C#

Arrays and Lists

```
int[] array = new int[] { 1, 2, 3 };
List<int> list = new List<int> { 1, 2, 3 };
```

Lambdas

```
list.Where(item => item == 1)
Action<int> myAct = new Action<int>((myInt) =>
{
  myInt = 2;
});
```

Control Flow

```
for (int i = 0; i <= length; i++)
{
  expression
}
```

```
foreach (var item in collection)
{
  expression
}
```

```
if (statement)
{
  expression
}
else if (statement)
{
  expression
}
else
{
  expression
}
```

```
switch (variable)
{
  case a:
    expression
    break;
  default:
    expression
    break;
}
```

```
while (statement)
{
  expression
}
```

```
try
{
  expression
}
catch (Exception e)
{
  expression
}
finally
{
  expression
}
```

```
do
{
  expression
} while (statement);
```

Synergy .NET

Arrays and Lists

```
DATA array, [#]int, NEW int[#] { 1, 2, 3 }
DATA list, @List<int>, NEW List<int>() { 1, 2, 3 }
```

Lambdas

```
list.Where(LAMBDA (item) {item == 1})
LAMBDA MyLambda(myInt)
BEGIN
  myInt = 2
END
DATA myAct, @Action<int>, NEW Action<int>(MyLambda)
```

Control Flow

```
DATA i, int
FOR i FROM 0 THRU length
BEGIN
  expression
END
```

```
DATA variable, type
FOREACH variable IN collection
BEGIN
  expression
END
```

```
IF (statement) THEN
BEGIN
  expression
END
ELSE IF (statement) THEN
BEGIN
  expression
END
ELSE
BEGIN
  expression
END
```

```
USING variable SELECT
  (a),
    BEGIN
      expression
      EXIT
    END
  (),
    BEGIN
      expression
      EXIT
    END
END
```

```
WHILE (statement) DO
BEGIN
  expression
END
```

```
TRY
BEGIN
    expression
END
CATCH (e, @exception)
  BEGIN
    expression
  END
FINALLY
  BEGIN
    expression
  END
ENDTRY
```

```
DO statement
BEGIN
  expression
END
UNTIL (statement)
```

SYNERGEX

Need more help?

Resource Center:	synergex.com/welcomeRC
Documentation:	synergex.com/docs
Email:	support@synergex.com
Call:	800.366.3472
	916.635.7300

	C#	Synergy .NET

Namespace	`namespace MyNamespace`
	`{`
	`  …`
	`}`
		`NAMESPACE MyNamespace`
		`  …`
		`ENDNAMESPACE`

Class & Constructor	`class MyClass {`
	`  MyClass (string args)`
	`  {`
	`  …`
	`  }`
	`}`
		`CLASS MyClass`
		`  METHOD MyClass`
		`  args, string`
		`  ENDPARAMS`
		`  PROC`
		`  …`
		`  ENDMETHOD`
		`ENDCLASS`

Method	`static int MyMethod(string param)`
	`{`
	`  …`
	`  return 42;`
	`}`
		`STATIC METHOD MyMethod, int`
		`  param, string`
		`  ENDPARAMS`
		`PROC`
		`  MRETURN 42`
		`ENDMETHOD`

Calling a Method	`int argA = 10;` `MyMethod(argA);`	`DATA argA, int, 10` `MyMethod(argA)`

Declaring an Interface	`interface IInterface`
	`{`
	`  void MyMethod();`
	`}`
		`INTERFACE IInterface`
		`  METHOD MyMethod, void`
		`  ENDPARAMS`
		`  ENDMETHOD`
		`ENDINTERFACE`

Implementing an Interface	`class MyClass : IInterface`
	`{`
	`  public void MyMethod()`
	`  {`
	`  throw new Exception();`
	`  }`
	`}`
		`CLASS MyClass IMPLEMENTS IInterface`
		`  PUBLIC METHOD MyMethod, void`
		`  ENDPARAMS`
		`  PROC`
		`  THROW NEW Exception()`
		`  ENDMETHOD`
		`ENDCLASS`

Auto-Implemented Property	`string MyString { get; set; }` `string MyString { get; }`	`READWRITE PROPERTY MyString, string` `READONLY PROPERTY MyString, string`

Full Property	`private string _myString;`
	`public string MyString`
	`{`
	`  get`
	`  {`
	`  return _myString;`
	`  }`
	`  set`
	`  {`
	`  _myString = value;`
	`  }`
	`}`
		`PRIVATE _myString, string`
		`PUBLIC PROPERTY MyString, string`
		`  METHOD get`
		`  PROC`
		`  MRETURN _myString`
		`  ENDMETHOD`
		`  METHOD set`
		`  PROC`
		`  _myString = value`
		`  ENDMETHOD`
		`ENDPROPERTY`

Structure	`struct MyStruct`
	`{`
	`  public string fname;`
	`  public string lname;`
	`  public string phoneNum;`
	`}`
		`CLS STRUCTURE MyStruct`
		`  PUBLIC fname, string`
		`  PUBLIC lname, string`
		`  PUBLIC phoneNum, string`
		`ENDSTRUCTURE`