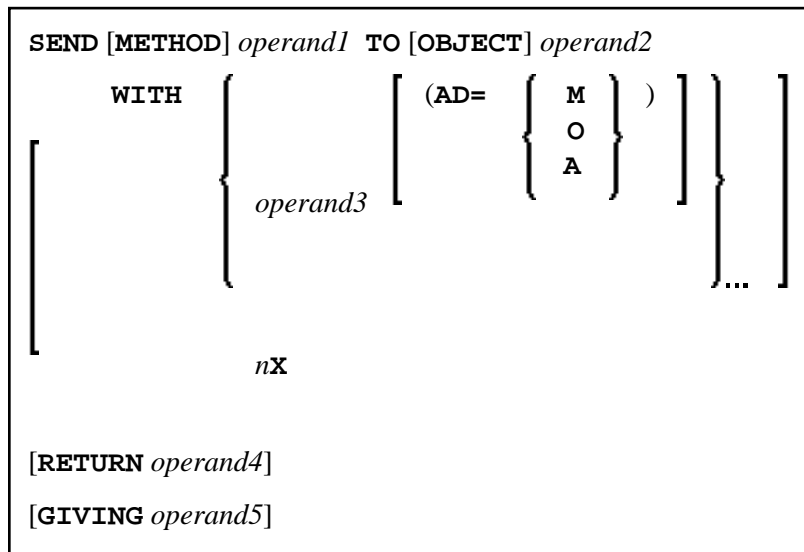


# SEND METHOD



Dieses Kapitel behandelt folgende Themen:

- Funktion
- Syntax-Beschreibung
- Beispiel

Eine Erläuterung der in dem Syntax-Diagramm verwendeten Symbole entnehmen Sie dem Abschnitt *Syntax-Symbole*.

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

Gehört zur Funktionsgruppe: *Komponentenbasierte Programmierung*

## Funktion

Das SEND METHOD-Statement dient dazu, eine bestimmte Methode eines Objekts aufzurufen. Informationen zur komponentenbasierten Programmierung, siehe *NaturalX* im *Leitfaden zur Programmierung*.

## Syntax-Beschreibung

Operanden-Definitionstabelle:

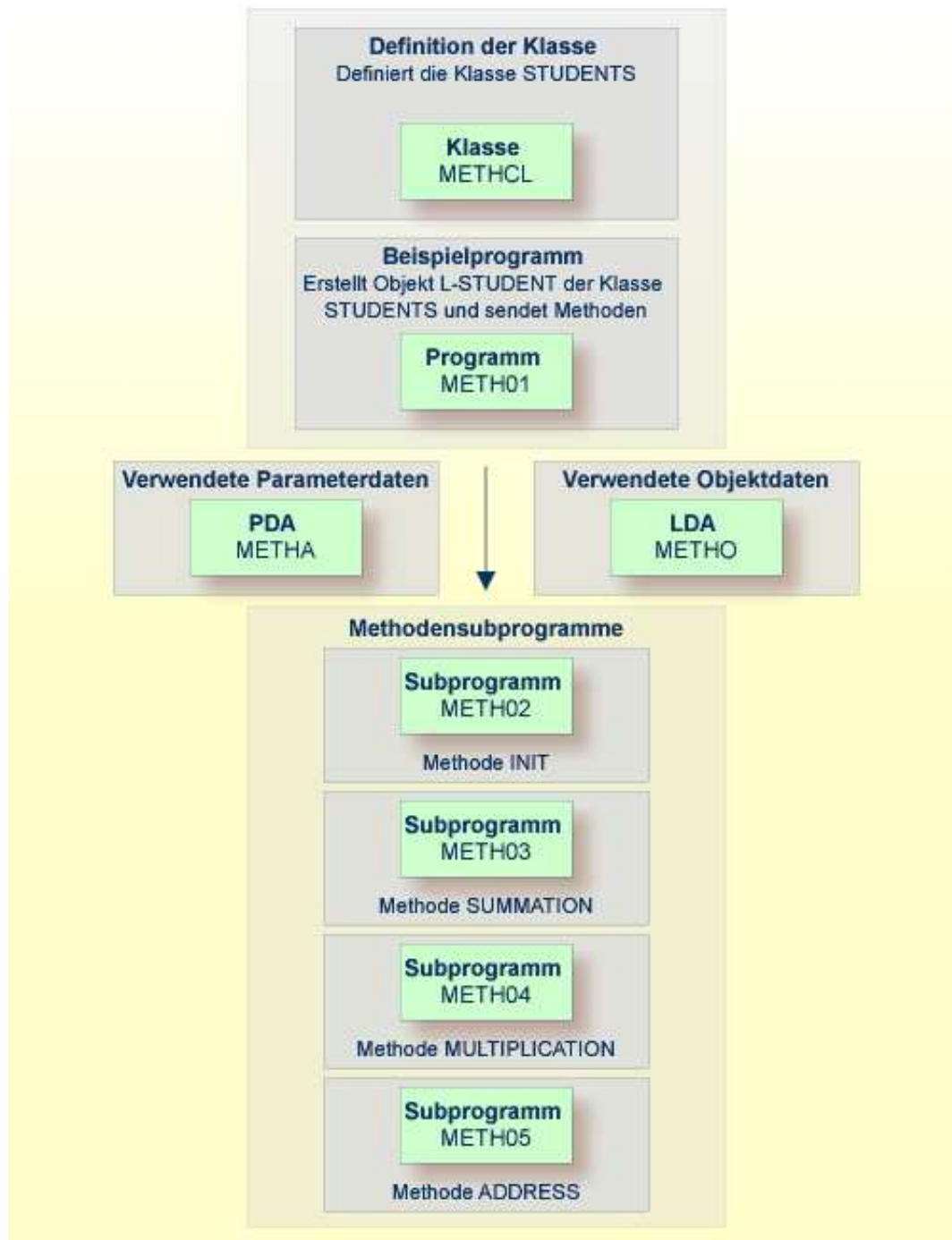


<b>operand3</b>	<p><b>Methodenspezifische Parameter:</b></p> <p>Als <i>operand3</i> können Sie Parameter angeben, die methodenspezifisch sind.</p> <p>Im folgenden Beispiel hat das Objekt #03 die Methode <code>PositionTo</code> mit dem Parameter <code>Pos</code>. Die Methode wird wie folgt aufgerufen:</p> <pre>SEND 'PositionTo' TO #03 WITH Pos</pre> <p>Methoden können optionale Parameter haben. Optionale Parameter brauchen nicht angegeben zu werden, wenn die Methode aufgerufen wird. Um einen optionalen Parameter wegzulassen, verwenden Sie den Platzhalter <code>1X</code>. Um <i>n</i> optionale Parameter wegzulassen, verwenden Sie den Platzhalter <code>nX</code>.</p> <p>Im folgenden Beispiel hat die Methode <code>SetAddress</code> des Objekts #04 die Parameter <code>FirstName</code> (Vorname), <code>MiddleInitial</code> (Mittlere Initiale), <code>LastName</code> (Nachname), <code>Street</code> (Straße) und <code>City</code> (Stadt), wobei <code>MiddleInitial</code>, <code>Street</code> und <code>City</code> optional sind. Es gelten die folgenden Statements:</p> <pre>* Specifying all parameters. SEND 'SetAddress' TO #04 WITH FirstName MiddleInitial LastName Street City * Omitting one optional parameter. SEND 'SetAddress' TO #04 WITH FirstName 1X LastName Street City * Omitting all optional parameters. SEND 'SetAddress' TO #04 WITH FirstName 1X LastName 2X</pre> <p>Wenn ein Pflichtparameter weggelassen wird, führt dies zu einem Laufzeitfehler.</p>						
<b>AD=</b>	<p><b>Attribut-Definition:</b></p> <p>Wenn <i>operand3</i> eine Variable ist, können Sie sie wie folgt kennzeichnen:</p> <table border="1" data-bbox="435 1052 1417 1255"> <tr> <td data-bbox="435 1052 732 1098">AD=O</td> <td data-bbox="732 1052 1417 1098">Nicht modifizierbar, siehe Session-Parameter AD=O.</td> </tr> <tr> <td data-bbox="435 1098 732 1209">AD=M</td> <td data-bbox="732 1098 1417 1209">Modifizierbar, siehe Session-Parameter AD=M. Dies ist die Voreinstellung.</td> </tr> <tr> <td data-bbox="435 1209 732 1255">AD=A</td> <td data-bbox="732 1209 1417 1255">Nur für Eingabe, siehe Session-Parameter AD=A.</td> </tr> </table> <p>Wenn <i>operand3</i> eine Konstante ist, kann AD nicht explizit angegeben werden. Für Konstanten gilt immer AD=O.</p>	AD=O	Nicht modifizierbar, siehe Session-Parameter AD=O.	AD=M	Modifizierbar, siehe Session-Parameter AD=M. Dies ist die Voreinstellung.	AD=A	Nur für Eingabe, siehe Session-Parameter AD=A.
AD=O	Nicht modifizierbar, siehe Session-Parameter AD=O.						
AD=M	Modifizierbar, siehe Session-Parameter AD=M. Dies ist die Voreinstellung.						
AD=A	Nur für Eingabe, siehe Session-Parameter AD=A.						
<b>nX</b>	<p><b>Zu überspringende Parameter:</b></p> <p>Mit der Notation <i>nX</i> können Sie angeben, dass die nächsten <i>n</i> Parameter übersprungen werden sollen (zum Beispiel <code>1X</code>, um den nächsten Parameter zu überspringen, oder <code>3X</code>, um die nächsten drei Parameter zu überspringen). Dies bedeutet, dass für die nächsten <i>n</i> Parameter an die Methode keine Werte übergeben werden.</p> <p>Bei einer in Natural implementierten Methode muss ein zu überspringender Parameter mit dem Schlüsselwort <code>OPTIONAL</code> im <code>DEFINE DATA PARAMETER</code>-Statement des Subprogramms der Methode definiert sein. <code>OPTIONAL</code> bedeutet, dass ein Wert vom aufrufenden Objekt an einen solchen Parameter übergeben werden kann, aber nicht unbedingt muss.</p>						

<b>RETURN <i>operand4</i></b>	<p><b>RETURN-Klausel:</b></p> <p>Wenn die RETURN-Klausel weggelassen wird und die Methode einen Rückgabewert hat, wird der Rückgabewert nicht berücksichtigt.</p> <p>Wenn die RETURN-Klausel angegeben wird, enthält <i>operand4</i> den Rückgabewert der Methode. Wenn die Ausführung der Methode ohne Erfolg abgebrochen wird, wird <i>operand4</i> auf seinen ursprünglichen Wert zurückgesetzt.</p> <p><b>Anmerkung:</b></p> <p>Bei in Natural geschriebenen Klassen wird der Rückgabewert einer Methode durch Eingabe eines zusätzlichen Parameters in der Parameter Data Area der Methode und durch Kennzeichnung mit <code>BY VALUE RESULT</code> definiert. Weitere Informationen siehe Abschnitt <i>PARAMETER-Klausel</i>. Deshalb enthält die Parameter Data Area einer Methode, die in Natural geschrieben ist, und die einen Rückgabewert hat, neben den Methoden-Parametern immer ein zusätzliches Feld. Dies ist zu berücksichtigen, wenn Sie eine Methode einer in Natural geschriebenen Klasse aufrufen und die Parameter Data Area der Methode im SEND-Statement verwenden möchten.</p>
<b>GIVING <i>operand5</i></b>	<p><b>GIVING-Klausel:</b></p> <p>Wenn die GIVING-Klausel nicht angegeben wird, wird die Natural-Laufzeitfehlerverarbeitung angestoßen, wenn ein Fehler auftritt.</p> <p>Wenn die GIVING-Klausel angegeben wird, enthält <i>operand5</i> die Natural-Meldungsnummer, wenn ein Fehler aufgetreten ist, oder Null (0), wenn kein Fehler aufgetreten ist.</p>

## Beispiel

Das folgende Diagramm gibt eine Übersicht über die Programmierobjekte, die in diesem Beispiel benutzt werden. Der entsprechende Sourcecode und die Programm-Ausgabe sind im Folgenden veranschaulicht



### Programm METH01: CTREATE OBJECT und SEND METHOD mit einer Klasse und mehreren Methoden:

```

** 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

```

### Vom Programm METH01 benutzte Klassen-Definition METHCL:

```

** 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 METHO2
    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 (Objektdaten), die von der Klasse METHCL und den Subprogrammen METH02, METH03, METH04 und METH05 benutzt wird:**

Local Command	METHO	Library SYSEXSYN	DBID	10 FNR	32
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, die vom Programm METH01, der Klasse METHCL und den Subprogrammen METH02, METH03 und METH04 benutzt wird:**

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

**Subprogramm METH02 - vom Programm METH01 verwendete Methode INIT:**

```

** 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

```

### Subprogramm METH03 - vom Programm METH01 verwendete Methode SUMMATION:

```

** 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

```

### Subprogramm METH04 - vom Programm METH01 verwendete Methode MULTIPLICATION:

```

** 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

```

### Subprogramm METH05 - vom Programm METH01 verwendete Methode ADDRESS:

```

** 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

```



**Ausgabe des Programms 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			