1. SECS類別定義：
   1. SXFY：

依據SECS格式定義對應的Stream Function類別的父類別。

SECS Control會根據SECS類別宣告的樣式，進行轉換為SECS格式之物件(Raw Data)。

而SECS Control也會根據收到的SECS格式之物件(Raw Data)，依序轉換為SECS類別之物件。

因此，必須注意SXFY子類別宣告的Field順序，即是SECS物件的元素順序。

* 1. S1F6：

|  |
| --- |
| public class S1F6 : SXFY //必須繼承SXFY  {  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 1)] //指定SFCD於SECS中 //代表的型別為ASCII，以及//長度為1  public string SFCD;  public S1F6()  {  StreamFunction = "S1F6"; //設定StreamFunction  }  } |

* 1. S1F6\_SFCD1：此為S1F6的延伸類別

|  |
| --- |
| public class S1F6\_SFCD1 : SXFY  {  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 1)]  public string SFCD;  public SFCD\_1\_T SFCD\_1; //繼承自SXFY類別的屬性，不必指定SECS Element型別  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 2)]  public string PORTID;  public SITEM[] SARY\_2;  [SecsElement(Type = SecsElement.SecsElementType.NOT\_USED)] //指定version欄位非SECS的元素  public int version;  public S1F6\_SFCD1()  {  StreamFunction = "S1F6";  }  public class SFCD\_1\_T : SXFY  {  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 1)]  public string CRST;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 1)]  public string EQST;  public SITEM[] SARY;  }  public class SITEM : SXFY  {  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 2)]  public string INDEX;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 2)]  public string KEY;  }  } |

* 1. S1F6\_SFCD2：此為S1F6的延伸類別

|  |
| --- |
| public class S1F6\_SFCD2 : SXFY  {  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 1)]  public string SFCD;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_LIST, Length = 3)]  public SFCD\_2\_T SFCD\_2;  public S1F6\_SFCD2()  {  StreamFunction = "S1F6";  }  public class SFCD\_2\_T : SXFY  {  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 3)]  public string PTID;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 2)]  public string PTTYPE;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 2)]  public string PTUSETYPE;  }  } |

1. Reply SECS：

|  |
| --- |
| S1F6\_SFCD1 s1f6\_sfcd1 = new S1F6\_SFCD1()  {  PORTID = "P1",  SFCD = "A",  SFCD\_1 = new S1F6\_SFCD1.SFCD\_1\_T()  {  CRST = "A",  EQST = "B",  SARY = new S1F6\_SFCD1.SITEM[2]  },  SARY\_2 = new S1F6\_SFCD1.SITEM[1]  };  s1f6\_sfcd1.SFCD\_1.SARY[0] = new S1F6\_SFCD1.SITEM()  {  INDEX = "1",  KEY = "K1"  };  s1f6\_sfcd1.SFCD\_1.SARY[1] = new S1F6\_SFCD1.SITEM()  {  INDEX = "2",  KEY = "K2"  };  s1f6\_sfcd1.SARY\_2[0] = new S1F6\_SFCD1.SITEM()  {  INDEX = "0",  KEY = "K0"  };  TrxSECS.ReturnCode rplyCode = secsAgent.TrxSECS.replySECS(s1f6\_sfcd1); //使用此Method來傳遞SECS訊息 |

1. Send And Receive SECS：

|  |
| --- |
| public class S1F75 : SXFY  {  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 5)]  public string LINE\_ID;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 4)]  public string EQPT\_ID;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 2)]  public string PORT\_ID;  public S1F75()  {  StreamFunction = "S1F75";  }  } |
| public class S1F76 : SXFY  {  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 5)]  public string LINE\_ID;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 4)]  public string EQPT\_ID;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 2)]  public string PORT\_ID;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 5)]  public string CST\_ID;  [SecsElement(Type = SecsElement.SecsElementType.TYPE\_ASCII, Length = 1)]  public string IS\_EMPTY;  public S1F76()  {  StreamFunction = "S1F76";  }  } |
| BCFApplication app = BCFApplication.getInstance();  //依據SECSConnectionSetting設定的「Connection Name」取得SECS Agent  SECSAgent secsAgent = app.getSECSAgent("EQ");  S1F75 s1f75 = new S1F75()  {  LINE\_ID = "LN-01",  EQPT\_ID = "PCLN",  PORT\_ID = "P1"  };  S1F76 s1f76 = null;  string rtnMsg = null;  //預期送出S1F75，接收S1F76。並且符合LINE\_ID為「LN-01」的條件  TrxSECS.ReturnCode rtnCode = secsAgent.TrxSECS.sendRecv<S1F76>(s1f75, out s1f76, out rtnMsg,  s => (s.LINE\_ID == "LN-01")); //針對指定的過濾條件進行過濾  if (rtnCode == TrxSECS.ReturnCode.Normal)  {  Console.WriteLine("Receive suceess!");  }  else  {  Console.WriteLine("Receive fail [{0}]", rtnCode);  } |

1. Receive SECS Listener：

|  |
| --- |
| void function()  {  //…….  secsAgent.addSECSReceivedHandler("S1F76", (\_sender, \_e) => onReceiveSecsHandler(\_sender, \_e));  }  void onReceiveSecsHandler(object sender, SECSEventArgs e)  {  Console.WriteLine("Event Text:{0}", e.EventText);  int S = e.S;  int F = e.F;  Console.WriteLine("S = {0}, F = {1}", S, F);  //Do Something  } |

1. SECS Connection Config：
   1. SECSConnectionSetting.config

|  |
| --- |
| <SECSConnectionSetting>  <!-- ConnectMode: 0 = Passive, 1 = Active -->  <Connection Name="EQ" DeviceID="0" LocalIP="127.0.0.1" LocalPort="5001" RemoteIP="127.0.0.1" RemotePort="5000"  T3Timeout="30" T5Timeout="5" T6Timeout="5" T7Timeout="5" T8Timeout="5" RetryCount="2"  ConnectMode="1"  TXLogPath="D:\SECSLog\">  </Connection>  <Connection Name="MES" DeviceID="0" LocalIP="127.0.0.1" LocalPort="5001" RemoteIP="127.0.0.1" RemotePort="5000"  T3Timeout="30" T5Timeout="5" T6Timeout="5" T7Timeout="5" T8Timeout="5" RetryCount="2"  ConnectMode="1"  TXLogPath="D:\SECSLog\">  </Connection>  </SECSConnectionSetting> |

* 1. App.config

|  |
| --- |
| <configuration>  <configSections>  <section name="SECSConnectionSetting" type="com.mirle.ibg3k0.bcf.ConfigHandler.SECSConnectionConfigHandler,BlockControlFramework" allowLocation="true" allowDefinition="Everywhere"/>  </configSections>  <SECSConnectionSetting configSource="Config\SECSConnectionSetting.config"/>  </configuration> |

1. DLL Name：

|  |
| --- |
| SECSTrxControl.dll |