指定フォーマットのデータを設定します。  |
| 5 | public void copy()  | DshPPara クラスの当該インスタンスの内容を別のインスタンスにコピーします。                                    |
| 6 | public void Dispose()                                     | クラス内部で使用された資源(メモリ)をシステムに返却します。<br>他のクラスをプロパティにしている場合、そのクラスの Dispose() も行います。 |

### 22. 3. 3. 1 clear()

インスタンスの value をクリアします。

**【構文】**

```
public void clear()
```

**【引数】**

なし

**【戻り値】**

なし。

**【説明】**

value に使用されている非管理メモリを開放します。  
そして、format, size を = 0 にします。



### 22. 3. 3. 2 set\_data()

パラメータ値を設定します。

#### 【構文】

```
public void set_data(string str_val)
public void set_data(int format, int size, IntPtr ptr )
```

#### 【引数】

str\_val  
追加する文字列型(フォーマット-A)のパラメータデータ

format  
値のフォーマット

size  
値のデータ数(文字列であればバイト数)

ptr  
値のデータが格納されているメモリのポインタ

#### 【戻り値】

なし。

#### 【説明】

引数に指定されているデータをパラメータ値として設定します。

str\_val の場合、プロパティの format = A(20), size=文字列のバイト数、そして、value には、val の値を確保した非管理メモリに移し変え、そのポインタを設定します。

(注) format A は、本クラスライブラリでは、HSMS. ICODE\_A で参照します。

### 22. 3. 3. 3 copy()

当該インスタンスの内容を別の DshCCode クラスのインスタンスにコピーします。

#### 【構文】

```
public void copy(ref DshCCode dst)
```

#### 【引数】

dst  
コピー先インスタンスです。

#### 【戻り値】

なし。

#### 【説明】

当該インスタンス内に保存されているプロパティ情報を dst で指定されたインスタンスにコピーします。