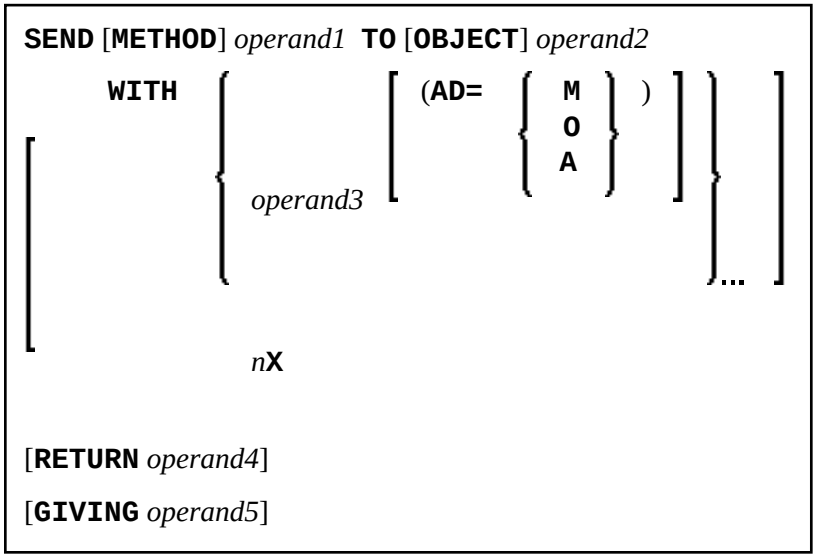
SEND METHOD



This chapter covers the following topics:

- Function
- Syntax Description
- Example

For an explanation of the symbols used in the syntax diagram, see *Syntax Symbols*.

Related Statements: CREATE OBJECT | DEFINE CLASS | INTERFACE | METHOD | PROPERTY

Belongs to Function Group: *Component Based Programming*

Function

The `SEND METHOD` statement is used to invoke a particular method of an object. See the section *NaturalX* in the *Programming Guide* for information on component-based programming.

Syntax Description

Operand Definition Table:

Operand	Possible Structure					Possible Formats													Referencing Permitted	Dynamic Definition
operand1	C	S				A													yes	no
operand2		S																O	no	no
operand3	C	S	A	G		A	U	N	P	I	F	B	D	T	L	C	G	O	yes	no
operand4		S	A			A	U	N	P	I	F	B	D	T	L	C	G	O	yes	no
operand5		S			N					I									yes	no

The formats C and G can only be passed to methods of local classes.

Syntax Element Description:

operand1	**Method-Name:**
	
	operand1 is the name of a method which is supported by the object specified in *operand2*.
	
	Since the method names can be identical in different interfaces of a class, the method name in *operand1* can also be qualified with the interface name to avoid ambiguity.
	
	In the following example, the object `#O3` has an interface `Iterate` with the method `Start`. The following statements apply:
	
	`* Specifying only the method name.`
	`SEND 'Start' TO #O3`
	`* Qualifying the method name with the interface name.`
	`SEND 'Iterate.Start' TO #O3`
	
	If no interface name is specified, Natural searches the method name in all the interfaces of the class. If the method name is found in more than one interface, a runtime error occurs.

operand2	**Object Handle:** The handle of the object to which the method call is to be sent. *operand2* must be defined as an object handle (HANDLE OF OBJECT). The object must already exist. To invoke a method of the current object inside a method, use the system variable `*THIS-OBJECT`.

<table>
<tr><td>operand3</td><td colspan="2">Parameter(s) Specific to the Method:

As operand3 you can specify parameters specific to the method.

In the following example, the object #O3 has the method PositionTo with the parameter Pos. The method is called in the following way:

<pre>SEND 'PositionTo' TO #O3 WITH Pos</pre>
Methods can have optional parameters. Optional parameters need not to be specified when the method is called. To omit an optional parameter, use the placeholder 1X. To omit n optional parameters, use the placeholder nX.

In the following example, the method SetAddress of the object #O4 has the parameters FirstName, MiddleInitial, LastName, Street and City, where MiddleInitial, Street and City are optional. The following statements apply:

<pre>* Specifying all parameters.
SEND 'SetAddress' TO #O4 WITH FirstName MiddleInitial LastName Street City
* Omitting one optional parameter.
SEND 'SetAddress' TO #O4 WITH FirstName 1X LastName Street City
* Omitting all optional parameters.
SEND 'SetAddress' TO #O4 WITH FirstName 1X LastName 2X</pre>
Omitting a non-optional (mandatory) parameter results in a runtime error.</td></tr>
<tr><td>AD=</td><td colspan="2">Attribute Definition:
If operand3 is a variable, you can mark it in one of the following ways:</td></tr>
<tr><td></td><td>AD=O</td><td>Non-modifiable, see session parameter AD=O.</td></tr>
<tr><td></td><td>AD=M</td><td>Modifiable, see session parameter AD=M.

This is the default setting.</td></tr>
<tr><td></td><td>AD=A</td><td>Input only, see session parameter AD=A.</td></tr>
<tr><td></td><td colspan="2">If operand3 is a constant, AD cannot be explicitly specified. For constants AD=O always applies.</td></tr>
<tr><td>nX</td><td colspan="2">Parameter(s) to be Skipped:

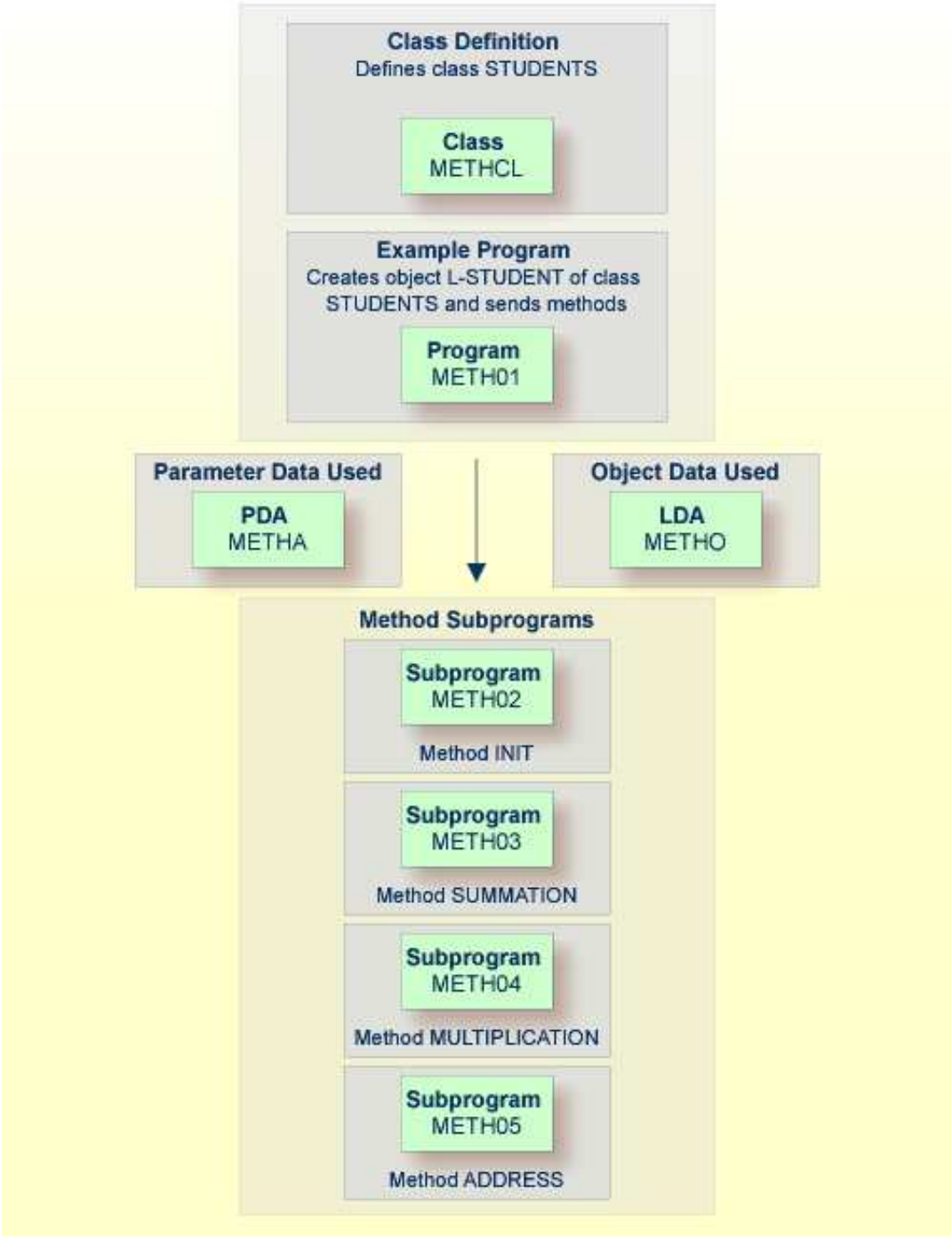
With the notation nX you can specify that the next n parameters are to be skipped (for example, 1X to skip the next parameter, or 3X to skip the next three parameters). This means that for the next n parameters no values are passed to the method.

For a method implemented in Natural, a parameter that is to be skipped must be defined with the keyword OPTIONAL in the dialog's DEFINE DATA PARAMETER statement. OPTIONAL means that a value can - but need not - be passed from the invoking object to such a parameter.</td></tr>
</table>

RETURN *operand4*	**RETURN Clause:** If the `RETURN` clause is omitted and the method has a return value, the return value is discarded. If the `RETURN` clause is specified, *operand4* contains the return value of the method. If the method execution fails, *operand4* is reset to its initial value. **Note:** For classes written in Natural, the return value of a method is defined by entering one additional parameter in the parameter data area of the method and by marking it with `BY VALUE RESULT`. (For more information, see the section *PARAMETER Clause*.) Therefore the parameter data area of a method that is written in Natural and that has a return value always contains one additional field next to the method parameters. This is to be considered when you call a method of a Natural written class and want to use the parameter data area of the method in the `SEND` statement.
GIVING *operand5*	**GIVING Clause:** If the `GIVING` clause is not specified, the Natural run time error processing is triggered if an error occurs. If the `GIVING` clause is specified, *operand5* contains the Natural message number if an error occurred, or zero on success.

Example

The following diagram gives an overview of the programming objects that are used in this example. The corresponding source code and the program output are shown below. Links to the source code of an object are provided in the diagram.



Program METH01 - CREATE OBJECT and SEND METHOD Using a Class and Several Methods:

```
** Example 'METH01':  CREATE OBJECT and SEND METHOD
**                    using a class and several methods (see METH*)
************************************************************************
DEFINE DATA
LOCAL
  USING METHA
LOCAL
1 L-STUDENT HANDLE OF OBJECT
1 #NAME     (A20)
```

```
1 #STREET   (A20)
1 #CITY     (A20)
1 #SUM      (I4)
1 #MULTI    (I4)
END-DEFINE
*
CREATE OBJECT L-STUDENT OF CLASS 'STUDENTS' /* see METHCL for class
*
L-STUDENT.<> := 'John Smith'
*
SEND METHOD 'INIT' TO L-STUDENT             /* see METHCL
     WITH #VAR1 #VAR2 #VAR3 #VAR4
*
SEND METHOD 'SUMMATION' TO L-STUDENT        /* see METHCL
     WITH #VAR1 #VAR2 #VAR3 #VAR4
*
SEND METHOD 'MULTIPLICATION' TO L-STUDENT   /* see METHCL
     WITH #VAR1 #VAR2 #VAR3 #VAR4
*
#NAME  := L-STUDENT.<>
#SUM   := L-STUDENT.<>
#MULTI := L-STUDENT.<>
*
SEND METHOD 'ADDRESS' TO L-STUDENT          /* see METHCL
*
#STREET := L-STUDENT.<>
#CITY   := L-STUDENT.<>
*
*
WRITE 'Name  :' #NAME
WRITE 'Street:' #STREET
WRITE 'City  :' #CITY
WRITE ' '
WRITE 'The summation of     ' #VAR1 #VAR2 #VAR3 #VAR4
WRITE 'is' #SUM
WRITE 'The multiplication of' #VAR1 #VAR2 #VAR3 #VAR4
WRITE 'is' #MULTI
*
END
```

Class Definition METHCL Used by METH01:

```
** Example 'METHCL': DEFINE CLASS (used by METH01)
************************************************************************
* Defining class STUDENTS for METH01
*
DEFINE CLASS STUDENTS
  OBJECT
    USING METHO                 /* Object data for class STUDENTS
  /*
  INTERFACE STUDENT-ARITHMETICS
    PROPERTY FULL-NAME
      IS NAME
    END-PROPERTY
    PROPERTY SUM
    END-PROPERTY
    PROPERTY MULTI
    END-PROPERTY
*
    METHOD INIT
      IS METH02
      PARAMETER USING METHA
```

```
    END-METHOD
    METHOD SUMMATION
      IS METH03
      PARAMETER USING METHA
    END-METHOD
    METHOD MULTIPLICATION
      IS METH04
      PARAMETER USING METHA
    END-METHOD
  END-INTERFACE
*
  INTERFACE STUDENT-ADDRESS
    PROPERTY STUDENT-NAME
      IS NAME
    END-PROPERTY
    PROPERTY STREET
    END-PROPERTY
    PROPERTY CITY
    END-PROPERTY
*
    METHOD ADDRESS
      IS METH05
    END-METHOD
  END-INTERFACE
END-CLASS
END
```

Local Data Area METHO (object data) Used by Class METHCL and Subprograms METH02, METH03, METH04 and METH05:

```
Local     METHO     Library SYSEXSYN                      DBID    10 FNR    32
Command                                                                 > +
I T L  Name                            F Length     Miscellaneous
All -- --------------------------------- - ---------- ------------------------>
     1 NAME                            A         20
     1 STREET                          A         30
     1 CITY                            A         20
     1 SUM                             I          4
     1 MULTI                           I          4
```

Parameter Data Area METHA Used by Program METH01, Class METHCL and Subprograms METH02, METH03 and METH04:

```
Parameter METHA     Library SYSEXSYN                      DBID    10 FNR    32
Command                                                                 > +
I T L  Name                            F Length     Miscellaneous
All -- --------------------------------- - ---------- ------------------------>
     1 #VAR1                           I          4
     1 #VAR2                           I          4
     1 #VAR3                           I          4
     1 #VAR4                           I          4
```

Subprogram METH02 - Method INIT Used by Program METH01:

```
** Example 'METH02': Method INIT (used by METH01)
************************************************************************
DEFINE DATA
PARAMETER
  USING METHA
OBJECT
  USING METHO
```

```
END-DEFINE
*
* Method INIT of class STUDENTS
*
#VAR1 := 1
#VAR2 := 2
#VAR3 := 3
#VAR4 := 4
*
END
```

Subprogram METH03 - Method SUMMATION Used by Program METH01:

```
** Example 'METH03': Method SUMMATION (used by METH01)
************************************************************************
DEFINE DATA
PARAMETER
  USING METHA
OBJECT
  USING METHO
END-DEFINE
*
* Method SUMMATION of class STUDENTS
*
COMPUTE SUM = #VAR1 + #VAR2 + #VAR3 + #VAR4
END
```

Subprogram METH04 - Method MULTIPLICATION Used by Program METH01:

```
** Example 'METH04': Method MULTIPLICATION (used by METH01)
************************************************************************
DEFINE DATA
PARAMETER
  USING METHA
OBJECT
  USING METHO
END-DEFINE
*
* Method MULTIPLICATION of class STUDENTS
*
COMPUTE MULTI = #VAR1 * #VAR2 * #VAR3 * #VAR4
END
```

Subprogram METH05 - Method ADDRESS Used by Program METH01:

```
** Example 'METH05': Method ADDRESS (used by METH01)
************************************************************************
DEFINE DATA
  OBJECT USING METHO
END-DEFINE
*
* Method ADDRESS of class STUDENTS
*
IF NAME = 'John Smith'
  STREET := 'Oxford street'
  CITY   := 'London'
END-IF
END
```

Output of Program METH01:

```
Page      1                                          05-01-17  15:59:04

Name  : John Smith
Street: Oxford street
City  : London

The summation of               1          2          3          4
is          10
The multiplication of          1          2          3          4
is          24
```